Adapting DINOv2 Embeddings for Fine-Grained Species Classification and Novelty Detection in Automated Moth Monitoring

Master Thesis

Master of Science in Computing in the Humanities

Johannes Leick

February 12, 2026

Supervisor:

1st: Prof. Dr. Christian Ledig
2nd: Jonas Alle, M.Sc.

Chair of Explainable Machine Learning
Faculty of Information Systems and Applied Computer Sciences
Otto-Friedrich-University Bamberg

Abstract

This master's thesis investigates and evaluates modern computer vision methods for the automated monitoring of moths. Given the context of global biodiversity loss and the need for scalable monitoring systems, this work assesses the suitability of self-supervised embeddings – extracted from the DINOv2 (Small) foundation model – to address challenges in Fine-Grained Visual Classification (FGVC), Domain Generalization (DG), and Novelty Detection (ND).

The AMI (Automated Monitoring of Insects) data source, characterized by high taxonomic diversity and varying image quality (high-resolution GBIF vs. low-resolution trap imagery), effectively representing real-world trap conditions. For this study, the AMI dataset was utilized in an adapted format at the species level, while its technical availability and suitability as a benchmark were simultaneously evaluated.

In the field of FGVC, utilizing embeddings on high-resolution imagery of 91 moth species demonstrates high recognition capabilities: a Multi-Layer Perceptron (MLP) achieved a Balanced Accuracy exceeding 94%, outperforming a K-Nearest Neighbor (KNN) approach by 8 percentage points. The challenge of varying image quality (domain generalization) was evaluated by applying models trained on high-resolution data to low-resolution trap images (covariate shift). This resulted in a significant performance drop, with MLP accuracy falling below 45%, though still maintaining an 8 percentage point lead over the KNN approach.

For novelty detection, a Deep K-Nearest Neighbor (DKNN) classifier was evaluated, utilizing density-based thresholds in embedding space to distinguish between 91 known species (In-Distribution) and 37 unknown (Out-of-Distribution) species. While an AUROC of 0.65 was achieved on high-resolution imagery, the detection performance degraded to near-chance level (AUROC 0.53) when evaluated under the domain shift of low-resolution trap images.

The cropping of rectangular images during feature vector extraction and the subsequent loss of information prompted the creation of a second, padded square image dataset. Compared to the original imagery, the squared dataset exhibited a performance degradation on all classifiers for high-resolution images, whereas a slight improvement was observed for low-resolution imagery.

Overall, the DINOv2 embeddings effectively captured fine morphological differences between 91 moth species within the high-resolution latent space. However, under domain shift, the compactness of the representations was significantly degraded, reducing class separability for both MLP and KNN models. In particular, the density-based DKNN approach to Novelty Detection (ND) failed in this scenario, as it was no longer possible to distinguish between 91 known and 37 unknown species.

Abstract

Diese Masterarbeit evaluiert moderne Computer-Vision-Methoden für das automatisierte Monitoring von Motten. Vor dem Hintergrund des globalen Verlusts der biologischen Vielfalt und der Notwendigkeit skalierbarer Monitoringsysteme, untersucht diese Arbeit die Eignung selbst-überwachter Embeddings des Foundation Models DINOv2 (Small). Der Fokus liegt auf der Bewältigung dreier zentraler Herausforderungen im Bereich des Motten Monitoring: der Fine-Grained Visual Classification (FGVC), der Domain Generalization (DG) und der Erkennung unbekannter Arten (Novelty Detection, ND).

Der AMI-Datensatz (Automated Monitoring of Insects) als Datenquelle repräsentiert durch eine hohe taxonomische Vielfalt und seine variierender Bildqualität (hoch und niedrig auflösend) reale Bedingungen des Motten Monitorings. Für diese Studie wurde dieser in adaptierter Form auf taxonomischer Ebene der Spezies verwendet. Ebenso wurde seine technische Verfügbarkeit und Eignung als Benchmark überprüft.

Im Bereich der hochauflösenden FGVC unter Verwendung von 91 Mottenarten erwies sich die Embeddings als äußerst leistungsfähig: Ein Multi-Layer Perceptron (MLP) erzielte eine Balanced Accuracy von über 94% und übertraf damit einen K-Nearest Neighbor (KNN) Ansatz um 8%. Die Herausforderung der variierenden Bildqualität (Domain Generalization) wurde auf hochauflösenden Trainingsdaten und zu klassifizierende niedrigauflösende Fallenbildern (Covariate Shift) untersucht. Dies führte zu einem deutlichen Leistungseinbruch, wobei die Genauigkeit des MLP auf unter 45% sank, damit aber weiterhin mit 8% den KNN-Ansatz übertraf.

Zur Detektion neuer Arten (ND) wurde ein Deep K-Nearest Neighbor (DKNN) Classifier evaluiert, der mittels dichte-basierter Schwellenwerte im latent Space zwischen 91 bekannten Spezies (In-Distribution) und 37 unbekannten Spezies (Out-of-Distribution) unterscheidet. Während auf hochauflösenden Bildern ein AUROC-Wert von 0,65 erreicht wurde, fiel die Detektionsleistung unter Hinzunahme des Domain Shifts bei niedrigauflösende Fallenbilder auf nahezu Zufallsniveau (0.53) zurück.

Das Zuschneiden (cropping) rechteckiger Bilder bei der Codierung zu Embeddings und damit einhergehender Informationsverlust war Grund, einen zweiten erweiterten quadratischen Bilddatensatzes zu erstellen. Vergleichend mit dem ursprünglichen Bilddaten zeigte sich bei allen Classifiern bei hochauflösenden Bildern eine Verschlechterung der Ergebnisse, hingegen eine geringfügige Verbesserung bei niedrig auflösenden Bildern.

Insgesamt konnten die DINOv2 Embeddings feine morphologische Unterschiede zwischen 91 verschiedenen Motten Spezies aus hochauflösenden Bildern im Latent Space kompakt und unterscheidbar abbilden. Bei einem zusätzlichen Domain Shift mit niedrig auflösenden Bildern aus Mottenfallen reduzierte sich die kompakte Abbildung deutlich und damit die Differenzierbarkeit der Klassen des MLP und KNN. Hier scheiterte insbesondere auch der dichte-basierten DKNN-Ansatz zur ND, der keine Trennung zwischen 91 bekannten und 37 unbekannten Spezies mehr vornehmen konnte.

Contents

List of Figures

List of Tables

List of Acronyms

AI	Artificial Intelligence
AMI	Automated Monitoring of Insects
AUPR	Area Under the Precision-Recall Curve
AUROC	Area Under the Receiver Operating Characteristic
BMFTR	Federal Ministry of Research, Technology, and Space
CLIP	Contrastive Language-Image Pretraining
CPU	Central Processing Unit
CS	Covariate Shift
CSV	Comma-Separated Values
DG	Domain Generalization
DKNN	Deep K-Nearest Neighbor
DNN	Deep Neural Networks
DSS	Decision Support Systems
EU	European Union
FEdA	Research initiative for the preservation of biodiversity
FGIR	Fine-Grained Image Recognition
FGVC	Fine-Grained Visual Classification
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
FPR95	False Positive Rate at a True Positive Rate of 95%
GBIF	Global Biodiversity Information Facility
GDPR	General Data Protection Regulation
GPU	Graphics Processing Unit
HDD	Hard Disk Drive
HTTP	Hypertext Transfer Protocol
IAS	Invasive Alien Species
ID	In-Distribution
IPM	Integrated Pest Management
IoT	Internet of Things
KNN	K-Nearest Neighbor
ML	Machine Learning
MLP	Multi-Layer Perceptron
ND	Novelty Detection
OOD	Out-of-Distribution
PA	Precision Agriculture
PCA	Principal Component Analysis
PIL	Python Imaging Library
RAM	Random Access Memory
RGB	Red Green Blue
SSL	Secure Sockets Layer

TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
URL	Uniform Resource Locator
VRAM	Video Random Access Memory
ViT	Vision Transformer

1 Introduction

1.1 Relevance of Monitoring of Insects

Ecological Context

> "Almost one-third of all species on the Red Lists are at risk, meaning they are critically endangered or endangered; about 3% are already considered extinct. Many reptile and amphibian species, as well as numerous insect species and other arthropods, are endangered. However, for the latter and many other species groups, there is a lack of data necessary for a reliable assessment."[1] (Wirth et al., 2024a, p. 37, own translation)

This is stated in the Faktencheck Artenvielfalt (Biodiversity Fact Check), published in Germany in 2024. It represents the first comprehensive assessment of biological diversity in Germany and outlines various courses of action for its conservation and sustainable use.

Insects are estimated to account for more than half of all animal species worldwide. Nevertheless, data on many insect populations remains sparse. This is due, in part, to the high taxonomic effort required and the limited number of experts available for reliable identification. Comprehensive monitoring and systematic inventorying of insect species are therefore essential prerequisites for better assessing the extent and rate of species extinction and for developing effective countermeasures.

The significance of the extinction of individual insect species must not be underestimated, as such losses can jeopardize the stability of entire ecosystems. For example, nocturnal moths, which contribute to nocturnal pollination, play a central role in maintaining ecological processes. Conversely, certain insect species can exhibit mass outbreaks under changing environmental conditions, posing a risk to agricultural production. Ultimately, these ecological systems constitute the fundamental basis for human life and economic activity (Jain et al., 2024, p. 1[2]; Bjerge et al., 2021, pp. 1–2).

The publication Faktencheck Artenvielfalt (Biodiversity Fact Check) further highlights that emerging technologies will fundamentally transform biodiversity monitoring. Alongside various methodologies, automated image recognition for camera traps (Bjerge, Mann, and Høye, 2022) and the use of smartphone applications such as Flora Incognita (www.floraincognita.com) and iNaturalist (www.inaturalist.org) are identified as particularly promising approaches.

[1] "Fast ein Drittel aller Arten in den Roten Listen sind bestandsgefährdet, das heißt, sie sind vom Aussterben bedroht oder stark gefährdet; etwa 3 % gelten bereits als ausgestorben. Stark gefährdet sind viele Reptilien- und Amphibienarten sowie zahlreiche Insektenarten und andere Gliedertiere. Für Letztere und viele weitere Artengruppen fehlt jedoch die Datengrundlage, die für eine verlässliche Einstufung notwendig ist."

[2] This master's thesis is based on the arXiv preprint, as it was the only version preceived at the time of implementation. However, the dataset and findings have been officially published at ECCV 2024 and should be cited as a supplement: (Jain et al., 2025)

These technologies can effectively complement conventional monitoring methods by significantly increasing their temporal and spatial resolution, expanding the range of recorded species, and broadening the participation in data collection (Wirth et al., 2024a, p. 158). Developments in the field of artificial intelligence – particularly machine learning and deep learning – act as a key driver in this process, enabling the establishment of new, automated monitoring systems.

Insect image classification represents just one of many application domains where these technological advancements are already being successfully utilized (Wirth et al., 2024a, pp. 158, 373, 662). Accordingly, the Faktencheck Artenvielfalt (Biodiversity Fact Check) calls for automated monitoring methods to be rapidly developed towards operational readiness and standardization, as they uncover new facets of biological diversity while making relevant environmental variables and influencing factors measurable (Wirth et al., 2024b, p. 82).

Agricultural Context A critical need for insect monitoring is also recognized within the agricultural sector. Global warming influences not only the cultivation of crops but also the dynamics and distribution of associated agricultural pests.

As a result of changing climatic conditions, insects will be subject to shifts in their geographical distribution, population dynamics, and interactions with natural enemies. The responses of insect populations will vary significantly across regions, driven by the diversity of species, the availability of host plants, and a multitude of climatic variables.

Furthermore, the expansion of international travel, the global agricultural sector, and global trading systems since the 2000s have accelerated the spread of invasive species[3]. These species are considered a threat to biodiversity and can result in high economic costs for the agricultural industry.

It is therefore expected that farmers will increasingly face new and more severe pest infestations. To address these challenges and mitigate risks to food security, effective insect monitoring is considered a fundamental prerequisite. Particularly within the framework of Integrated Pest Management (IPM)[4], monitoring strategies play a crucial role in the early detection and management of potential pest outbreaks (Skendžić et al., 2021).

It is therefore unsurprising that technological advancements in deep learning are also fundamentally transforming insect monitoring within agriculture (Oubabas et al., 2024, p. 1). In a review article on trap development and deep learning-based

[3]Invasive species – also referred to as Invasive Alien Species (IAS) by Skendžić et al., 2021 – are species introduced by humans, either intentionally or unintentionally (e.g., via food, crops, ornamental plants, or livestock), beyond their natural habitats. Invasive insects typically act as pests in agriculture, storage, forestry, households, or buildings, and often serve as vectors for various diseases or parasites. (Skendžić et al., 2021, p. 14)

[4]Integrated Pest Management (IPM) leverages monitoring data to ensure that pesticides are applied only when necessary and as targeted as possible. The primary objective is to minimize environmental impact while optimizing crop yields and economic viability. (Skendžić et al. 2021, p. 20; Passias et al., 2024, p. 31)

pest detection, Passias et al. (2024) describe how deep learning methods, integrated with other technologies – specifically the Internet of Things (IoT) – are significantly enhancing traditional monitoring practices on agricultural land (Passias et al., 2024, p. 323).

Camera-based traps enable the detailed identification of insect species, provide precise data on infestation levels, and facilitate the observation of behavioral patterns. Through automated data processing, real-time information become available at an early stage, enabling informed decision-making within pest management. These new monitoring methods thus significantly expand the scope for action in pest control while reducing the effort required for conventional, labor-intensive monitoring.

The European Union (EU) estimates that such pest monitoring approaches could facilitate a reduction in pesticide use of 20% to 50% by 2030. This provides further impetus for the transition toward Precision Agriculture (PA), which aims to enhance both productivity and ecological sustainability (Passias et al., 2024; Oubabas et al., 2024).

Technological Potential and Scalability The primary potential of emerging technical developments in insect monitoring lies in scalability. The abundance of different insect species and their variations in different geographical regions requires a large number of specific experts for monitoring, who are not available for extensive recording. With the advent of high-resolution cameras and low-cost sensors, as well as the aforementioned data processing methods of machine learning and deep learning, insect monitoring can be automated and scaled (Jain et al., 2024, pp. 1-3).

1.2 Existing Moth Projects

Through the research initiative for the preservation of biodiversity (FEdA, 2025), the federal ministry of research, technology, and space (BMFTR, 2023) is already funding the research area *Artificial Intelligence Methods as an Instrument for Biodiversity Research*. These projects were initiated with a one-year concept phase, followed by a subsequent research and implementation phase lasting from one to three years. Currently, nine projects have progressed to the implementation stage.

LEPMON One such project is LEPMON, which aims to monitor moth biodiversity (Lepidoptera) using automated camera traps and artificial intelligence (AI). It is a comprehensive initiative that encompasses the development of automated light traps, automated species identification via AI, and database management, while also incorporating Citizen Science to foster public engagement. The project successfully concluded its initial field trials in 2025 and is scheduled to be completed by December 21, 2027 (Analyse des Wandels der Biodiversität (LIB), 2025).

Automated Monitoring of Insects (AMI) A similar international initiative, known as Automated Monitoring of Insects (AMI), is already further progressed. It

encompasses 11 initiatives from four countries working collectively to advance automated insect monitoring. Similar to the LEPMON project, development is partitioned into distinct work areas distributed across various organizations (Consortium-Steering-Committee, 2025). One outcome is the UKCEH AMI automated light trap, which is currently deployed in over 30 countries (Bjerge et al., 2021). Additionally, "Antenna - The Insect Data Platform" (www.insectai.org) was established under the leadership of the Quebec Artificial Intelligence Institute (Mila). This platform facilitates interdisciplinary collaboration between entomologists, ecologists, and computer scientists. The goal is to classify large numbers of insects using images in order to develop and provide a growing dataset. As a result, the AMI dataset, a moth dataset containing over 2.5 million images with over 5,000 classes, was published in 2024. According to the authors, it represents the largest fine-grained insect classification dataset currently available (Jain et al., 2024). While currently focused on moths, the project plans to broaden its taxonomic scope to include other insect families.

1.3 Rationale for Moth Monitoring

The projects mentioned above focus primarily on nocturnal Lepidoptera, commonly referred to as moths. There are several key reasons for this focus:

- Moths represent approximately one-fifth of all known insect species worldwide. Within the order Lepidoptera (butterflies and moths), they account for 91% of all described species – with estimates ranging from 135,700 to 160,000 species – while the remaining 9% are butterflies (Heppner 2008; Jain et al. 2023, p. 1).
- Moths further exhibit the advantage that most moth species are visually distinct at both the genus and species level. Additionally, they react strongly to ultraviolet light, allowing them to be efficiently attracted and captured using UV-based light traps (Jain et al., 2023, p. 1).
- Moths also play an important role as pests in agriculture. Among the 85 agriculturally relevant insect pests identified by Passias et al., 2024, seven of the twelve most frequently mentioned species can be categorized as moths (Passias et al. 2024, p. 326; Bjerge et al. 2021, pp. 1–2).

These factors underscore why moths are well suited candidates for the development and validation of automated monitoring systems. Furthermore, current research aims to establish fundamental methodologies through moth classification that can subsequently be generalized and applied to other insect taxa.

1.4 Core Components of Automated Moth Monitoring

The development of automated monitoring systems for moths can be divided into three primary functional areas:

Image Acquisition

The first area focuses on the technical development and deployment of automated moth traps. These systems utilize UV light to attract nocturnal insects, which are then captured as digital images by integrated camera sensors.

Image Analysis

The second area involves the processing of the captured imagery. This stage encompasses object detection to locate insects within the image frame, followed by classification. This process typically starts at a binary level (moth vs. non-moth) and progresses to fine-grained classification to identify specific species.

Data Interpretation

The third area focuses on the synthesis of the classification results. Data derived from the image analysis are utilized to calculate ecological metrics, such as species richness, relative abundance, or shifts in population structures. Furthermore, long-term datasets enable the analysis of spatiotemporal trends and the conduct of longitudinal ecological studies.

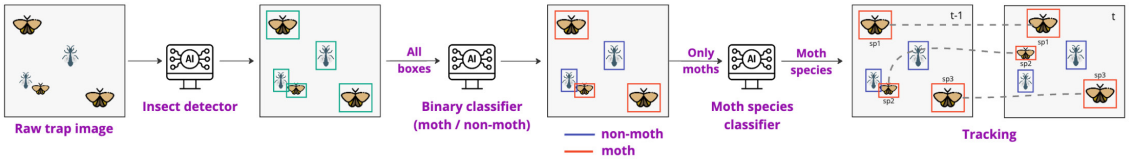


Figure 1: Machine learning workflow for Moth Classification (Jain et al., 2023, p. 10)

Each of these functional areas can be further decomposed into more granular stages. For instance, Jain et al. (2023) provides a detailed breakdown of the insect classification process, as illustrated in Figure 1:

Insect Detector

First, the process focuses on object detection. In this stage, objects within the trap imagery are automatically localized and saved as individual image crops. This results in a large number of smaller images, each containing a potential candidate, such as a moth.

Moth Classifier

Subsequently, only the crops relevant for further analysis are processed. A binary classification is employed to categorize all detected objects as either “moth” or “non-moth”.

Species Classifier

Objects classified as moths then undergo fine-grained classification

at the species level. This stage distinguishes between a vast array of different moth species.

Tracking The whole pipeline is supplemented by a tracking module that monitors all classified moths on the trap screen. The objective is to mitigate double counting by ensuring that individual insects are not recorded multiple times (Jain et al. 2023; Jain et al. 2024, p. 6-7).

The performance of individual processing stages directly and indirectly impacts the quality of downstream stages. For instance, moth traps that generate high-resolution imagery provide a superior foundation for precise object detection, reliable binary classification, and robust fine-grained classification compared to devices with lower image resolutions. Likewise, an effective binary classification stage ensures that only objects belonging to the target moth species reach the subsequent fine-grained classification stage. This filters out non-target insects that might otherwise be misclassified as moths, thereby ensuring that the species classifier only operates on data it is designed to handle.

These interdependencies illustrate that automated insect monitoring can be decomposed into numerous sequential subtasks, each posing unique challenges. This modularization enables the independent development, optimization, and implementation of individual components (Jain et al., 2023, p. 2).

1.5 Overview and Research Scope

This work addresses challenges in automated moth monitoring and situates itself within the second sub-area: image analysis, especially moth species classification.

To evaluate methodologies in realistic application scenarios, it is essential to utilize the outputs of the preceding processing stages. This is facilitated by the release of the AMI dataset (Jain et al., 2024), which comprises the results of the aforementioned stages, including the initial image acquisition via light traps and subsequent object detection. The resulting image crops are provided as a binary dataset ("moth" vs. "non-moth") and a fine-grained dataset featuring 5,364 distinct moth species. Furthermore, the dataset includes standardized splits for training, validation, and testing, which can be directly utilized for model development and evaluation.

The curated and preprocessed AMI dataset allows various challenges of moth classification under field conditions to be addressed in isolation. Furthermore, it serves as a standardized benchmark for comparing the performance of different classification methodologies. The objective of the AMI dataset is to foster development and stimulate competition for high-performance models in automated moth monitoring. (Jain et al., 2024)

The AMI dataset thus provides the foundation for this master's thesis to address challenges in visual moth classification. These challenges are situated within the fields of Fine-Grained Visual Classification (FGVC), Domain Generalization (DG),

and Novelty Detection (ND). This work investigates these areas by applying established classification methods – specifically K-Nearest Neighbor, Multi-Layer Perceptron, and Deep K-Nearest Neighbor.

All methods are implemented based on embeddings extracted from the vision foundation model DINOv2 (Oquab et al., 2024). By evaluating their performance in the context of moth classification, this research aims to contribute to the advancement of the automated moth monitoring field. The specific research questions addressed in this work are as follows:

1. During the verification of the complete retrievability of the AMI dataset, particular focus is placed on the AMI-GBIF sub-dataset, as its images must be sourced from various external sources. This distributed nature may compromise the integrity of the AMI dataset as a standardized benchmark for future research and raises the question:

   To what degree can the AMI-GBIF subset be reconstructed, and to what extent does it remain viable as a standardized benchmark in the long term?

2. All classifiers employed in this study leverage embeddings extracted from the DINOv2 foundation model. For the generation of these feature vectors, images are processed via a preprocessing pipeline. In this stage, non-square images are cropped, which results in a loss of peripheral image data. To mitigate this information loss, three distinct padding strategies were investigated to assess their impact on the resulting embeddings. These strategies extend rectangular images into a square format, thereby circumventing the need for cropping. The objective is to identify a padding method that minimizes distortion of the embeddings while ensuring that the visual information of the moth from the original image is fully preserved within the extracted representation. This raises the question:

   Which padding strategy is most suitable for converting rectangular moth images into the square input format of DINOv2 without adversely impacting embedding extraction?

3. Based on the previously described padding strategies, a secondary square image dataset was generated to complement the original AMI dataset. Both datasets are utilized for all subsequent visual moth classification tasks, enabling a comparative assessment of the impact of image padding on classification performance. This raises the question:

   Can the selected padding strategy be used to create a dataset that minimizes information loss, thereby yielding higher-quality embeddings and improving classification performance?

4. A crucial stage in the automated monitoring of moths is the fine-grained visual classification. The primary challenge lies in the accurate visual identification of a vast number of morphologically similar moth species. To address this, two

classification methodologies – K-Nearest Neighbor and Multi-Layer Perceptron – are evaluated using high-resolution training and test datasets comprising 91 distinct moth species, which raises the following question:

How effectively can KNN and MLP classifiers, leveraging DINOv2 embeddings, distinguish between 91 moth species?

5. Beyond the high number of species, a significant challenge in moth monitoring involves the ability to generalize across a domain shift. This shift occurs between the high-resolution training imagery and the low-resolution field data captured by automated moth traps. To address this, the aforementioned KNN and MLP classifiers are evaluated regarding their domain generalization capabilities, utilizing low-resolution trap images. This prompts the following question:

   How robustly do the classifiers respond to the covariate shift between high-resolution training imagery and low-resolution trap images?

6. Another challenge arises from invasive species and climate-induced distribution shifts, which lead to changes in insect habitats. To effectively capture these dynamics, automated monitoring systems must be able to distinguish between regionally known species and novel (Out-of-Distribution) species. For this purpose, a Deep K-Nearest Neighbor approach is evaluated for its performance in novelty detection. The DKNN approach was evaluated based on high-resolution data comprising 91 distinct known and 37 distinct unknown moth species, which raises the following question:

   How reliably can the DKNN method discriminate between 91 known species (ID) and 37 unknown species (OOD), leveraging DINOv2 embeddings?

7. In a final experiment, novelty detection is evaluated within the framework of domain generalization. This synthesis transfers the theoretical challenges of moth classification into a realistic application context. The objective is to robustly distinguish between In-Distribution and Out-of-Distribution species, even when the system is confronted with the domain shift caused by low-resolution field imagery. To this end, the Deep K-Nearest Neighbor classifier is assessed for its generalization capabilities under these combined constraints, utilising low-resolution trap images. This prompts the following question:

   Can the DKNN method detect unknown species even when they are captured in low-quality trap imagery?

To address the research questions outlined above, this thesis is organized into seven chapters. *Chapter 2* delineates the challenges and methodological approaches of this study. First, it derives the challenges of fine-grained visual classification, domain generalization, and novelty detection directly from the requirements of automated moth monitoring. It then details the AMI dataset, including its inherent challenges, the padding experiments, and the subsequent generation of the resized dataset.

Finally, it outlines the classification methodologies – KNN, MLP, and DKNN – to be evaluated within these specific domains.

Chapter 3 covers the acquisition of the AMI dataset and the extraction of subsets for model training and testing. It further details the implementation of the padding experiments and the creation of the resized dataset, concluding with the embedding extraction process. *Chapter 4* focuses on the implementation and testing of the classifiers used: KNN, MLP, and DKNN. *Chapter 5* presents the comprehensive results from the preceding chapters. *Chapter 6* provides a discussion and assessment of these results and puts forward a number of suggestions for future research. Finally, *Chapter 7* concludes the thesis with a summary of the work and answers the central research questions.

2 Challenges and Experimental Setup

This chapter delineates the challenges and methodological approaches addressed in this thesis in order to answer the previously defined research questions.

2.1 Challenges in Automated Moth Monitoring

2.1.1 Fine-Grained Visual Classification

A central challenge defined by the AMI dataset and addressed in this thesis is the aforementioned task of visual species classification. This domain is particularly demanding due to the immense taxonomic diversity of moths. Globally, there are an estimated 135,700 to 160,000 distinct moth species. Even though this number is significantly reduced in regional application contexts, thousands of species still require accurate classification (Jain et al., 2024; Jain et al., 2023, p. 5). For example, the LEPMON project specifies that Germany alone hosts over 1,100 species of nocturnal large butterflies out of a total of more than 4,000 nocturnal moth species (Analyse des Wandels der Biodiversität (LIB), 2025).

In the field of machine learning, this type of classification is formally known as Fine-Grained Image Recognition (FGIR) or Fine-Grained Visual Classification (FGVC). The objective is to differentiate between subcategories that belong to a common supercategory. This is particularly challenging because the shared higher-level category leads to high inter-class similarity. Consequently, the system must detect and correctly map subtle discriminative features (Chu, Ye, and Qian, 2024, p. 19640). The fundamental challenge for classifiers is to identify nuanced patterns at the pixel level and represent them as distinctive embeddings within a high-dimensional feature space to assign them to the corresponding semantic classes. Since the dataset provides annotated image data, strong supervision methods can be employed, which have proven to achieve high Accuracy in large-scale data scenarios (Chu, Ye, and Qian, 2024, pp. 19641, 19647–19648).

2.1.2 Domain Generalization – Covariate Shift

A second challenge identified in the AMI dataset publication and addressed in this thesis concerns the discrepancy in image quality between the data used for training and the field imagery captured by automated moth traps.

Typically, classification models are trained using high-resolution images, which are extensively available through the Global Biodiversity Information Facility (GBIF)[5]. This database provides annotated data from museums and citizen science initiatives.

[5]GBIF is an international network of countries and organizations maintaining a standardized database to record all forms of life on Earth (biodiversity). Its goal is to provide a centralized, uniform schema for biodiversity observations and ensure they are publicly accessible. https://www.gbif.org/what-is-gbif (visited on 08/02/2026)

These high-resolution images serve as a foundation for creating regional datasets tailored to training classifiers (GBIF, 2024; Jain et al., 2024, p. 6).

However, imagery captured by moth traps exhibits a significant degradation in quality compared to the high-fidelity images provided by GBIF. Moth traps are compact systems with performance limited by budgetary constraints and power consumption. Consequently, low-cost sensors with limited resolution, such as webcams, are frequently employed. These cameras are mounted at a fixed distance from the trap screen. Small insects occupy only a minimal portion of the field of view; once these are localized and extracted via object detection, the resulting low-resolution crops provide limited information for classification.

Environmental and behavioral factors further complicate the task. Insects may overlap, leading to occlusions, or settle near the edges of the frame, resulting in truncated detections. Furthermore, movement during the exposure period causes motion blur, while external factors such as spider webs introduce visual artifacts that further diminish image quality. As a result, the low-resolution, noisy field data from moth traps differs significantly from the curated GBIF datasets used for training (Jain et al., 2024, pp. 5-6).

This issue is also reflected in machine learning paradigms that operate under the closed-world assumption. This paradigm assumes that the training and test data are independent and identically distributed (i.i.d.). In real-world applications, however, this assumption is frequently violated, as models are often trained on datasets that differ significantly from the data encountered during inference. The open-world assumption addresses this discrepancy by accounting for the potential occurrence of Out-of-Distribution data within the sets to be classified (Yang et al., 2024, p. 5635).

According to Yang et al. (2024), OOD data can be principally categorized into two types: covariate shift and semantic shift. Covariate shift refers to a discrepancy in the distribution of the input data, while the underlying semantic labels remain constant.

An example for that is the variation in the visual representation (domains) of the same moth species – such as a photograph compared to a hand-drawn illustration, or a high-resolution image versus a low-resolution capture. Despite these differing modalities, the semantic category remains identical (Yang et al., 2024, pp. 5636–5638). The challenge for a classifier is to achieve semantic invariance, ensuring that the correct category is assigned regardless of variations in resolution, style, or image quality.

This classification capability is referred to as domain generalization (DG)[6]. The term domain generalization emphasizes that no data from the unseen target domains or

[6]This capability is often described as a model's generalization ability or robustness: "robust model should generalize to examples with covariant shift; a weak model should reject them." (Yang et al., 2024, p. 5641)

alternative representations are used during training. Consequently, the model must demonstrate inherent generalization performance[7] (Yang et al., 2024, p. 5642).

2.1.3 Novelty Detection – Semantic Shift

A third challenge addressed in this work is the potential appearance of previously undocumented or novel moth species in trap imagery. The probability of encountering such species is particularly high given the ecological impacts of climate change. Changes in insect populations and significant range shifts are leading to the northward expansion of various species. Furthermore, the introduction of non-native species via international trade presents an ongoing risk (Skendžić et al., 2021, pp. 10, 14, 23).

To reliably monitor dynamics within insect populations, automated moth traps must be capable of detecting novel, previously undocumented species. It is crucial that such species are identified as unknown rather than being misclassified into a pre-existing category. This requirement is equally critical for the early detection of emerging agricultural pests (Chen and Rolnick, 2023).

The identification of novel species (shifting insekts) aligns with the second variant of OOD data: semantic shift (Yang et al., 2024, pp. 5637–5638). While covariate shift – as previously described – deals with variations within the input space of known classes that should be mapped to the same semantic category, semantic shift concerns cases where images cannot be assigned to any pre-defined class. These images contain moth species that the model never encountered during training and, consequently, cannot classify correctly[8]. The capability to identify these unknown instances is known as novelty detection (ND)[9].

Novelty detection can be described as a binary classification task. One class encompasses all categories the classifier encountered during training; these are designated as In-Distribution. Conversely, all novel, unknown classes are defined as OOD, as they lie outside the learned training distribution. Consequently, the objective of novelty detection is to differentiate OOD samples from ID samples through a binary decision process.

In the context of moth monitoring, these novel species often exhibit a high degree of similarity to the target classes due to shared morphological characteristics. This fine-grained relationship significantly complicates the task, as the distinction between a known species and a novel one relies on subtle visual cues.

[7] If samples from the target domains are available during the training phase, the process is instead referred to as domain adaptation.

[8] Because conventional classifiers are designed to assign inputs only to known categories, such novel samples would inevitably be misclassified into one of the existing classes.

[9] Yang (2024) defines this as follows: "Novelty detection aims to detect any test samples that do not fall into any training category. The detected novel samples are usually prepared for future constructive procedures, such as more specialized analysis, or incremental learning of the model itself." (Yang et al., 2024, p. 5639)

2.2 Dataset and Embeddings

The following section provides a detailed description of the AMI dataset, covering its characteristics and utilization within this study. Furthermore, it details the padding experiments conducted to mitigate information loss during the embedding extraction process. Finally, the creation and purpose of the resized data set are outlined.

2.2.1 Original – AMI Dataset

Before detailing the methodologies used to address the classification tasks, the foundational AMI dataset is presented. This dataset comprises several sub-datasets designed to simulate the diverse challenges of moth monitoring within a real-world application. The core of this study focuses on the fine-grained components of the AMI dataset:

AMI-Traps Fine-Grained: This subset consists of low-resolution images of 516 different moth species, captured directly by automated traps in the field.

AMI-GBIF Fine-Grained: To complement the Trap Fine-Grained data, this subset provides high-resolution images of 5,364 species, sourced from the Global Biodiversity Information Facility (GBIF).

Together, these datasets enable the development and evaluation of classifiers specifically tailored to identify a wide variety of moth species under varying image qualities.

In addition to the fine-grained data, the AMI dataset contains a binary classification subset, which is not utilized in this thesis. This subset follows a similar structure, consisting of low-resolution trap images (AMI-Traps Binary) and high-resolution GBIF counterparts (AMI-GBIF Binary). While the fine-grained data focuses on distinguishing between moth species, the binary subset is designed to differentiate between moths and non-moth entities (such as other insects), thereby facilitating the development and evaluation of binary detection tasks.

The following sections provide a detailed account of the relevant AMI-Traps Fine-Grained and AMI-GBIF Fine-Grained subsets, including their origin and scope. This description details the processing steps executed by Jain et al., 2024, with a focus on the subsets relevant to the present thesis (Jain et al., 2024, p. 6–9).

AMI-Traps The sub-dataset containing low-resolution imagery from moth traps. It comprises images collected from 22 individual traps deployed across three regions: Northeastern of North America (NE-America), including data from Quebec (Canada) and Vermont (USA); Western Europe (W-Europe), encompassing the United Kingdom and Denmark; and Central America (C-America), represented by data from Panama. (Jain et al., 2024, p. 6)

The cameras were configured for motion-triggered acquisition and captured photos at fixed 10-minute intervals. To mitigate visual redundancy and ensure a diverse

training set, an annotated subset was curated from the raw footage. In total, 2,893 trap images were aggregated across all camera systems (Jain et al., 2024, p. 6-7).

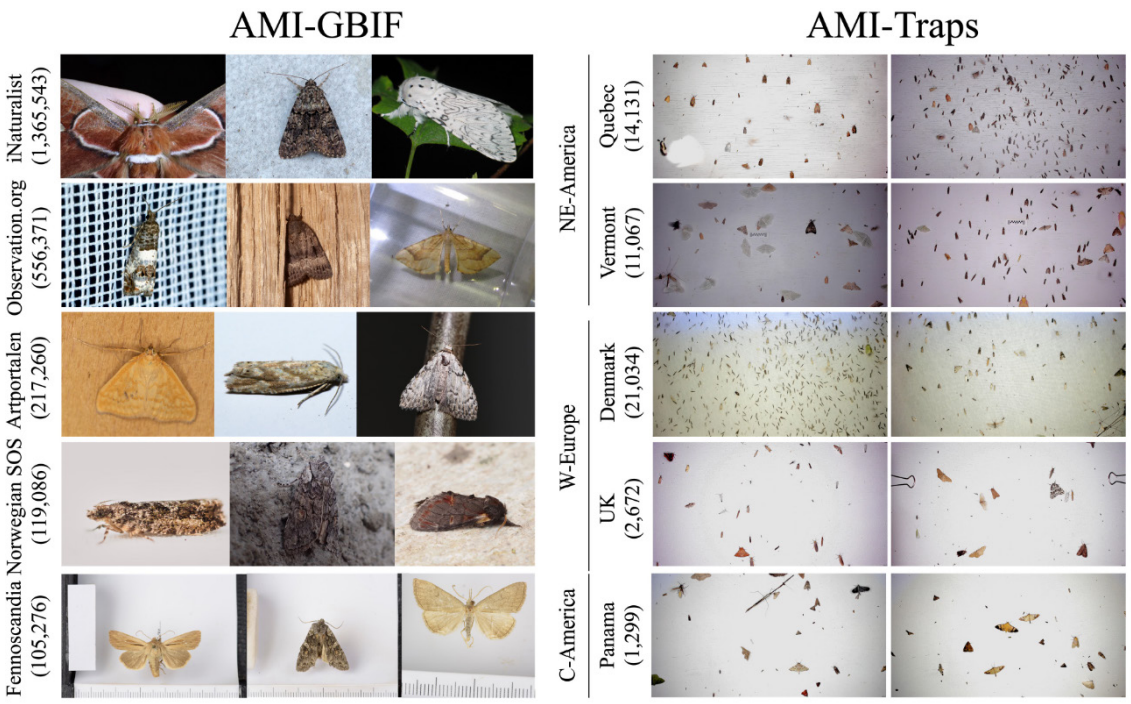


Figure 2: "Sample images from the AMI Dataset. The AMI Dataset is composed of (1) AMI-GBIF, curated from a number of sources with imagery from citizen science and museum collections, and (2) AMI-Traps, drawn from automated camera traps for insects across five countries in three regions. The number of individual insects annotated per sub-dataset is denoted in parentheses." (Jain et al., 2024, p. 7)

The trap images (see Figure 2) were manually classified by taxonomic experts in a subsequent stage. Initial bounding boxes were generated using an object detection model and were manually refined by the experts during the annotation process where necessary.

The specimens were categorized into three primary classes: moth, non-moth, or unidentifiable. Individuals identified as moths underwent further hierarchical classification within a taxonomy of species, genus, and family. Specifically, experts attempted to identify the moth at the species level; if this was not feasible, the specimen was assigned to its respective genus or, failing that, to its family.

The 2,893 raw trap images yielded a total of 52,948 (100%) labeled insect crops, consisting of 37,105 non-moths, 1,738 unidentifiable specimens, and 14,105 moths. Among the moth images, only 5,374 (10.15% of all labeled insects) could be definitively assigned to a specific species. (Jain et al., 2024, p. 7-8)

As this thesis restricts its classification to the species level, this selection process underscores the inherent difficulty of taxonomic assignment, even for experts. Simultaneously, this curation step filters out particularly ambiguous or irrelevant cases. The resulting refined dataset enables a focused investigation into the core challenges of fine-grained classification and domain generalization.

AMI-GBIF The sub-dataset containing high-resolution imagery. It comprises images of moth species known to occur in the geographic regions covered by the AMI-Traps dataset.

The species list for these regions was compiled by local entomologists and consolidated into a master checklist. This list was then reconciled for synonyms and verified for taxonomic accuracy using the GBIF Taxonomic Backbone. Utilizing the GBIF search interface, 15.1 million occurrence records were identified across various databases accessible via the GBIF platform; these records were capped at 1,000 per species.

Additional data curation steps were implemented to enhance the quality of the dataset. First, URL deduplication was performed to eliminate redundant entries, which often occurred when a single image contained multiple individuals. Furthermore, subsets with a high proportion of images unsuitable for training – such as those depicting only anatomical fragments – were excluded.

Low-resolution artifacts, including thumbnails with dimensions below 64 pixels, were also removed. Subsequently, the remaining images were resized to a minimum dimension of 450 pixels on the shorter side. To ensure the dataset consisted exclusively of adult specimens, images of eggs and larvae were filtered out. This selection was based on either the provided metadata or a custom-trained classifier designed to identify the life stage of the insects. (Jain et al., 2024, p. 8)

The resulting dataset comprises 2,564,392 moth images covering 5,364 species. While the imagery is global in origin, the selection is restricted to species occurring in the previously defined geographic regions. The class distribution is notably imbalanced: 3,099 species are represented by more than 100 images, 1,358 species have between 20 and 100, and 907 species have fewer than 20 images. (Jain et al., 2024, p. 9)

The dataset is provided with a predefined split into training (70%), validation (10%), and test sets (20%). This stratification is maintained at the species level[10]. Consequently, species with a very low number of observations might be excluded from the validation set to prioritize training and testing. (Jain et al., 2024, pp. 8–9)

A notable characteristic of the dataset is that a single observation may comprise multiple images, for example, taken from different perspectives. To maintain the integrity of the evaluation, all images associated with a specific Observation ID are restricted to a single split. This was ensured by performing the data partitioning at the observation level rather than the image level. This approach prevents data

[10]This ensures that the 70/10/20 ratio is preserved within each species, provided the number of observations is sufficient.

leakage, where the same individual insect would appear in both training and test sets.

2.2.2 Availability of the AMI Dataset

The aforementioned subsets of the AMI dataset are provided in different formats. The core dataset is available for download as a compressed archive, which contains the full AMI-Traps subset – comprising both the accompanying metadata (e.g., labels) and the actual moth trap imagery.

In contrast, the AMI-GBIF dataset is provided as a metadata-only CSV file. This file contains detailed information for each observation, including taxonomic classification (family, genus, and species) and source URLs for remote image retrieval.

Since the data originates from diverse sources – including museums and citizen science platforms aggregated via the GBIF portal – the images must be retrieved individually using the provided URLs. It is to be expected that not all download attempts will be successful due to link decay. Consequently, the utility of the AMI-GBIF dataset as a standardized benchmark is limited in the long term, as the availability of the underlying image data may decrease over time.

Question of Availability This raises the question of the extent to which the AMI dataset remains a viable benchmark, given the challenges regarding its long-term availability and accessibility.

As the complete GBIF dataset was retrieved as part of this study, it is possible to evaluate its current status. Consequently, this thesis assesses the availability of the image data as a research objective.

2.2.3 AMI Dataset Adaptation and Constraints

The AMI-Trap and AMI-GBIF datasets serve as the foundation for this research, as their specific characteristics define the challenges addressed in this study.

Due to the substantial size of the AMI-GBIF dataset (approximately 1.8 TB) and the limited computational resources available, this work exclusively focuses on the fine-grained subsets (AMI-Trap Fine-Grained and AMI-GBIF Fine-Grained). This thesis had access to around 8 TB of HDD storage, a 16-core CPU (Intel i7-13700) with 62.5 GB of RAM, and a GPU with 24 GB of VRAM.

For the experiments, partial data sets of smaller storage size were therefore extracted, which reflect the properties of the original data set in a limited form. Consequently, the performance of the classification approaches under investigation can initially be evaluated on a smaller scale in a more resource-efficient manner.

Should the methods prove effective in this reduced setting, they can subsequently be transferred to larger parts of the data set. Such a transition would then enable the dataset to fulfill its intended role as a benchmark, allowing for a direct comparison

with existing and future approaches – provided the dataset maintains its viability as a benchmark over time.

2.2.4 Square Padding to Compensate for Information Loss

A central aspect of implementing the classification methods discussed below is the generation of robust embeddings. For this purpose, the DINOv2 (Small) visual foundation model was employed. During the embedding extraction process, however, it became evident that the standard preprocessing pipeline caused information loss by discarding or cropping parts of the input imagery.

The preprocessing pipeline transforms the input images into a format that meets the requirements of the DINOv2 (Small) model. During this process, the image dimensions are adjusted: the shorter side is scaled to 518 pixels using bicubic interpolation, followed by the extraction of a 518×518 pixel center crop. The resulting square image is then encoded by the DINOv2 model into an embedding. This representation is derived from the model's latent feature space and can either be the specific as [CLS] token (global embedding)[11] or an aggregation of multiple local patch embeddings (local embeddings).

Consequently, the pipeline converts all inputs into the square format necessitated by DINOv2. Since moth recordings exhibit rectangular aspect ratios, this adaptation involves cropping peripheral areas. In practice, this may result in the loss of morphological relevant image content located near the edges.

Specifically, this cropping process may lead to the loss of critical anatomical regions. Since morphological features – including body shape, silhouette, and wing geometry – provide essential indicators for taxonomic differentiation, such information loss is likely to compromise the quality of the generated embeddings and, consequently, the overall classification performance[12].

Given these considerations, this work investigates how to mitigate the loss of relevant visual information during embedding extraction while simultaneously satisfying the requirements of the underlying DINOv2 model.

Padding Variants To address this challenge, an approach was developed to pre-expand images to a square format using various padding techniques. The strategies detailed below differ in how they utilize intrinsic image information to transform a rectangular input into a square representation. Since the addition of artificial pixels alters the original image and, by extension, the embedding generated by the model, this study investigates the resulting impact on the embeddings.

[11] The global image representation is obtained from the [CLS] token, which aggregates information from all local patch embeddings.

[12] The LEPMON project, for instance, highlights the significance of shape and symmetry features in automated moth identification. https://lepmon.de/kuenstliche-intelligenz/ (visited on 08/02/2026)

1. Mean Color

The padded areas are filled with the mean color value of the original image, creating a uniform, unstructured background.

Concept This method conveys global color information without introducing artificial structures. By maintaining a featureless background, the goal is to minimize the model's focus on padded areas during embedding extraction. However, the sharp transition at the image boundaries may still induce high contrast relative to the original image content.

2. Small Blur

The fill areas are composed of numerous small, heavily blurred image fragments, resulting in a stochastic, noise-like pattern derived from the image's own palette.

Concept This texture is designed to reduce the contrast at the image boundaries compared to Method 1. The high degree of blurring ensures that no distinct structures are recognizable.

3. Large Patches

The peripheral areas are divided into a 3×3 grid (i.e., nine subareas) and filled with randomly sampled patches from the original image.

Concept This ensures that the padding contains no structures fundamentally different from the main motif. While this approach preserves the overall image, it may duplicate relevant morphological features. This carries the risk of introducing semantic interference, potentially misleading the model during the encoding of the global embedding ([CLS] token) by suggesting non-existent spatial relationships or anatomical structures.

Research Objective Regarding the different padding strategies, the following question arises: To what extent can the three variants convert a rectangular image into a square format without the padded regions negatively impacting or biasing the extracted embeddings vectors?

Metrics for Evaluating Impact of Padding Variants To evaluate the impact of the padding methods on the generated embeddings, a segmentation approach is employed to differentiate the moth object from the background. This segmentation performance is visualized for qualitative assessment and compared against the results of an unmodified original image. Since the integrity of the main subject is central for subsequent classification, using the segmentation quality as a benchmark for embeddings stability was considered an appropriate metric.

A decline in segmentation fidelity compared to the original image would indicate that the padding introduces noise that obstructs the differentiation between the object (moth) and the background. In such a case, the object's structural integrity within the feature space would be compromised, likely leading to suboptimal classification performance. Specifically, critical morphological cues, such as the silhouette or wing geometry, might no longer be reliably encoded.

Conversely, maintaining a clear foreground-background separation would suggest that the added padding does not interfere with the generation of the embeddings. This would enable the model to holistically represent the subject, effectively leveraging the padding as a neutral context. Consequently, the moth's diagnostic characteristics would be fully preserved within the feature vectors, ensuring that the information remains accessible for subsequent classification.

The segmentation method employed here was introduced alongside the DINOv2 (Small) vision foundation model and relies on the first principal component (PCA) of the local patch embeddings (Oquab et al., 2024, p. 17). In this context, positive PCA values typically correspond to the foreground (main object), while negative values represent the background. The assessment is conducted qualitatively using a curated selection of six representative samples.

2.2.5 Resized – AMI Dataset (Squared)

Based on the evaluation of the various padding strategies, we can now address the primary research question: How do these techniques affect the classification performance in the contexts of fine-grained classification, domain generalization, and novelty detection?

For this purpose, the padding method that yielded the highest segmentation fidelity is selected to generate a second, squared (resized) version of the dataset. This version is a copy of the original AMI dataset, modified using the selected padding strategy. To evaluate the impact on the respective models, each classifier was trained and evaluated on both the original and the resized datasets, allowing for a direct comparison of the results.

2.3 Experimental Approaches

The following sections detail the classification methods – specifically K-Nearest Neighbors, Multi-Layer Perceptron, and Deep K-Nearest Neighbors – which utilize the embeddings generated by the DINOv2 (Small) visual foundation model. These approaches are presented within the context of the specific requirements of automated moth monitoring, including Fine-Grained Visual Classification, Domain Generalization, and Novelty Detection. Additionally, the metrics used to evaluate the performance of these methods are described.

2.3.1 Fine-Grained Visual Classification

This section delineates the classification methodologies of KNN and MLP, founded upon the embeddings of DINOv2. The aforementioned methods address the challenges of Fine-Grained Visual Classification.

1. KNN Classification Pipeline The first challenge regarding automated moth monitoring concerns Fine-Grained Visual Classification. To address this task, a method originally introduced by Nakata et al., 2022 is employed. This approach is particularly notable for its strong interpretability and its capacity for continual learning, while maintaining competitive classification performance.

Doerrich et al., 2024 builds upon this method by utilizing the DINOv2 (Small) visual foundation model within the specific context of medical imaging datasets. Furthermore, it discusses the approach in relation to the "Right to Erasure" (or "right to be forgotten") as mandated by the General Data Protection Regulation (GDPR) of the European Union (*General Data Protection Regulation* 2016).

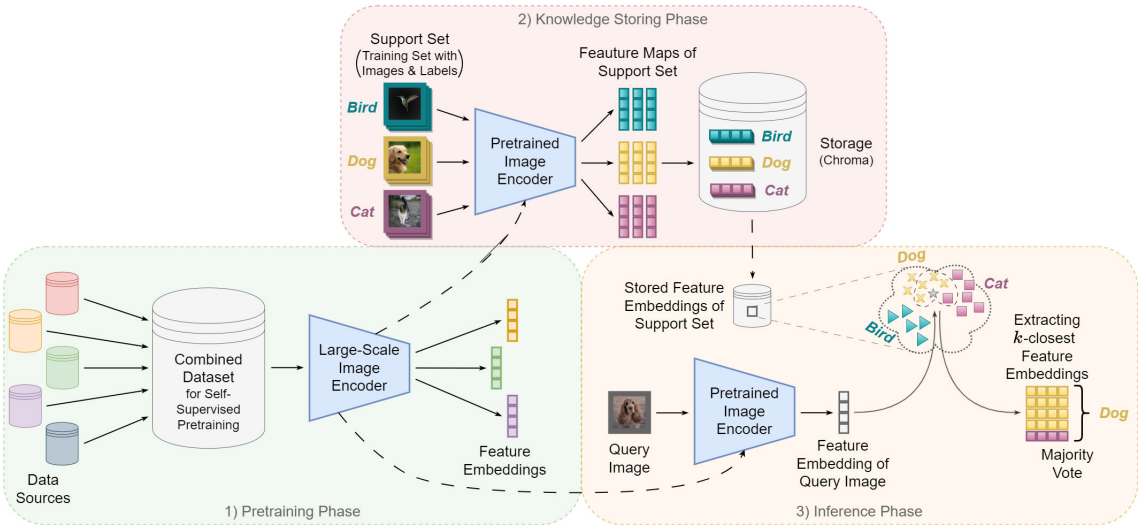


Figure 3: "During pretraining (1), the image encoder is trained to extract representative embeddings. The knowledge-storing phase (2) utilizes the pre-trained (now frozen) encoder to extract and store task-relevant knowledge from the training data. During inference (3), that knowledge allows the classification of query images through majority voting on the top-k similar embeddings." (Doerrich et al., 2024, p. 2)

The application of this method is organized into three primary phases – pre-training, knowledge-storing, and inference – as illustrated in Figure 3 and detailed below (Nakata et al. 2022, pp. 457-460; Doerrich et al. 2024, pp. 1-3).

1. **Pre-Training Phase**

   In the pre-training phase, a pre-trained Deep Neural Network (DNN) is utilised, or a new model is trained from scratch. The objective of this stage is to encode image data into representative embeddings.

   A successful model is characterized by its ability to recognize visual subtleties of objects and to map them in its latent space in such a way that similar objects are located in close geometric proximity to each other, and different objects are mapped far apart from each other. This results in robust representations (embeddings) that enable effective differentiation for subsequent classification tasks.

2. **Knowledge-Storing Phase**

   During the knowledge-storing phase, the images from the training dataset – containing both the imagery and their respective labels – are encoded by the DNN. The resulting embeddings are organized within the high-dimensional latent space to create a knowledge storage, which serves as the foundation for the KNN classification.

   Once this knowledge storage is established, new and previously unseen data can be classified. This is achieved by computing the position of their embedding within the feature space, comparing them against the stored entries in the training set, and identifying the nearest neighbors.

3. **Inference Phase**

   During the inference phase, a query image is classified. First, the image is encoded by the DNN and projected as a embedding into the latent space. Subsequently, the K-Nearest Neighbors are identified in the knowledge store based on a suitable distance metric. The final classification is determined through majority voting among the labels of these k neighbors.

   In practice, cosine similarity is frequently employed as the distance metric. It measures the angular relationship between feature vectors, making it robust against variations in vector magnitude (scaling differences). (Doerrich et al., 2024, p. 2)

Limitations of Conventional Deep Neural Networks This method builds upon conventional Deep Neural Networks (DNNs), which have achieved remarkable success across various image classification domains. DNNs excel at recognizing complex patterns by leveraging Multi-Layer Perceptron architectures, massive datasets, increasing computational power, and refined training paradigms. However, their internal decision-making processes often lack transparency, leading them to be characterized as “black-box” models (Nakata et al. 2022, p. 457; Doerrich et al. 2024, p. 1). Furthermore, adapting model parameters to incorporate new data remains a

significant challenge, as it carries the risk of overwriting previously acquired knowledge. This phenomenon, known as catastrophic forgetting, can result in substantial performance degradation.

The approach adopted here, following Nakata et al., 2022 and Doerrich et al., 2024, extends the conventional DNN framework and addresses several of the aforementioned limitations.

Advantages within the Context of Moth Classification By utilizing a KNN classifier, the rationale behind each classification becomes transparent, as it is directly derived from the K-Nearest Neighbors within the knowledge store. This allows users to trace and validate the specific evidence leading to a taxonomic assignment. Consequently, the method is well-suited not only for fully automated classification but also for the development of Decision Support Systems (DSS) that assist human experts.

The knowledge storage itself is generated using a DNN but remains modular, allowing for retrospective expansion and correction. Individual samples can be added or removed at any time, enabling continual learning without the need for model retraining. This approach obviates both the high computational costs of retraining and the risk of catastrophic forgetting. Consequently, the knowledge store can be dynamically adapted to include new species or corrected to remove erroneous data, ensuring a high level of data quality – all without the inherent risks associated with traditional DNN updates (Nakata et al. 2022, pp. 457-460; Doerrich et al. 2024, p. 4).

Furthermore, this method facilitates compliance with data protection regulations, such as the European "Right to Erasure", without significantly affecting performance. This is achieved by decoupling the training data used for the pre-trained DNN from the specific instances stored in the KNN knowledge store. Thus, individual records can be deleted from the store with minimal impact on the overall classification capability (Doerrich et al., 2024, p. 4).

Collectively, these attributes establish a flexible framework for moth classification, facilitating adaptation to real-world monitoring requirements.

2. MLP Classification Head An alternative classifier that typically yields higher Accuracy scores than KNN models while leveraging the same embeddings is the Multi-Layer Perceptron[13]. Within this study, the MLP is implemented as a comparative model to evaluate the performance trade-offs against the KNN approach based on the generated embeddings. The application of the MLP differs from the KNN in the following ways:

1. Pre-Training Phase

This stage remains identical to the KNN classification pipeline.

[13]Notably, even simple linear classifiers based on DINOv2 embeddings have been shown to outperform KNN classifiers by approximately 3 percentage points. (Oquab et al., 2024, pp. 11-12)

2. Training Phase (Weight Adaptation)
Unlike the KNN approach, the generated embeddings are not stored in a knowledge storage. Instead, they serve as training data to optimize the weights of an MLP. Through this process, the MLP learns to map the high-dimensional embeddings to specific class labels.

3. Inference Phase
During inference, new images are encoded into embeddings, which are then processed by the trained MLP classifier to predict the most likely class.

Advantages and Disadvantages An advantage of this approach compared to DNN lies in its computational efficiency. updates or adjustments only require retraining the MLP classifier, which possesses significantly fewer parameters than the underlying DNN. Furthermore, it remains possible to decouple the datasets used for generating embeddings from those used for training the MLP head.

At the same time, the inherent limitations of DNNs persist. Specifically, the internal decision-making processes of the MLP head lack interpretability, and correcting individual misclassifications is not possible without retraining the head – unlike with KNN, where the knowledge storage remains directly editable. Despite these limitations, the MLP's typically superior classification performance warrants its evaluation as a comparative baseline against the KNN method within the specific context of moth classification.

DINOv2 Foundation Model The DINOv2 (Small) visual foundation model serves as the backbone for both classification methods (KNN and MLP). Utilized as a frozen feature extractor without further fine-tuning, it generates high-dimensional embeddings (feature vectors) upon which the classifiers are built. Consequently, the pre-trained model must be capable of recognizing subtle morphological differences between moth species to map them effectively into the latent space.

The objective is to encode relevant species-specific features such that species instances cluster together, while dissimilar species remain well-separated within the latent space. This organization is essential for successful classification using both the KNN and MLP approaches. Ultimately, the performance of the entire pipeline depends fundamentally on the quality of the embeddings generated by the DINOv2 (DNN).

Nakata et al., 2022 utilize various encoders in their implementation. Among these, the pre-trained CLIP[14] (Contrastive Language–Image Pre–training) model demonstrated the highest Accuracy scores across several datasets, including ImageNet-1k,

[14]"Vision Transformer [...] Base model with input patch sizes of 16×16 (ViT-B/16) trained by CLIP on 400 million image and text pairs collected from the internet" ((Nakata et al., 2022, p. 462))

CIFAR-10/100, and STL-10. This success is attributed to its robust generalization capabilities (e.g., 74.0% on ImageNet-1k, 94.4% on CIFAR-10, 74.3% on CIFAR-100, and 98.9% on STL-10).

In contrast, Doerrich et al., 2024 employe the DINOv2 visual foundation model as the encoder. While CLIP was trained on multimodal data (here: image-text pairs), DINOv2 relies exclusively on visual data. It was trained on approximately 142 million images using a self-supervised learning paradigm. The underlying LVD-142M dataset consists of curated public datasets supplemented with similar images retrieved from the web. (Oquab et al., 2024, pp. 11–14; Dosovitskiy et al., 2020)

In benchmark comparisons, DINOv2[15] outperforms competing models such as CLIP in both linear probing and KNN scenarios on the ImageNet-1k dataset. DINOv2 also achieves higher Accuracy scores on fine-grained datasets like iNaturalist 2018 (iNat2018) and iNaturalist 2021 (iNat2021). These results make DINOv2 particularly suitable for moth classification, where recognizing subtle visual differences is critical. (Oquab et al., 2024)

It should be noted that the performance scores mentioned for both DINOv2 and CLIP were achieved using their largest models. For DINOv2, this refers to DINOv2-Giant (ViT-G/14, 1,136.5 million parameters at 518×518 resolution); for CLIP, it refers to ViT-L/14@336px (428 million parameters at 336×336 resolution).

However, this study employs the distilled DINOv2-Small variant (ViT-S/14, 22.1 million parameters). While it exhibits lower classification performance than DINOv2-Giant, it allows for execution with significantly lower hardware requirements. In fine-grained benchmarks, the performance gap between the Small and Giant variants is approximately 12.6 percentage points on iNat2018 and 11.5 percentage points on iNat2021. (Oquab et al., 2024)

The employment of the smaller model constitutes a rational compromise between computational cost and predictive Accuracy. The method is initially assessed for its effectiveness in the context of moth classification, before transitioning to larger, more resource-intensive models.

Research Objective Finally, the challenge of fine-grained moth classification will be addressed by evaluating the two proposed methods – KNN and MLP – utilizing the DINOv2 (Small) embeddings. The objective is to compare and evaluate the performance of these classification approaches specifically within the domain of fine-grained moth classification.

Evaluation Metrics To evaluate the performance of the classification methods, several standard metrics are employed. The foundation for these calculations is the confusion matrix, which is generated by comparing the predicted class labels against the ground truth of the test dataset. In a multi-class context, four categories are distinguished for each individual class (following a one-vs-rest approach):

[15] using an out-of-the-box frozen backbone

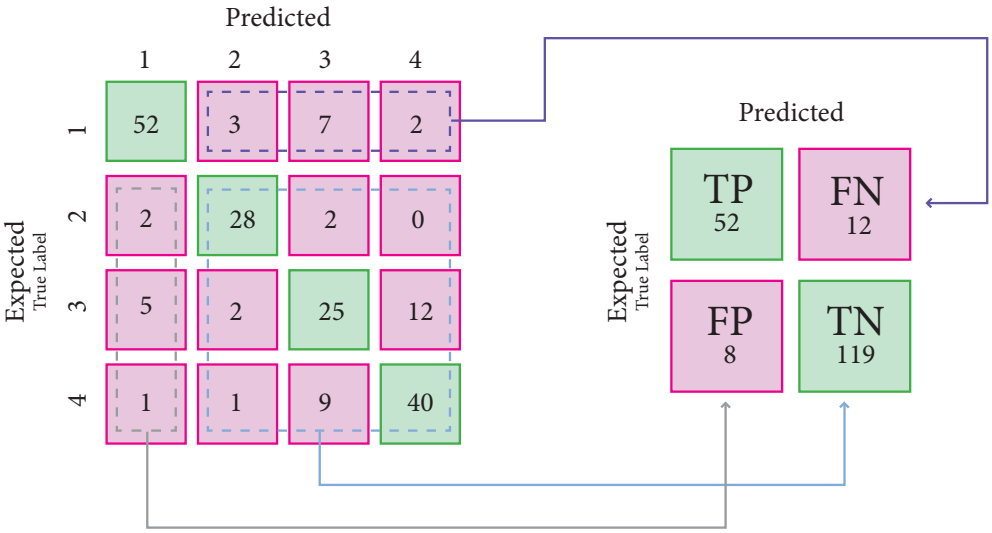


Figure 4: The illustration shows the derivation of TP, FP, FN, and TN values within a multi-class context. It specifically shows how the values from the multi-class matrix are consolidated into a binary format to evaluate a single class. (Adapted from Kundu, 2026)

Based on this matrix, various evaluation metrics can be calculated for each individual class. Common metrics include Precision, Recall, and the F1-Score. To provide an overall performance measure across all classes, these metrics can be aggregated using micro- or macro-averaging. In this study, macro-averaging is employed to evaluate classification performance. The primary advantage of this approach is that it assigns equal weight to each class, ensuring that performance for each species is reflected regardless of their sample size.

The Accuracy metric represents the proportion of correctly classified instances relative to the total number of samples in the dataset (Grandini, Bagli, and Visani, 2020, p. 3):

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

The Precision metric represents the ratio of True Positives (TP) to all instances assigned to a specific class (TP + FP). This score must be interpreted in the context of the incidence[16] rate, as a higher frequency of a class in the dataset increases the baseline probability of correctly identifying True Positives by chance (Grandini, Bagli, and Visani, 2020, p. 2):

$$\text{Precision} = \frac{TP}{TP + FP}$$

[16]The incidence rate indicates the number of samples belonging to a specific class within the test dataset.

Sensitivity, also referred to as Recall or the True Positive Rate (TPR), describes the proportion of correctly classified instances of a class relative to all actual instances of that class present in the dataset (Grandini, Bagli, and Visani, 2020, p. 3):

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

The F1-Score describes a weighted average between Precision and Sensitivity, ranging from 0 to 1. In the case of unbalanced values between Precision and Recall, it weights the smaller values more heavily than balanced weights. For example, if both Recall and Precision are 0.80, the F1-Score is also 0.80. However, if there is a disparity – such as a Recall of 0.60 and a Precision of 1.0 – the F1-Score drops to 0.75 (Grandini, Bagli, and Visani, 2020, p. 5):

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Sensitivity}}{\text{Precision} + \text{Sensitivity}}$$

Accuracy, Precision, and Sensitivity can also be calculated globally across all samples. These aggregated metrics are referred to as Micro Accuracy, Micro Precision, and Micro Sensitivity. In this approach, classes (K) with many samples have a greater influence on the score than classes with a small number of samples. Consequently, class imbalance within the test dataset can significantly affect these metrics, as the results primarily reflect the performance on the most frequent species:

$$\text{Micro Precision} = \frac{\sum_{k=1}^{K} TP_k}{\sum_{k=1}^{K} (TP_k + FP_k)}$$

Macro-averaging serves as an alternative to micro-metrics. This approach accounts the average classification performance of all classes equally for metrics such as Accuracy, Precision, and Sensitivity. Consequently, each class is weighted equally in the final score, irrespective of class imbalance or the varying number of samples per category within the test dataset. These metrics are formally referred to as Macro Accuracy, Macro Precision, and Macro Sensitivity. The latter is also referred to as Balance Accuracy. The following formula illustrates the calculation using Balance Accuracy as an example (Jain et al. 2024, p. 11; Grandini, Bagli, and Visani 2020, p. 7):

$$\text{Sensitivity}_k = \frac{TP_k}{TP_k + FP_k}$$

$$\text{Balanced Accuracy} = \frac{\sum_{k=1}^{K} \text{Sensitivity}_k}{K}$$

2.3.2 Domain Generalization & Fine-Grained Classification

The second challenge addresses domain generalization. In this context, domains refer to different image quality of moth captures. Ideally, these different representations should not affect the semantic classification of the species.

Specifically, this work defines domains through the discrepancy between high-resolution training data and low-resolution test data. Consequently, the domain of the low-resolution test samples is **not** represented within the training dataset, requiring the model to generalize across the resolution gap.

To address this challenge, the same KNN and MLP classification frameworks previously detailed in the context of fine-grained moth classification are employed. Once again, the foundation for effective classification is established by the DINOv2 (Small) vision foundation model, which encodes the image data into high-dimensional embeddings.

In bridging the domain gap between high-resolution training data and low-resolution test data, the central objective remains the same: the model must map image data into a latent space where intra-species samples cluster closely together, while inter-species differences are clearly demarcated, regardless of image resolution. Achieving such a robust representation is a prerequisite for maintaining classification performance across varying image conditions.

The ability of DINOv2-Giant to successfully generalize across different domains using its embeddings was demonstrated by Oquab et al., 2024 across various benchmark datasets[17]. The model's classification performance reached Top-1 accuracies of 75.9% on ImageNet-A (Im-A), 78.8% on ImageNet-R (Im-R), and 62.5% on Sketch. For ImageNet-C (Im-C), the error rate was evaluated at 28.2%, where a lower value indicates better performance.

As previously mentioned in the section on fine-grained classification, this study utilizes the distilled DINOv2-Small variant. However, this smaller model exhibits a significantly lower generalization performance compared to DINOv2-Giant. Specifically, the Top-1 Accuracy scores for DINOv2-Small are 42.4, 25.1, and 21.3 percentage points lower for the Im-A, Im-R, and Sketch benchmarks. Regarding the ImageNet-C (Im-C) dataset – where lower scores indicate superior performance – DINOv2-Small's error rate is 26.2 percentage points higher than that of the "Giant" version. (Oquab et al., 2024, pp. 12-13)

Research Objective This leads to the question of this study: to what extent can the KNN and MLP classification methods – utilising the DINOv2 (Small) embeddings – successfully generalise across the domain gap between high-resolution images (training data) and low-resolution images (inferences data) typically captured by moth traps?

Evaluation Metrics To evaluate the domain generalization performance of the proposed methods, the same metrics previously described for fine-grained visual classification are employed. For a detailed description of these metrics, refer to Section 2.3.1.

[17]To evaluate this, a linear classifier was trained on ImageNet-1k embeddings and subsequently tested on datasets specifically designed to challenge covariate shift (outperforming competing models in Im-a and Im-C such as OpenCLIP).

2.3.3 Novelty Detection

The third challenge of this work involves the detection of novel, previously unseen species. These are species that were not part of the training set and, therefore, cannot be correctly categorized by the fine-grained classifier.

The objective is to perform a binary classification to distinguish between known, In-Distribution samples and unknown or novel, Out-of-Distribution samples.

DKNN Classification Pipeline For this purpose, a method introduced as Out-of-Distribution Detection with Deep Nearest Neighbors (DKNN) by Sun et al., 2022 is employed. This distance- or density-based binary classifier builds directly upon the previously described KNN (fine-grained) classification approach.

The DKNN approach operates on the same feature vectors (DINOv2 embeddings) as the KNN classifier, extending it with a threshold-based decision mechanism. By analyzing the distances or respectively the local density of the k nearest neighbors in the knowledge storage, the system determines whether a given input sample belongs to the known ID or should be categorized as OOD.

Analogous to the fine-grained classification approach, the DKNN method is structured into three primary phases: Pre-Training, Knowledge Storing, and Inference. All three stages follow the same principles with additional steps for the inference-phase. The following description focuses exclusively on these modifications. For a fundamental description of the three phases, please refer to the KNN Classification Pipeline in Section 2.3.1.

1. Pre-Training Phase

This phase is identical to the KNN pre-training phase described in Section 2.3.1.

2. Knowledge Storage and Normalization

A crucial factor for the robust performance of the DKNN approach in the experiments conducted by Sun et al., 2022 was a compact and well-structured representation of the data within the latent space. To achieve this, embeddings were normalized, leading to a significant improvement in the FPR95 metric. Since Sun et al., 2022 employed Euclidean distance as a distance metric – where the magnitude of a vector directly influences the distance value – the performance gains resulting from normalization are highly plausible.

In contrast, this study utilizes cosine distance (or equivalently, cosine similarity). This is an angular metric that does not account for the magnitude of the vectors, focusing instead on their angular relationship within the latent space. Nevertheless, the embeddings in this work were also normalized prior to classification to ensure

that all vectors are scaled to a unit norm. Since the DKNN framework is built directly upon the KNN classifier, this normalization was applied consistently across both methods KNN and DKNN.

3. Inference Phase and Thresholding

During the inference phase, a query sample is classified as either ID or OOD. In contrast to standard KNN classification, an additional thresholding mechanism is introduced. Specifically, the distance to the k-th nearest neighbor is compared against a predefined threshold value.

If the distance to the k-th neighbor remains below this threshold, the sample is considered to be part of the known distribution (ID). Conversely, if the angular distance exceeds the threshold, the instance is flagged as Out-of-Distribution. Thus, the threshold defines a required local density of neighbors in the knowledge storage; only samples that meet this density criterion are recognized as known species.

Three different Thrashold Calculations This thesis compares three distinct strategies for implementing threshold calculations and, consequently, three variants of the DKNN classifier. The objective of this analysis is to evaluate these variants regarding their classification performance. The following provide a brief overview of the different approaches employed in this study:

1. The first variant follows the standard procedure described in the inference phase. It calculates the distance to the k-th nearest neighbor (regardless of class) and compares it against a global, predefined threshold. The threshold is calculated on the basis of all classes.

2. The second variant differentiates the threshold used on a pre-determined class assignment. First, a fine-grained classification is performed on the sample. Based on this prediction, a class-specific threshold is selected. Subsequently, the distance to the k-th neighbor is determined and compared against this class-specific threshold. These thresholds are therefore calculated separately for each class.

3. The third variant utilizes the same selection logic as the second approach but differs in the underlying methodology for calculating the class-specific thresholds. The calculation of the various thresholds is described in detail in the Section 4.3.1.

Advantages Compared to many other OOD detection methods, the DKNN approach does not rely on prior assumptions regarding the underlying distribution of ID data. Consequently, it is classified as a non-parametric method, offering greater flexibility across diverse applications.

In their experiments, Sun et al., 2022 demonstrate that DKNN achieves higher OOD detection rates than common parametric methods – particularly on large-scale datasets. This superior performance is attributed to the high number of classes and the structural complexity of embeddings[18]. In such scenarios, traditional distributional assumptions are often violated and no longer hold true.

Using ImageNet-1k as a training set and testing against OOD benchmarks like iNaturalist, SUN, Places, and Textures, Sun et al., 2022 showed that the non-parametric nature of DKNN is especially effective for large-scale data. This suggests that for the real-world application of automated moth classification, which involves high class counts and extensive datasets, the DKNN method is likely to outperform traditional parametric approaches.

Another advantageous of this method is that it does not require prior knowledge of OOD data. The thresholds used to distinguish between ID and OOD samples are derived exclusively from the known ID data. This makes the approach particularly suitable for the described moth monitoring scenario, where novel species may emerge unexpectedly due to climate change or global trade.

Sun et al., 2022 further demonstrate that the method is highly flexible and compatible with various model architectures, such as CNNs and ViTs[19]. Consequently, it can be seamlessly integrated into existing systems.

In this study, the method is combined with the previously established fine-grained classifier. Due to the algorithmic similarity between KNN and DKNN, the OOD detection can leverage existing computational results, such as pre-calculated distance matrices.

Research Objective This leads to the question of how effectively the proposed DKNN approach – leveraging the DINOv2 (Small) embeddings – can discriminate between known and unknown species within the context of moth classification. This research objective is addressed through a comparative evaluation of the DKNN method across the three distinct implementation threshold strategies previously outlined.

Evaluation Metrics To assess the performance of the three DKNN variants, performance curves and derived metrics are described. In the literature, ROC curve (Receiver Operating Characteristic curve), metrics such as AUROC (Area Under the Receiver Operating Characteristic), AUPR (Area Under the Precision-Recall Curve), and the False Positive Rate at a True Positive Rate of 95% (FPR95) are widely established. Additionally, the F1-Score can be utilized for performance evaluation. This study does not employ all of the aforementioned metrics. Instead, the analysis is restricted to the ROC curve with its associated AUROC value and the

[18]This refers to high-dimensional embeddings mapped into complex latent spaces.

[19]The method remains applicable regardless of the training objective, including cross-entropy or contrastive loss.

FPR95 metric. (Yang et al. 2024, pp. 5639–5640; Chen and Rolnick 2023, p. 3; Sun et al. 2022)

The novelty detector (i.e., OOD classifier) is a binary classifier designed to distinguish between ID samples (known instances) and OOD samples (novel instances). For evaluation purposes, it is essential to define the positive and negative classes: in this work, OOD samples are designated as the positive class (True), while known ID samples represent the negative class (False).

The foundation for generating performance curves and calculating scores is the binary confusion matrix, which is constructed by comparing the classifier's predictions with the ground truth. Consequently, ID samples may result in False Positives or True Negatives, whereas OOD samples are categorized as either True Positives or False Negatives.

In OOD detection, the selection of the decision threshold can significantly impact the classifier's performance. For instance, a threshold with great length (corresponding to low local density) may enable high ID detection, but simultaneously lead to poor OOD detection. This inherent trade-off must be addressed during the model evaluation. Consequently, classification performance is calculated across a sequence of 100 distinct threshold values to provide a comprehensive analysis.

The ROC curve visualizes the trade-off between ID and OOD classification performance across all calculated threshold values. To generate the curve, the True Positive Rate[20] of the OOD data and the False Positive Rate[21] of the ID data are calculated for each individual threshold.

The AUROC quantifies the area beneath this curve, providing a single scalar score for the overall model performance. A advantage of the ROC curve and the AUROC metric is that they are independent of the incidence rate[22]. Furthermore, the AUROC considers both the TPR for OOD data and the FPR for ID data with equal weight.

Another key metric for evaluating an OOD classifier is the False Positive Rate at a True Positive Rate of 95% (FPR95). This metric identifies the specific point on the ROC curve where the model correctly detects 95% of the OOD samples. Specifically, the FPR95 answers the following question: What percentage of ID samples are erroneously classified as OOD when the classifier is tuned to recognize 95% of the actual OOD samples?

2.3.4 Domain Generalization & Novelty Detection

The fourth challenge revisits domain generalization, as previously introduced in Section 2.3.2. In this context, however, it is examined specifically through the lens of novelty detection. This setup mirrors real-world conditions where an automated

[20] Also referred to as Sensitivity or Recall

[21] Equivalent to 1-specificity

[22] Incidence rate refers to the proportion of positive = OOD samples within the test dataset.

moth monitoring system must reliably identify novel species despite degraded image quality (domains).

To address this challenge, the same DKNN classification framework described in the section on novelty detection is employed. Once again, the DINOv2 (Small) visual foundation model provides the essential basis for successful classification by encoding the image data into high-dimensional embeddings. Specifically, the model must ensure that images of the same species cluster closely together, while distinct species remain clearly demarcated, regardless of image resolution. Such robust representation is a prerequisite for maintaining classification performance across varying image quality.

Research Objective Consequently, this raises the question of how effectively the proposed DKNN approach (2.3.3) – utilizing the DINOv2 (Small) embeddings – can discriminate between known and unknown species while simultaneously generalizing across the domain gap between high-resolution training data and low-resolution test data.

Evaluation Metrics To assess the performance the same evaluation metrics previously established for novelty detection are employed.

3 Dataset and Embeddings

Organized into five sections, this chapter details the preparation and adaptation of the AMI dataset and resized dataset for subsequent classification, as well as the generation of the corresponding embeddings.

The implementation and source code associated with this thesis are available in the following GitHub repository: https://github.com/Johannesproximo/masterThesis/tree/main.

3.1 AMI Dataset Acquisition

The AMI dataset, published by Jain et al., 2024, encompasses several sub-datasets, specifically the high-resolution AMI-GBIF and the low-resolution AMI-Trap datasets. To utilize the data, the initial step requires downloading a ZIP archive from Zenodo[23], which is approximately 16 GB in size.

3.1.1 Download of the AMI-GBIF Image Data

This archive contains the metadata for the AMI-GBIF datasets, covering both the AMI-GBIF Fine-Grained and the AMI-GBIF Binary subsets. It is important to note that the corresponding image files are not included in the ZIP archive and must be acquired through a separate download process.

For this purpose, the datasets are provided in predefined splits (training, validation, and test) as CSV files (see Figure 5). In addition to several region-specific splits (North America, Western Europe, and Central America), a combined split encompassing "all regions" is available for the fine-grained dataset. This study exclusively utilizes the combined split, which aggregates the data from all geographical regions.

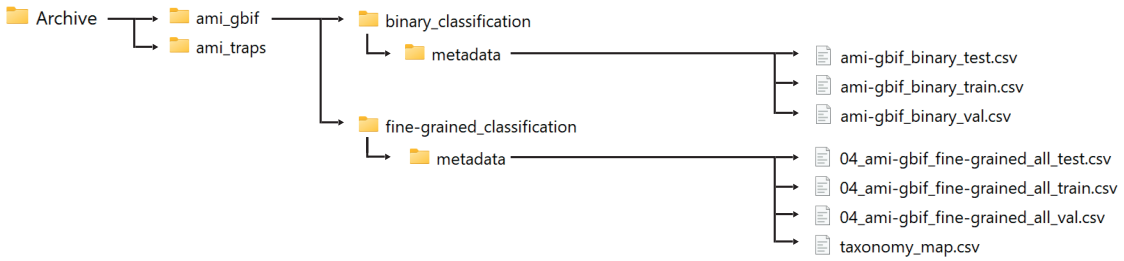


Figure 5: The folder structure shows how the available AMI data set archive is organized and where the GBIF data sets (CSV files on the right hand side) used for the work are located.

Image acquisition is performed using the predefined CSV files, which contain comprehensive metadata for each observation along with the respective image URLs.

[23]https://zenodo.org/records/12554005 (visited on 08/02/2026)

For every observation in the AMI-GBIF dataset, the corresponding image had to be downloaded via the provided link. Notably, there is an overlap of approximately 350,000 observations between the Binary and Fine-Grained subsets, accounting for roughly half of the entries in the Binary dataset.

Implementation To facilitate the download process, Python-based Jupyter Notebooks[24] were developed for both subsets (Fine-Grained and Binary).The metadata CSV files are loaded as DataFrames, which are then augmented with two additional columns: "image_download" and "image_download_fail_reason". The former uses a Boolean value to indicate success, while the latter documents the specific cause in case of failure.

To accelerate the acquisition, the download process was parallelized using Python's concurrent package. Furthermore, to avoid redundant downloads – such as the 350,000 identical entries mentioned above – the script verifies whether the image is already available in a local cache directory and copies it from there if necessary.

The entire download process was executed multiple times to mitigate potential network instabilities, with only previously failed attempts being retried.

For the domain monarch.calacademy.org, SSL verification was disabled to facilitate successful downloads. This measure allowed for the acquisition of approximately 30 additional images within the GBIF Binary dataset.

Despite successful retrieval, some files could not be opened due to file corruption. To identify these instances, all downloaded images are validated[25] using the *Image.verify()* function from the Python library Pillow (PIL) (Clark, 2026). A total of 156 images failed this verification; consequently, their "image_download" status is updated to False, and a corresponding entry is made in the "image_download_fail_reason" field.

3.1.2 Extraction of the AMI Trap Data

The AMI Trap dataset similarly comprises two sub-datasets: AMI Trap Fine-Grained and AMI Trap Binary. In contrast to the AMI-GBIF datasets, the associated image data is provided within the ZIP archive as WebDatasets, distributed across multiple TAR files (see Figure 6). While WebDatasets can be utilized directly in machine learning frameworks, this study involves extracting their contents for subsequent processing, such as generating custom datasets.

[24] See Jupyter Notebooks:
/jupyter_notebooks/00_download_image_high/fine_grain/0_download_images.ipynb
/jupyter_notebooks/00_download_image_high/binary/0_download_images.ipynb

[25] See Jupyter Notebooks:
/jupyter_notebooks/00_download_image_high/fine_grain/1_download_images_validate.ipynb
/jupyter_notebooks/00_download_image_high/binary/1_download_images_validate.ipynb

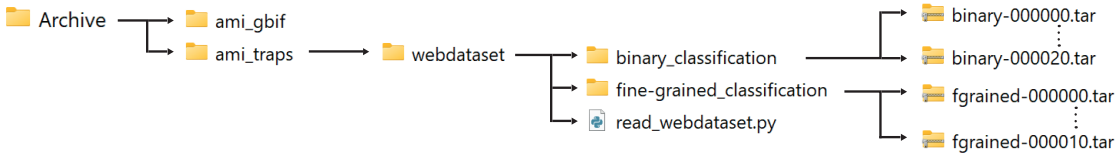


Figure 6: The folder structure shows how the available AMI data set archive is organized and where the Trap data sets (TAR files) used for this work are located.

Implementation For this study, both datasets (AMI Trap Fine-Grained and AMI Trap Binary) were extracted from the WebDatasets, providing direct access to the captured trap images and their associated metadata[26].

To ensure the integrity of the extracted files, a validation process[27]. Is once again conducted using the *Image.verify()* function from the Pillow (PIL) library, following a procedure similar to the one used for the AMI-GBIF data acquisition.

3.2 Dataset Splitting and Preprocessing

The datasets utilized in this study are based on the AMI-GBIF Fine-Grained and the AMI-Trap Fine-Grained datasets. Specific subsets were extracted from both to address the research challenges defined in this work. For the novelty detection experiments, the test sets (GBIF and Trap) were partitioned into In-Distribution and Out-of-Distribution subsets. An initial overview of the resulting datasets is provided in Figure 7.

Furthermore, several constraints were applied, such as restricting the data to the taxonomic level of "species", which resulted in a reduction of the overall AMI dataset size. This filtering process was specifically designed to limit computational requirements and minimize the resource consumption of this study.

General Constraints – Impact on Dataset Volume To manage the data volume, it was decided to restrict the AMI dataset exclusively to the species taxonomic level. This reduction primarily affects the Fine-Grained Trap (test) dataset, as not all instances in this set are annotated down to the species level. As a result of this filtering, the Trap dataset was significantly reduced from 14,105 to 5,371 samples.

Furthermore, all instances with failed image downloads were removed from the Fine-Grained GBIF datasets (Train, Val, Test). Consequently, the GBIF dataset was reduced by 1.69% (43,480 samples), decreasing from 2,564,392 to 2,520,912 samples.

[26]See Jupyter Notebooks:
/jupyter_notebooks/01_extract_image_low/0_extract_image_low(trap).ipynb
[27]See Jupyter Notebook:
/jupyter_notebooks/01_extract_image_low/1_extracted_images_check.ipynb

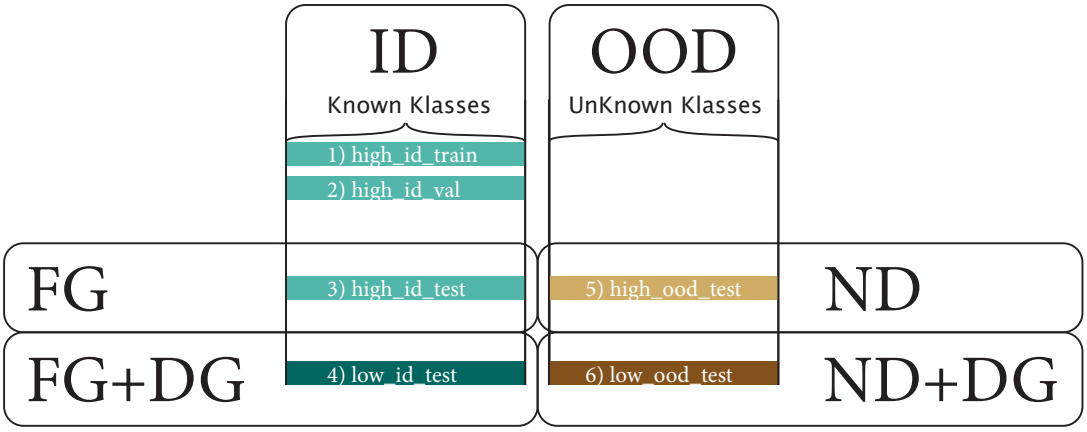


Figure 7: Overview of the datasets created for this study and their characteristics: ID indicates classes known to the classifier, while OOD refers to unknown classes. Additionally, the specific challenges for which each dataset is utilized are indicated: FG for fine-grained classification, DG for domain generalization, and ND for novelty detection.

ID and OOD Dataset Partitioning Since the AMI dataset lacks a predefined split for distinguishing between In-Distribution and Out-of-Distribution, a custom partition was applied for this study. This split affects all utilized subsets, including the GBIF Fine-Grained datasets training, validation, test and the Trap Fine-Grained test set.

The primary objective was to generate a split that maximizes the number of classes while simultaneously minimizing class imbalance.

To ensure consistency across domains, the partitioning was anchored to the smallest subset: The Trap Fine-Grained test set. While the GBIF dataset encompasses 5,364 classes, the Trap test set contains only 487. By using the Trap dataset as the baseline for class selection, the highest possible class overlap is maintained when applying the split to the larger GBIF datasets.

Since the Fine-Grained Trap (test) dataset exhibits significant class imbalance – with sample counts ranging from over 500 to just a single instance – large classes were capped at 100 samples, and classes with fewer than six samples were excluded. Following this adjustment, 140 of the original 487 classes were retained and subsequently partitioned into 100 ID and 40 OOD classes.

This division served as the basis for the dataset split, which was subsequently applied to the GBIF Fine-Grained datasets. Specifically, the training and validation sets were restricted to the ID classes, while the test set was partitioned and pruned into ID-Test and OOD-Test classes.

Furthermore, the sample counts per class for the resulting GBIF datasets were constrained: the training set was capped at 1,000 samples with a minimum requirement of 500 samples per class. The validation set was limited to a range of 50 to 100 samples, and the test sets (both ID and OOD) were restricted to between 100 and 200 samples per class.

To this end, classes exceeding the maximum sample threshold were downsampled by randomly[28] selecting a subset of examples. Conversely, classes that fell below the minimum sample requirement were excluded entirely.

As a result of these constraints, the ID datasets were reduced to 91 classes, and the OOD dataset to 37 classes. The following Table 2 provides an overview of the final datasets generated during this partitioning process.

Dataset Type	Resolution	Filename	Classes	min	max	total
Training	high (GBIF)	high_id_train.csv	91	529	1000	85,803
Validation	high (GBIF)	high_id_val.csv	91	78	100	8,949
Test FG	high (GBIF)	high_id_test.csv	91	138	200	17,832
Test FG + DG	low (Trap)	low_id_test.csv	91	6	100	2,138
Test ND	high (GBIF)	high_ood_test.csv	37	122	200	7,213
Test ND + DG	low (Trap)	low_ood_test.csv	37	6	100	698

The generated datasets listed in the table can be found under the path: /data/datasets/created/.

Table 2: Summary of the generated datasets: The table details the dataset type, image resolution (high/GBIF vs. low/Trap), number of classes, and sample distribution (minimum, maximum, and total counts) for the training, validation, and testing datasets.

Implementation The generation of the six datasets leveraged the CSV metadata files from the AMI-GBIF download and those generated during the AMI-Trap extraction process. This implementation was carried out using a Python-based Jupyter Notebook[29], in which the previously described constraints were applied through indexing and filtering operations to produce the final subsets. In addition to these datasets, ID and OOD label maps were generated[30]

Finally, the images corresponding to the newly created datasets were isolated from the initial AMI-GBIF and AMI-Trap acquisition and transferred to a separate storage location[31].

[28] Random selection was based on a fixed seed to ensure reproducibility.

[29] See Jupyter Notebook:
/jupyter_notebooks/02_create_datasets/1_create_datasets.ipynb

[30] See Jupyter Notebook:
/jupyter_notebooks/02_create_datasets/2_create_label_map.ipynb

[31] See Jupyter Notebook:
/jupyter_notebooks/03_copy_images/02_copy_images.ipynb

3.3 Image Padding Strategies

To mitigate information loss during the generation of embeddings with DINOv2, three distinct padding methods were implemented. The primary objective was to determine how effectively these variants could transform a rectangular input into a square format without the additional padded areas introducing significant bias into the resulting embeddings.

3.3.1 Implementation Steps

1. **Image Selection:** Description of the image samples used for the experiments.
2. **Padding Strategies:** Description and implementation of the three selected padding techniques.
3. **Cropping Visualization:** Visualization of the impact of the preprocessing pipeline's cropping on the original and square images.
4. **Segmentation Visualization:** Visualization of the segmentation based on the local (patch) embeddings of the original and padded square images.
5. **Joint Visualization:** A comprehensive overview of all stages, including padding, preprocessing cropping, and the final embedding-based segmentation.

The individual steps were implemented in a Jupyter Notebook[32] using Python. The implementation utilizes several key libraries, including timm for loading the DINOv2 model, scikit-learn (Pedregosa et al., 2011) for Principal Component Analysis (PCA), and Pillow (PIL) (Clark, 2026) for image processing.

Image Selection Initially, six rectangular images were selected for the experiment: three from the high-resolution GBIF dataset and three from the low-resolution AMI-Trap dataset. The goal was to cover a variety of image types. For the GBIF dataset, the selection included one image with a uniform background, one with a natural ecological setting, and one featuring a museum specimen. In contrast, the trap images are generally more homogeneous, as they were captured against the white screen of a moth trap. Three representative images were chosen from this category.

Padding Methods

[32] See Jupyter Notebook:
/jupyter_notebooks/05_resize_images/experiment/0_resize_image(padding).ipynb

1. Mean Color The first padding method (Mean Color) begins by loading the original rectangular image and calculating the arithmetic mean for the red, green, and blue (RGB) color channels. Using these values, a solid-colored square canvas is initialized with a side length corresponding to the longest dimension of the original image. Finally, the rectangular image is centered onto this generated square canvas.

The result is a square image with the original rectangular image at its center, augmented by background areas that match the average RGB values of the source image.

2. Small Blur The second padding method begins by loading the original rectangular image and generating a square canvas with a side length equal to the image's longest dimension. This canvas is then filled with 32×32 pixel patches, which are randomly sampled[33] from the original image and processed with a Gaussian blur. Finally, the original image is centered on the generated square background.

The result is a square image with the original rectangular image at its center, augmented by areas filled with small blurred, representative patches derived from the original image itself.

3. Large Patches The third padding method begins by loading the original image and generating a square canvas with a side length equal to the image's longest dimension. This background is then filled with nine large patches that are randomly sampled[34] from the original image. Finally, the original image is centered on the generated square background.

The result is a square image with the original rectangular image at its center, augmented by large areas of the original image itself.

Visualization of Preprocessing Pipeline Cropping To evaluate the influence of different padding methods on the DINOv2 embeddings, the DINOv2 (Small) model was loaded via the timm library along with its standardized preprocessing pipeline. To visualize the impact of the preprocessing pipeline's cropping on both the original and square images, the normalization (mean and standard deviation) was reversed. Consequently, after processing the images through the pipeline, only the cropping to 518×518 pixels and the interpolation remained active. The resulting output reveals the specific image area that DINOv2 encodes into an embedding.

Visualization of PCA-based Segmentation Local (patch) embeddings[35] were extracted for both the rectangular images and the generated square images. A Principal Component Analysis was subsequently performed on these local embeddings.

[33] To ensure reproducibility, a fixed seed is utilized during the sampling process.
[34] To ensure reproducibility, a fixed seed is utilized during the sampling process.
[35] For a more detailed explanation of local embeddings, see Chapter 3.5.

The results are visualized as a heat map on a 37×37 grid, where each cell corresponds to the first principal component of its respective local (patch) embeddings (feature vector). In this representation, positive values are displayed in red, while negative values are shown in blue. This visualization allows for a qualitative assessment of how effectively the model distinguishes the object from the background under different padding methods.

Comparative Visualization Finally, all procedural steps and their corresponding images were compiled into a side-by-side comparison using Adobe InDesign[36], as illustrated in Figure 8. All side-by-side comparisons of all examples processed in this experiment can be found in the appendix of this thesis.

3.4 Generation of the Resized Dataset (Squared)

Based on the findings of the previous padding experiment, the most effective method was selected (3. Large Patches) to generate a secondary square image dataset from the original AMI dataset. By ensuring a square aspect ratio, the images bypass the cropping mechanism of the DINOv2 preprocessing pipeline, thereby preserving the entire visual information.

Implementation The implementation[37] relied on the CSV files generated during the initial dataset split. For each entry in these files, the corresponding image was loaded and transformed into a square format using the Large Patches padding strategy. The resulting images were saved to a dedicated directory. This process yielded a resized dataset suitable for comparative analysis alongside the original rectangular data.

3.5 Embedding Extraction

To implement the classification models, global embeddings ([CLS] Token) for each image are generated independently for both the original and the resized datasets. As previously noted, the visual foundation model DINOv2 (Small) is utilized for this purpose. With approximately 22.1 million parameters, it represents the most computationally efficient variant within the DINOv2 family.

The model extracts embeddings by partitioning the input image into a grid of 14×14 pixel patches. Each patch is individually analyzed and encoded into a feature vector (patch embedding). These local patch embeddings are subsequently processed to capture spatial dependencies, resulting in a global feature vector ([CLS] Token)

[36] https://www.adobe.com/products/indesign.html (visited on 08/02/2026)

[37] See Jupyter Notebook:
/jupyter_notebooks/05_resize_images/0_images_resize_high.ipynb
/jupyter_notebooks/05_resize_images/1_images_resize_low.ipynb

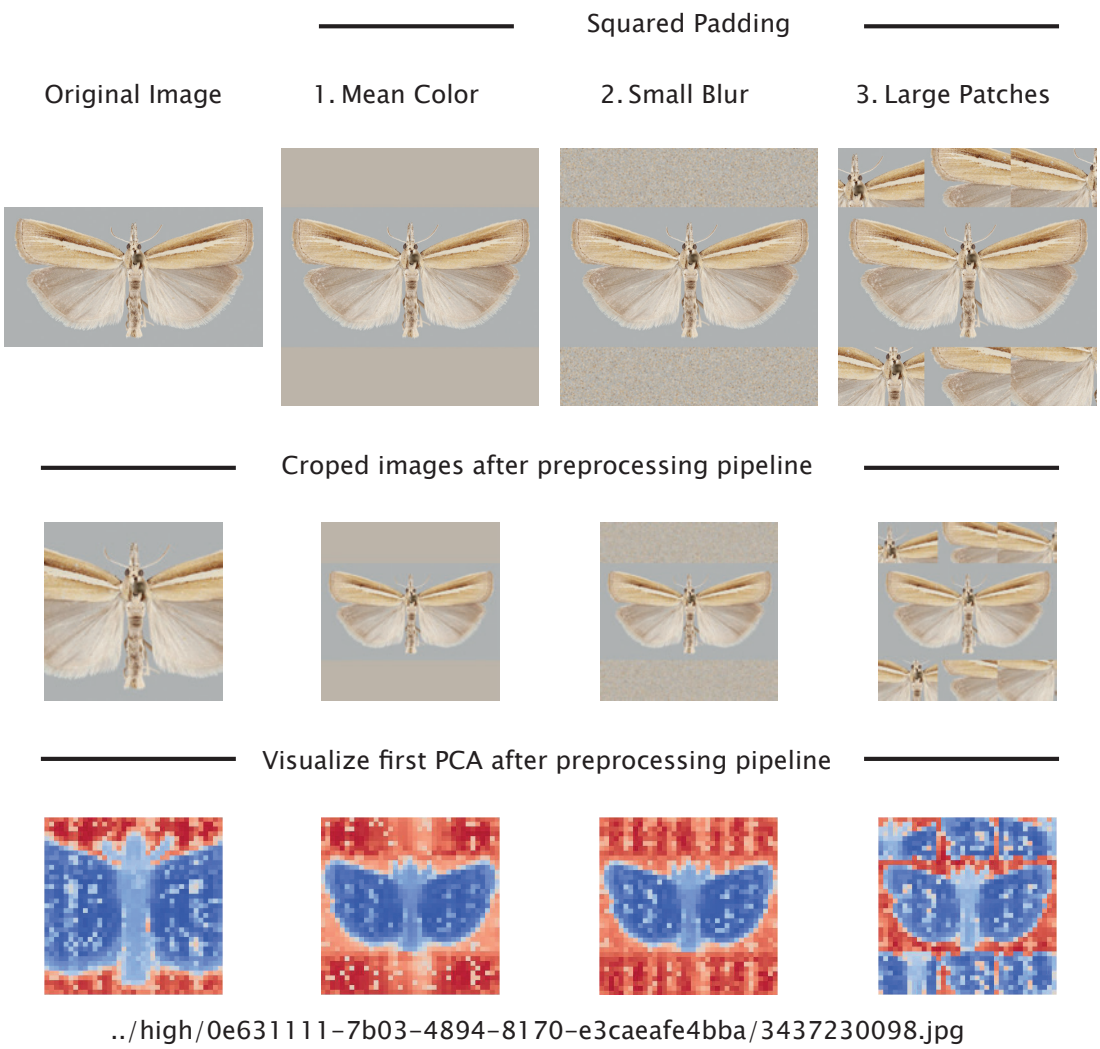


Figure 8: Overview of padding strategies sample one: The first row illustrates the different padding strategies compared to the original image. The second row displays the images after being processed by the preprocessing pipeline, and the third row presents the corresponding segmentation results.

with 384 dimensions. While the model produces both local patch and global embeddings, this study specifically utilizes the global embeddings for the downstream classification tasks. (Oquab et al., 2024)

Implementation The timm library was employed to implement the DINOv2 (Small) model, facilitating the loading of both the pre-trained model and the associated preprocessing pipeline. The workflow for embedding extraction was adapted from existing Jupyter notebooks used in the AMI-GBIF image download process[38].

To track the generation of embeddings, two columns, "embedding_created" and "embedding_created_fail", were maintained within the DataFrame. Furthermore, the extraction process was optimized by leveraging Python's concurrent module to parallelize the generation of embeddings, reducing processing time.

[38]See the Jupyter Notebooks:
/jupyter_notebooks/04_create_embeddings/1_create_Embeddings_copied_high.ipynb
/jupyter_notebooks/06_resized_embeddings/1_create_Embeddings_resized_high.ipynb

4 Methods (Classifier)

The following Chapter outlines the implementation, application, and score calcuation of three classification models: K-Nearest Neighbors (KNN), Multi-Layer Perceptron (MLP), and Deep K-Nearest Neighbors (DKNN). Each classifier was built on the embeddings from the original and resized datasets.

4.1 KNN Classifier

This section details the KNN classifier for original and resized dataset.

The classification results of the KNN are relevant for the DKNN classifier as well, which build on the KNN classifier and utilizes class-specific thresholds for its application. Consequently, all four test datasets were processed through the KNN, and the outputs were exported to CSV files for further use.

Memory Constraints A notable advantage of KNN classifiers is that classification occurs directly based on existing training examples (lazy learning). Typically, the entire training set is stored in an optimized format within the main memory to facilitate an efficient search for the K-Nearest Neighbors. (Cunningham and Delany, 2021, p. 1–2)

In this study, however, the embeddings of the training dataset total 237 GB significantly exceeds the available RAM of the system used (64 GB). Consequently, storing the full set of high-dimensional embeddings in memory is not feasible.

An alternative approach is to utilize a precomputed distance matrix. In this configuration, only the matrix of pairwise distances between samples is loaded into memory, rather than the high-dimensional feature vectors themselves. This can reduce the memory footprint and eliminate the need for distance calculations during the classification phase, as the values are already computed.

4.1.1 Implementation Steps

1. **Training Distance Matrix:** The pairwise distance matrices (Train × Train) was computed to facilitate the creation of the KNN classifier.
2. **Model Initialization:** The KNN classifier was then initialized using the precomputed distance matrix (via the *fit* method).
3. **Test Distance Matrix:** For the classification of test data, a separate distance matrix between the training and test sets (Train × Test) was calculated.
4. **Hyperparameter Tuning:** Prior to the final evaluation, the validation set was used for hyperparameter tuning to determine the optimal number of neighbors (k).

5. **Inference:** The classifier, configured with the optimized k-value, was then applied to all test datasets.

6. **Metric Calculation:** Finally, performance metrics were calculated to evaluate the results for the ID test sets.

1. Distance Matrix Computation The computation of the distance matrices[39] was performed iteratively using subsets of two respective datasets (Dataset 1 × Dataset 2). This batch-wise approach was necessary because the memory requirements of the GBIF Fine-Grained training dataset exceeded the available RAM.

The loaded subsets of embeddings were first normalized using the L2 norm. Subsequently, pairwise cosine distances were calculated. The computation was implemented using the *sklearn.metrics.pairwise_distances* function, which utilized parallelization to significantly accelerate the process. The resulting distance values were subsequently stored in a pre-initialized matrix.

For the square Train × Train distance matrix, the main diagonal was manually set to zero. This was done as rounding errors occasionally resulted in marginal deviations from zero. Table 3 provides an overview of all distance matrices computed within the scope of this work.

Original Dataset	**Matrix Shape**	**Memory**	**Time**
Training x Training	85,803 x 85,803	28 GB	11:14 h
Training x Validation	85,803 x 8,949	2,9 GB	01:43 h
Training x Test FG (ID)	85,803 x 17,832	5,7 GB	05:49 h
Training x Test FG+DG (ID)	85,803 x 2,138	700 MB	03:32 h
Training x Test ND (OOD)	85,803 x 7,213	2,4 GB	01:15 h
Training x Test ND+DG (OOD)	85,803 x 698	229 MB	00:30 h
Resized Dataset	**Matrix Shape**	**Memory**	**Time**
Training x Training	85,803 x 85,803	28 GB	12:20 h
Training x Validation	85,803 x 8,949	2.9 GB	01:57 h
Training x Test FG (ID)	85,803 x 17,832	5.7 GB	02:49 h
Training x Test FG+DG (ID)	85,803 x 2,138	700 MB	00:33 h
Training x Test ND (OOD)	85,803 x 7,213	2,4 GB	01:07 h
Training x Test ND+DG (OOD)	85,803 x 698	229 MB	00:32 h

Table 3: Summary of calculated distanz matrixes. The table details the matrix shapes, memory consumption, and computation times for calculated distance matrixes.

[39] See Jupyter Notebook:
/jupyter_notebooks/08_build_models/2_calculate_distanc_matrix_knn.ipynb

2. KNN Model Fitting To utilize the precomputed distance matrix (Train × Train) with the scikit-learn (Pedregosa et al., 2011) KNeighborsClassifier, the model had to be initialized using the *fit* method[40]. During this process, it can be assumed that the classifier internally validates the matrix structure and creates a mapping between the matrix indices and the corresponding class labels.

The resulting model instance was then saved in the same directory as the source distance matrix, allowing it to be loaded directly for subsequent classification tasks without re-initialization.

3. Hyperparameter Tuning for k To optimize the KNeighborsClassifier, the hyperparameter k (number of neighbors) was tuned. Using the validation dataset, the Balanced Accuracy score was evaluated across a search space of $k \in [1, 529]$[41].

The upper limit of $k = 529$ is derived from the constraints of the training set: the minority class contains 529 instances. Consequently, this represents the natural ceiling for k, as it is the maximum number of potentially correct neighbors available for every class in the dataset.

As illustrated in Figure 9, the classifier builds on the original embeddings data achieved its peak performance at $k = 10$. In contrast, the model utilizing resized embeddings data reached its optimum at $k = 13$. Consequently, these specific k-values were selected as the fixed hyperparameters for the respective configurations in all subsequent evaluations.

4. Inference Test Data The generated KNN models for both the original and resized datasets were employed to classify the four test sets: Test FG (ID), Test FG + DG (ID), Test ND (OOD), and Test ND + DG (OOD)[42]. An overview of these classification processes is provided in Figure 10.

The classification outputs were exported to individual CSV files. Each file contains the predicted class, the ground-truth label, and the corresponding image path. Furthermore, detailed metadata for the K-Nearest Neighbors used in each decision were recorded, including their respective labels, cosine distances, and image paths. The latter information is relevant for the subsequent DKNN classifier, which builds upon these KNN-derived neighbors.

5. Performance Evaluation The classification outcomes for the Test FG (ID) and Test FG + DG (ID) datasets were leveraged to generate the corresponding confusion matrices. Furthermore, the metrics Precision, Recall, and F1-Score were com-

[40] See the Jupyter Notebook:
/jupyter_notebooks/08_build_models/3_fit_database_knn.ipynb

[41] See Jupyter Notebook:
/jupyter_notebooks/09_test_models/1_find_hyperparameter_k_knn.ipynb

[42] See Jupyter Notebook:
/jupyter_notebooks/09_test_models/2_predict_testdata_knn.ipynb

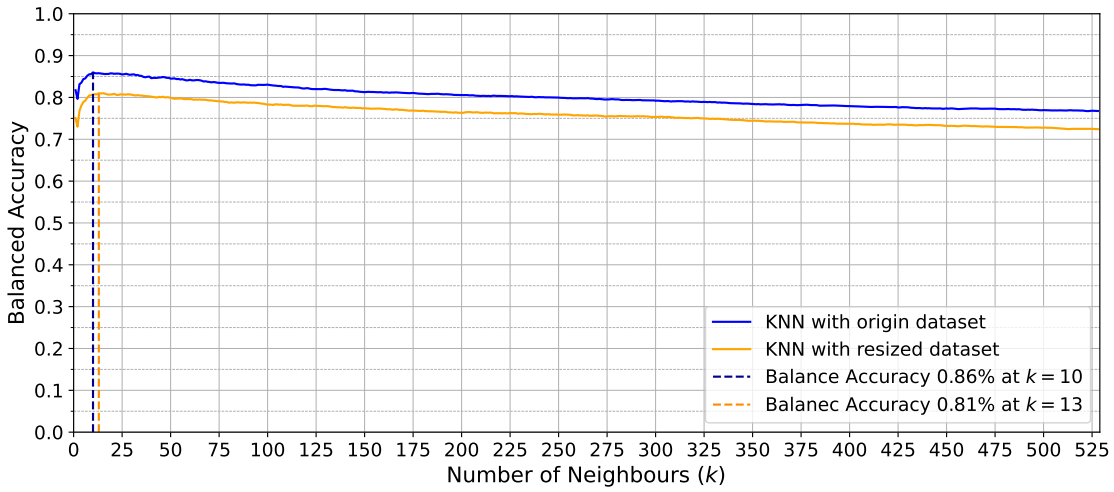


Figure 9: Comparison of the classification performance with varying numbers of $k \in [1, 529]$ neighbors for both origin and resized dataset

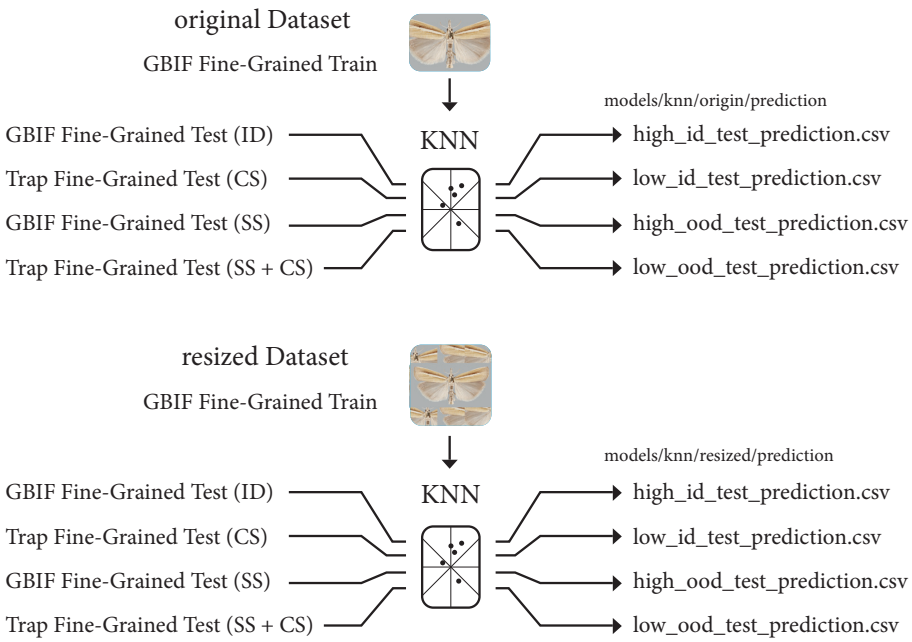


Figure 10: The figure provides an overview of the two KNN models (original and resized) and the corresponding test datasets used for performance evaluation.

puted. These metrics were determined at the sample level as well as through class-level aggregation, specifically employing micro and macro-averaging. The scikit-learn library (Pedregosa et al., 2011) was employed for these calculation[43].

4.2 MLP Head

This section details the MLP classifier for original and resized dataset.

The classification results of the MLP are relevant for the DKNN classifier as well, which utilizes class-specific thresholds for its application. Consequently, all four test datasets were processed through the MLP, and the outputs were exported to CSV files for further use.

4.2.1 Implementation Steps

1. **Architecture Initialization:** The MLP model was defined and initialized.
2. **Training Configuration:** The training setup was established specifying hyperparameters, loss functions, and optimizers.
3. **Convergence Analysis:** Model convergence was presented and discussed based on the classification scores achieved during training.
4. **Inference:** The test datasets were processed using the optimal model weights identified during training.
5. **Metric Calculation:** Performance metrics were calculated based on the classification results of the ID test sets.

1. Model Architecture The Multi-Layer Perceptron was implemented using the PyTorch library. The input layer consists of 384 neurons, matching the dimensionality of the extracted DINOv2 embeddings. A hyperbolic tangent (tanh) activation function was applied to enable the network to model non-linear relationships within the latent space.

The final output layer comprises 91 neurons, corresponding to the total number of taxonomic classes in the dataset. This layer generates a dedicated output value (logit) for each class, which is subsequently used for classification.

The following code snippet descripe the structure of the MLP model employed:

```
model = nn.Sequential(
        nn.Linear(384, 192),
        nn.Tanh(),
        nn.Linear(192, 91),
        )
```

[43]See Jupyter Notebook: /jupyter_notebooks/09_test_models/3_calculate_scores.ipynb

2. Training Setup The training procedure and the developed Python code[44] were guided by the fine-grained classification experiments of (Jain et al., 2024) and the implementation examples in (Stevens, Antiga, and Viehmann, 2020). To ensure reproducibility, a fixed deterministic seed was established. The model was trained with a batch size of 128 for 30 epochs. To address class imbalance, a WeightedRandomSampler was employed, ensuring that each class had an equal probability of being represented within each training batch.

Cross-Entropy Loss with Label Smoothing was selected as the objective function to mitigate overfitting. Optimization was performed using the AdamW optimizer with a weight decay of 1×10^{-5} ($1e-5$) and an initial learning rate of 0.001. The learning rate followed a scheduling strategy consisting of a linear warm-up, which increased over the first two epochs and then decreased via a cosine annealing phase over the remaining 28 epochs.

3. Training Dynamics and Discussion Following each epoch, the classification performance on the validation set was assessed using the Balanced Accuracy metric. A model checkpointing strategy was employed, ensuring that the model were saved whenever a new peak in validation Accuracy was achieved.

A thorough examination of the metrics attained by each epoch reveals a nearly identical behaviour exhibited by both datasets (original and resized). It has been demonstrated that maximum performance is typically achieved within the initial third of the training process.

The resized model reached its maximum Balanced Accuracy of 92.22% as early as the 7th epoch. Similarly, the original model attained its peak performance of 92.16% in the 9th epoch. These results indicate that the performance discrepancy between the two models on the validation dataset is negligible.

In contradistinction, the evaluation of performance on the training set following each epoch discloses substantial discrepancies between the two models. While the resized model initially achieves scores comparable to the original model due to the early-stage, its training Accuracy eventually stabilizes at a level approximately 10 percentage points lower than that of the original model.

An unusual phenomenon is observed with the resized dataset model. The Accuracy on the unseen validation set is approximately 5 percentage points higher than the Accuracy on the training set. Since this discrepancy does not occur with the original model – despite identical training configurations – it suggests that the data preprocessing (i.e., padding) fundamentally alters the model's learning behavior.

One difference between testing the validation and training data sets after each epoch lies in the sampling strategy used. The training process employs a WeightedRandomSampler for class balancing, whereas the validation set is evaluated using its natural (unbalanced) distribution.

[44] See Jupyter Notebooks:
/jupyter_notebooks/08_build_models/0_train_mlp.ipynb

The loss curves for both models (original and resized) exhibit similar convergence patterns, with the original model maintaining a slightly lower overall loss. The following Figure 11 illustrates the training progress, which is regulated by the cosine annealing scheduler.

4. Inference on Test Datasets The trained models (original and resized) were utilized to perform inference on the following test datasets: FG (ID), FG + DG (ID), ND (OOD), and ND + DG (OOD). The classification results for each dataset were exported to distinct CSV files. For each sample, the image path, the predicted class, and the ground truth label were stored[45].

4. Performance Evaluation The classification outcomes for the FG (ID) and FG + DG (ID) test datasets were utilized to generate the corresponding confusion matrices. Furthermore, the metrics Precision, Recall, and F1-Score were calculated. These metrics were computed both at the sample level and in aggregated forms at the class level, specifically providing micro- and macro-averages. The scikit-learn library was employed for these calculations[46].

4.3 DKNN Classifier (OOD)

This section details the DKNN classifier for original and resized dataset. The approach leverages the classification results from the previously described KNN and MLP models.

4.3.1 Implementation Steps

1. **Threshold Computation:** Three distinct threshold strategies were defined and calculated for OOD detection.
2. **DKNN Classification:** The calculated thresholds were applied to the ID and OOD test sets, leveraging the results from the initial KNN and MLP classification.
3. **Metric Calculation:** Performance metrics were calculated based on the predicted results of the ID and OOD test sets.

1. Threshold Definition To perform OOD classification using the DKNN method, establishing an appropriate decision threshold is essential. This study implements three distinct calculation strategies for both the original and resized datasets, which are detailed in the following sections.

[45] See Jupyter Notebook:
/jupyter_notebooks/09_test_models/0_predict_testdata_mlp.ipynb

[46] See Jupyter Notebook:
/jupyter_notebooks/09_test_models/3_calculate_scores.ipynb

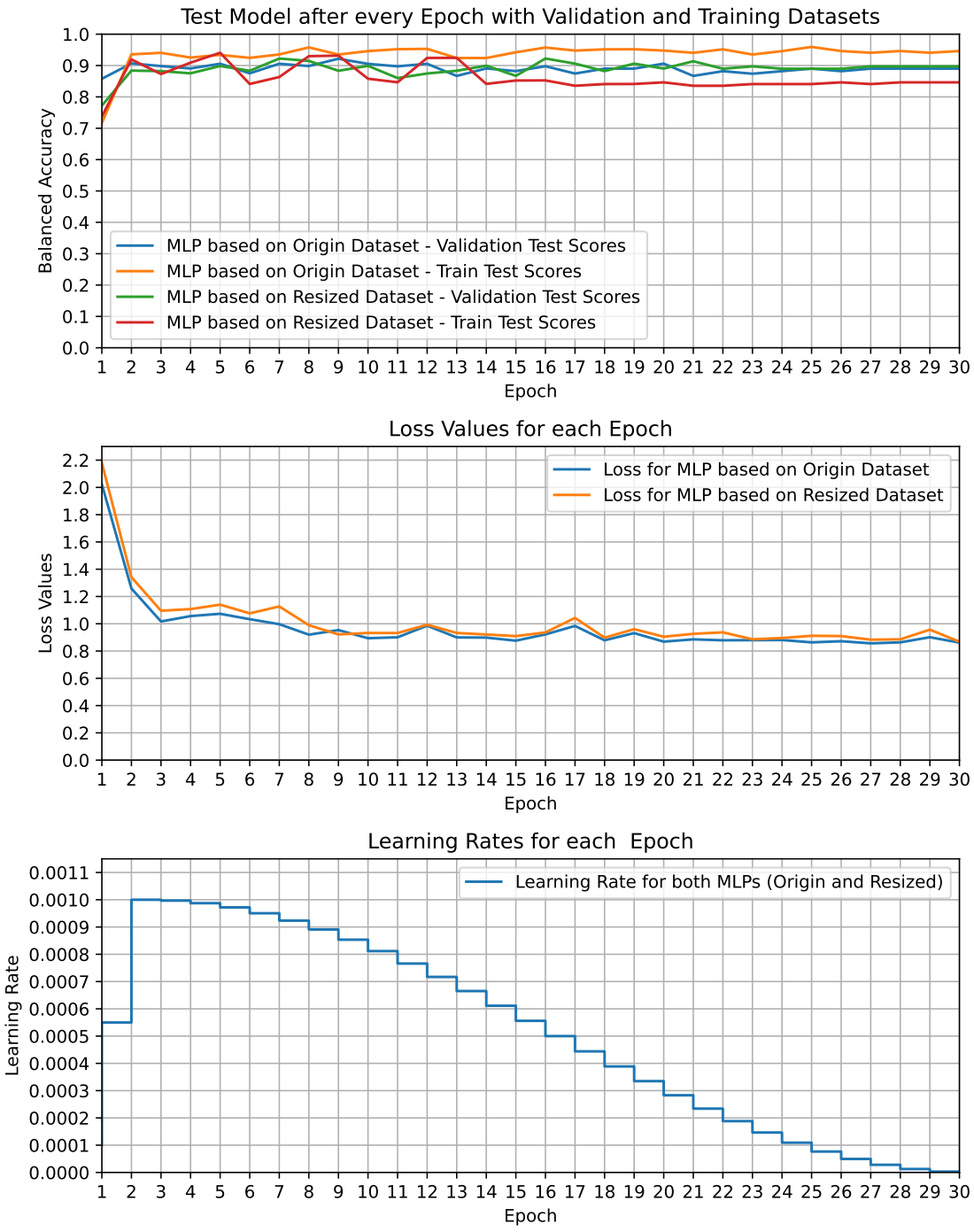


Figure 11: Learning curves of the MLP training process. The plots display the Balanced Accuracy for the training and validation sets, loss values, and the learning rate per epoch for both models based on the original and resized dataset.

Global Threshold (*over_all_examples*) The first strategy for calculating thresholds utilizes the KNN classification results derived from the validation dataset. Only those samples that were correctly classified by the KNN classifier

were taken into account. All incorrectly classified samples were excluded. Consequently, the threshold calculation only incorporates examples where the vast majority of the K-Nearest Neighbors belong to the same class.

For each correctly classified instance, the distance to the k-th neighbor in the knowledge base was determined and recorded. This collection of k-th neighbor distances was subsequently sorted in ascending order.

The specific threshold is derived by selecting a particular percentile from the sorted distance distribution. For instance, selecting the 95th percentile ensures that 95% of the correctly classified validation samples fall below this limit and are correctly identified as ID. Conversely, the remaining 5% exceed this threshold and are incorrectly classified as OOD. This demonstrates the inherent trade-off between true positives (TP) and false negatives (FN) when choosing an appropriate threshold.

To systematically evaluate the impact of the threshold on classification performance, values were calculated across a range of percentiles from 0.00 to 1.00 in incremental steps of 0.01. This procedure yielded a total of 100 discrete global thresholds.

Class-Specific Threshold (*for_each_species*) Unlike the first method, which utilized a global threshold across all categories, this second approach implements class-specific thresholds.

Analogous to the global method, this calculation relies on the results of the validation set generated by the KNN classifier. Again, only correctly predicted samples were included in the computation. This ensures that the thresholds are derived from instances where the majority of the K-Nearest Neighbors belong to the same class[47].

The correctly classified samples were partitioned according to their class membership. A dedicated distance list was generated for each class, capturing the distances to the k-th nearest neighbors.

These class-specific lists were sorted in ascending order, and the threshold values were derived by selecting the corresponding percentiles. Consistent with the first methodology, this selection was performed iteratively across a range of percentiles from 0.00 to 1.00 with increments of 0.01. This procedure yielded 100 unique thresholds for each individual class.

Class-Specific Threshold (*in_each_species*) The third method introduces a change in the underlying data source compared to the previous approaches. Rather than relying on the results of the validation dataset, this strategy utilises the distance matrix[48] of the training set to compute the thresh-

[47] In cases where a class contained no correctly classified examples, the model defaulted to the global "threshold for all species" as a fallback mechanism.

[48] The training distance matrix encompasses all pairwise distances between instances within the training set

old values. Analogous to the second method, a unique threshold is determined for each individual class.

In the initial step, only intra-class distances – between samples of the same species – were extracted from the matrix. For every training sample, the distance to its k-th nearest neighbor within its own class was determined. These values were then saved into class-specific distance lists.

This approach constructs a controlled framework that reflects the local density of samples within their respective classes in the training set. It evaluates the spatial distribution of the k intra-class neighbors and their corresponding distances within the latent space.

The distances list was sorted in ascending order, and the threshold was determined by selecting a specific percentile. Consistent with the previous methods, a range of percentiles from 0.00 to 1.00 was calculated with an increment of 0.01. This process was iterated across all classes, yielding 100 unique threshold per species.

Unlike the strategies described above, these thresholds are derived solely from the training data. Consequently, the model's performance on the validation set is not a determining factor, focusing instead on the intrinsic cluster compactness of the training data.

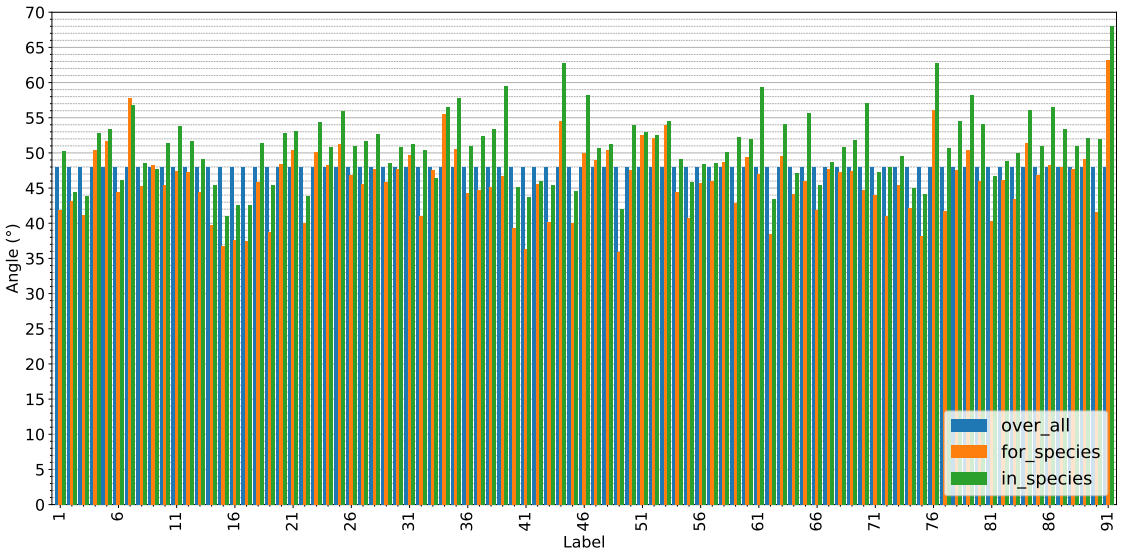


Figure 12: Illustration of the three distinct threshold calculation methods applied to the original dataset, based on the 95th percentile across all classes.

To provide an initial comparative overview of the calculated thresholds, the values corresponding to the 95th percentile are visualized in Figure 12 for the original dataset and Figure 13 for the resized dataset.

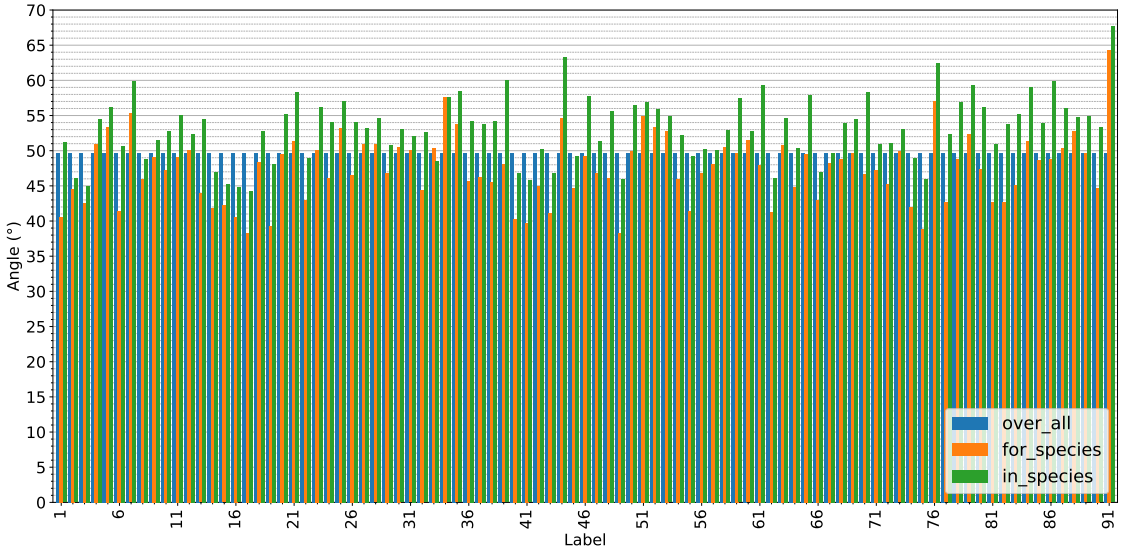


Figure 13: Illustration of the three distinct threshold calculation methods applied to the resized dataset, based on the 95th percentile across all classes.

2. Inference on Test Data To implement the DKNN, the results from the preceding classification were utilized for all test scenarios, including the OOD sets (Test ND, Test ND+DG) and the ID sets (Test FG, Test FG+DG). For each test sample, the KNN output provided both the cosine angular distances to the K-Nearest Neighbors and the predicted class. In contrast, the MLP output was just used to retrieve the predicted class label for subsequent class-specific threshold selection.

For the global threshold the distance to the k-th nearest neighbor of a test sample was compared against calculated global thresholds. If the distance is within the threshold limit, the sample is classified as ID. Conversely, if the distance exceeds the threshold, the sample is categorized as OOD.

In the case of class-specific thresholds, the selection of the threshold value was dependent on the preceding class prediction. Therefore, first the class predicted by the classifier KNN or MLP was identified before selecting the corresponding class-specific threshold.

Consequently, each test sample was evaluated across 100 incremental threshold levels (ranging from the 1st to the 100th percentile) for the global threshold approach. For the class-specific thresholds, a total of 100 evaluations were performed for each classifier (KNN and MLP).

The implementation[49] of this comparison was primarily vectorized using a Pandas DataFrame structure (McKinney, 2010), in which all relevant information was already contained or had to be merged first.

3. Performance Metrics Calculation The results from the DKNN classification were utilized to compute the coordinates for the ROC curves and to determine the Area Under the Curves (AUROC). Additionally, the FPR95 (False Positive Rate at a True Positive Rate of 95%) was calculated.

To construct the ROC curves, the TPR was computed using the OOD classification results (Test ND, Test ND+DG) and the FPR was determined using the ID classification results (Test FG, Test FG+DG) across a sequence of 100 thresholds. To ensure a complete curve, the boundary points (0,0) and (1,1) – representing the minimum and maximum thresholds – were added, resulting in 102 coordinates. Since a TPR of exactly 95% often fell between two discrete thresholds, the FPR95 score was determined via linear interpolation.

This procedure was performed for all three thresholding strategies for each experiment (Novelty Detector and Domain Generalization & Novelty Detection) across both the original and resized datasets. Figure 14 summarizes the various experimental scenarios for which OOD classification curves and scores were evaluated[50].

[49] See Jupyter Notebook:
/jupyter_notebooks/10_OOD/2_predict_ood.ipynb

[50] See Jupyter Notebook:
/jupyter_notebooks/10_OOD/3_calculate_scores_ood_detection.ipynb

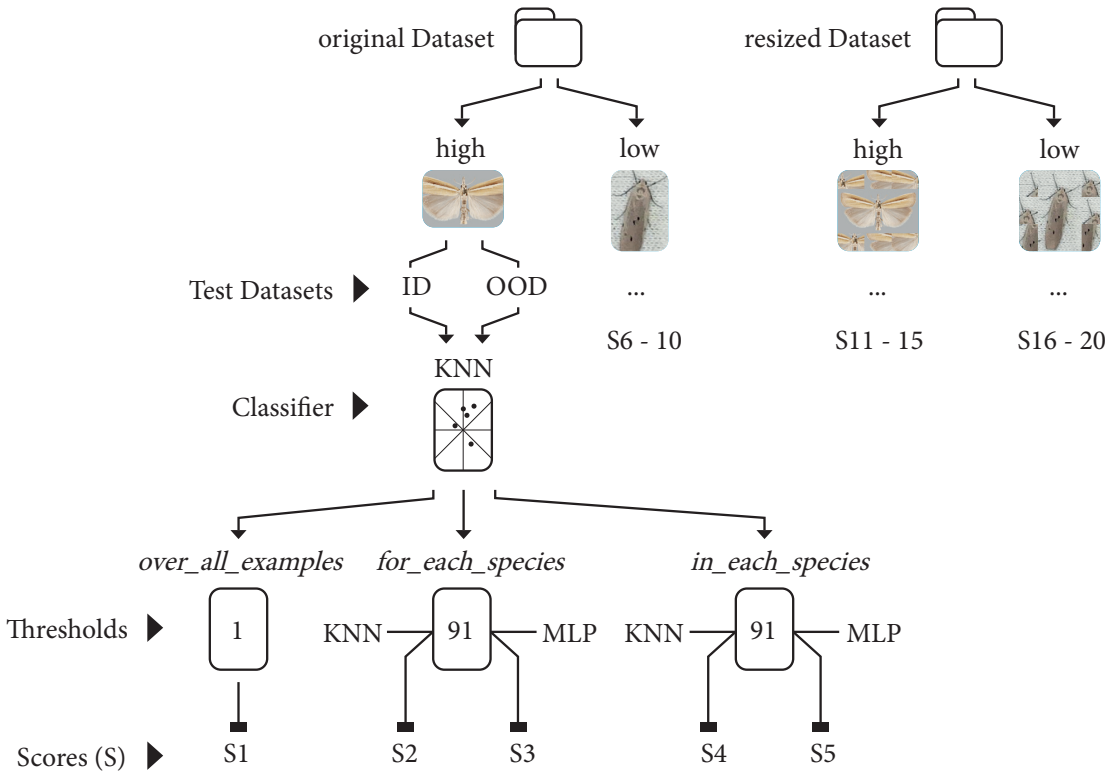


Figure 14: OOD Classification Process. The diagram outlines the evaluation of four test sets (FG, FG+DG, ND, ND+DG) using five threshold variants: global (*for_all_species*) and class-specific (*for_each_species* and *in_each_species*, both with MLP and KNN). Results are labeled S1–S10 for original datasets and S11–S20 for resized datasets.

5 Results

The following chapter presents the experimental results of this study, organized into seven subchapters that each address a distinct research question.

5.1 AMI Dataset Availability

The AMI dataset is composed of two primary subsets, the AMI-GBIF and AMI-Trap. The AMI-GBIF subset comprises high-resolution images from various providers, which must be retrieved via individual web links and downloaded manually. In contrast, the AMI Trap subset contains low-resolution camera trap images that are provided in their entirety within the distributed download archive.

The initial objective of this study was to verify the accessibility and completeness of the AMI dataset. Particular emphasis was placed on the AMI-GBIF subset; since its images are hosted across numerous external sources, any broken links could potentially compromise the reproducibility and long-term utility of the AMI dataset as a standardized benchmark.

Availability of the AMI-GBIF The AMI-GBIF subset is further categorized into two components: GBIF Fine-Grained and GBIF Binary. While the former is available for various geographic regions, this study utilized the "all regions" split, which aggregates all subregions. The second consists solely of a single split. The retrieval process for the URLs for each subset yielded the following results:

GBIF Fine-Grained
A total of 2,564,392 image URLs were processed for the GBIF Fine-Grained dataset, resulting in 2,520,912 successful downloads. The 43,480 failed or corrupted downloads represent an attrition rate of 1.69% of the total dataset. Failure rates across the individual partitions were highly consistent: 1.70% for the training set, 1.70% for the validation set, and 1.66% for the test set. These results demonstrate a nearly uniform distribution of missing data across all dataset splits.

GBIF Binary For the GBIF Binary dataset, 700,000 image URLs were processed, of which 692,311 were successfully retrieved. A total of 7,689 images (1.09% of the total dataset) could not be accessed or were corrupted during download. Consistent with the previous findings, the failure rates across the partitions remain stable: 1.08% for the training set, 1.14% for the validation set, and 1.12% for the test set. Consequently, the missing data is distributed almost uniformly across all splits.

A detailed breakdown of successful and unsuccessful downloads for both datasets is illustrated in Figure 15.

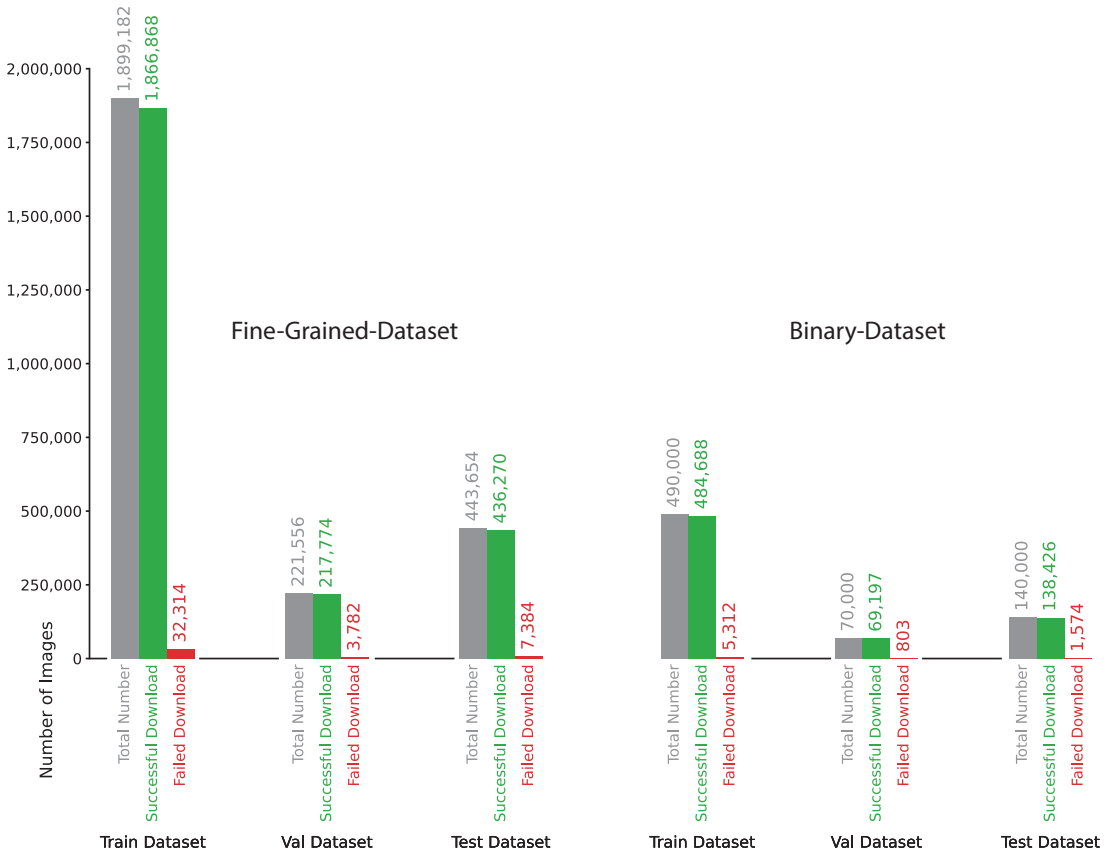


Figure 15: Illustrates the number of successful and unsuccessful downloads of the GBIF Fine-Grained and Binary dataset.

5.2 Generated Datasets

To address the primary research questions – fine-grained classification, domain generalization, and novelty detection – six distinct subsets were extracted from the AMI dataset. The creation of these subsets significantly reduced the data volume to be processed, thereby lowering the computational requirements. Furthermore, this selection enabled an initial assessment of the investigated classification methods' performance within the context of the AMI dataset, specifically moth classification.

Table 3.2, shown in Chapter 3.2, already provides an overview of the generated subsets. These are detailed in the following sections, which cover their specific properties and applications.

Training The Training dataset was utilized to deploy all three classification models. For the K-Nearest Neighbor (KNN) and Deep K-Nearest Neighbor (DKNN) classifier, it serves as the reference set (knowledge storage) for neighborhood calculations. In the DKNN framework, it is additionally employed to determine the underlying thresholds. For the MLP model, it is used for weight optimization. The dataset comprises 91 classes with a range of 529 to 1,000 examples per class, totaling 85,803 images.

Validation The Validation dataset was used to tune the hyperparameter k for both KNN and DKNN. Within the DKNN pipeline, it is also utilized for threshold calculation. For the MLP, it served as the validation set during training to monitor performance and identify the highest-performing model. The dataset consists of 91 classes with 78 to 100 examples per class, totaling 8,949 images.

Test FG The Test FG dataset was employed to evaluate the fine-grained classification performance of KNN and MLP, as well as to assess the ID classification accuracy of the DKNN. It comprises 91 classes with 138 to 200 examples per class, totaling 17,832 images.

Test FG + DG The Test FG + DG dataset is identical in class composition to the Test FG set but consists of low-resolution images captured by moth traps. This configuration allows for the evaluation of the models (KNN, DKNN, MLP) on identical classes while introducing variations in image quality, known as covariate shift. The dataset comprises 91 classes with 6 to 100 examples per class, totaling 2,138 images.

Test ND The Test ND dataset consists of high-resolution images of species not seen during training (semantic shift). It is utilized for the OOD evaluation of the DKNN by providing instances of novel classes. The dataset comprises 37 classes with 122 to 200 examples per class, totaling 7,213 images.

Test ND + DG The Test ND + DG dataset represents the low-resolution counterpart to the Test ND set. While it maintains the semantic shift (i.e., novel

classes), it introduces an additional covariate shift through the use of low-resolution images from moth traps. This configuration allows for the assessment of the DKNN's OOD classification performance when simultaneously confronted with unknown species and degraded image quality. The dataset comprises 37 classes with 6 to 100 examples per class, totaling 698 images.

Figure 16 provides a comparative visualization overview of all six generated subsets via bar charts.

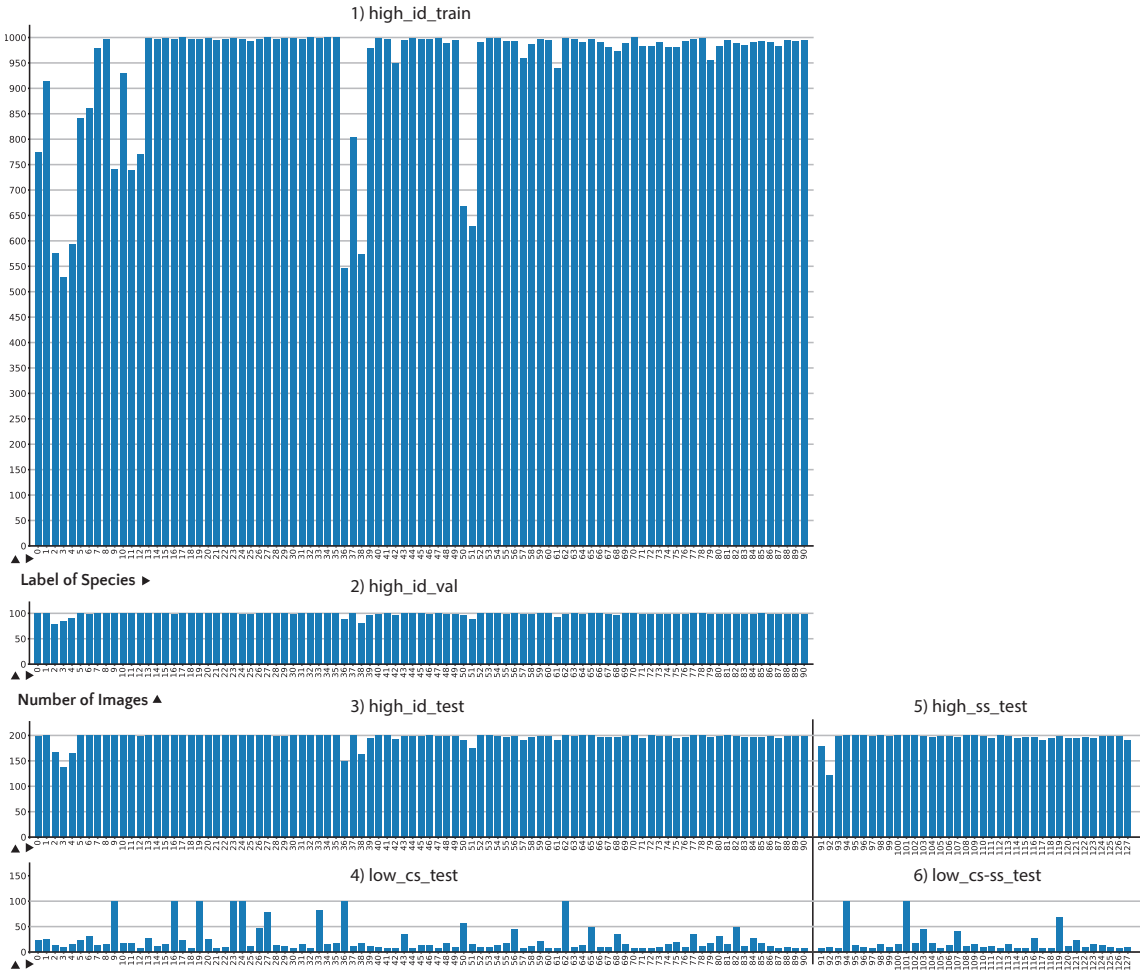


Figure 16: Class distribution and sample counts for all created and utilized datasets

Storage of Image Datasets The high-resolution GBIF imagery accounted for the largest portion of the storage requirements. The initial retrieval of the complete AMI-GBIF subsets (comprising GBIF Fine-Grained and GBIF Binary) required approximately 1.8 TB of disk space.

Following the curation of the datasets specific to this study, the storage footprint for the GBIF data was reduced to 72 GB. Accordingly, the number of GBIF Fine-Grained images decreased from 1,899,182 to 120,192.

In contrast, the storage requirements for the low-resolution trap imagery are negligible. These images are already contained within the AMI dataset archive. While the trap fine-grain images occupied 555 MB in the original dataset, this volume was reduced to 161 MB as a result of the specific constraints applied during dataset creation. Simultaneously, the total number of trap images decreased from 14,405 to 2,836.

5.3 Square Padding (Resized Dataset)

To generate embeddings, images were processed through the DINOv2 preprocessing pipeline, which crops rectangular images into a square format. To prevent the resulting loss of information, three distinct padding methods were investigated. These methods extend rectangular images into a square shape by leveraging internal image content.

The objective was to identify the padding technique that minimizes the impact of supplementary image areas on the embeddings while ensuring that the moth is fully preserved and encoded.

Figure 17 illustrates the effects of these padding methods on one of the six representative images selected for this analysis. Additional examples are provided in Appendix A.2.

Observation Observed variations between the three padding methods regarding the segmentation of local patch embeddings are particularly striking in images with heterogeneous backgrounds (see Figure 17). In these instances, variants 2 (Small Blur) and 1 (Mean Color) produce unnatural edges at the boundaries between the original image and the padded areas. The model tends to categorize these padded areas as background, focusing its encoding on the whole original image itself as the primary object. This, however, diminishes the granularity of feature differentiation within the original image area.

In contrast, variant 3 (Large Patches) utilizes large, unmodified image segments that blend seamlessly with the structure and color characteristics of the original image. The model treats the padding as an integral part of the global context.

While this allows the model to maintain a clear distinction between the moth and the background, it introduces artificial spatial redundancies. Features of the primary object may be duplicated in the padded areas, leading the model to interpret them as belonging to the main object (see Figure 32). This results in distorted spatial relationships between individual body parts and the moth as a whole.

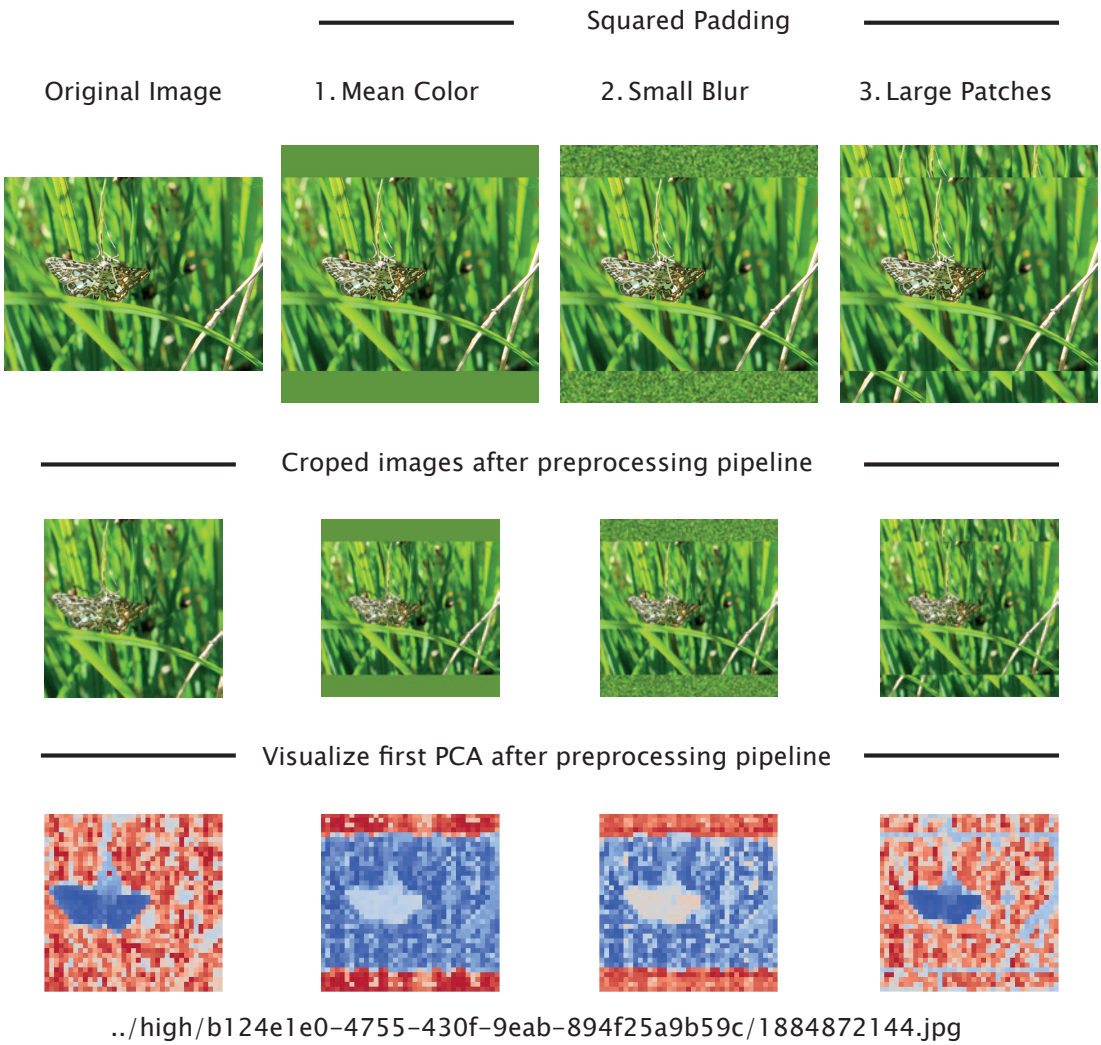


Figure 17: Overview of padding strategies sample two: The first row illustrates the different padding strategies compared to the original image. The second row displays the images after being processed by the preprocessing pipeline, and the third row presents the corresponding segmentation results.

Conclusion Based on these qualitative findings, the third Padding Method (Large Patches) demonstrated the minimal interference with the segmentation of the primary object within the embeddings. Consequently, Variant 3 was employed to generate a secondary square-resized dataset alongside the original image dataset.

5.4 Fine-Grained Classification

In order to address the challenges posed by automated moth monitoring – particularly the high species diversity – two classification methods, KNN and MLP, based on embeddings (DINOv2 Small), were evaluated.

These classifiers were trained and evaluated on both the original and resized datasets, comprising 91 distinct moth species (see Figure 18).

The objective was to assess the classification performance of these methods within the field of moth identification. Specifically, the resized dataset was designed to mitigate the information loss of rectangluar images by the preprocessing pipeline during embeddings extraction, thereby aiming to improve classification performance.

To evaluate and compare the performance of the various classifiers, scores for Macro-Precision, Macro F1-Score, and Balanced Accuracy were used.

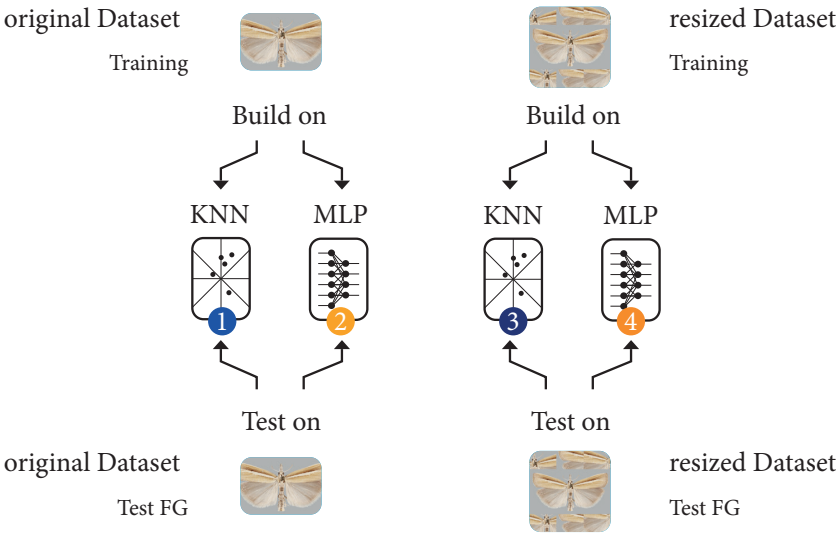


Figure 18: Overview of the experimental setup, detailing the KNN and MLP classifiers alongside their corresponding training and testing datasets

Classifier Comparison A comparison of the two classifiers demonstrates that the MLP models outperform the KNN classifiers across all evaluated metrics. On

the original dataset, the MLP achieved a score of 94% for all performance measures, while the KNN classifier scored 8 percentage points lower at 86%. On the resized dataset, the MLP consistently reached 92%, whereas the KNN classifier's performance was 10 percentage points lower at 82%.

Dataset Comparison A comparison of the original and resized dataset variants demonstrates that the original dataset consistently yields higher scores across both classifiers. When using the MLP classifier, the original dataset outperforms the resized version by approximately 2 percentage points, whereas the performance gap increases to 5 percentage points for the KNN classifier.

Summary In summary, the MLP classifier achieved the highest performance, reaching a score of 94% on all metrics on the original dataset and outperforming the KNN classifier by 8 percentage points. Regarding the datasets, the results indicate that the resized dataset leads to a general degradation in classification performance. This drop is more pronounced for the KNN (a 5 percentage point decrease) than for the MLP, which saw only a 2 percentage point decline compared to the original dataset.

The following Table 4 provides a comprehensive list of calculated scores, while Figure 19 offers a graphical comparison of these results[51].

	Model	Macro Precision	Macro F1-Score	Balanced Accuracy
•	KNN (original)	86.71%	86.22%	86.21%
•	KNN (resized)	82.56%	81.70%	81.67%
•	MLP (original)	**94.44%**	**94.34%**	**94.33%**
•	MLP (resized)	92.47%	0.92.31%	0.92.32%

Table 4: Performance metrics of Fine-Grained classification – including Macro Precision, Macro F1-Score, and Balanced Accuracy – achieved by the KNN and MLP classifiers on both the original and resized datasets. Bold values indicate the best performance for each respective metric.

[51] The confusion matrices presented in Appendix A.3.1 offer a supplementary means of evaluating the classification performance of the applied methods at the class level with greater precision.

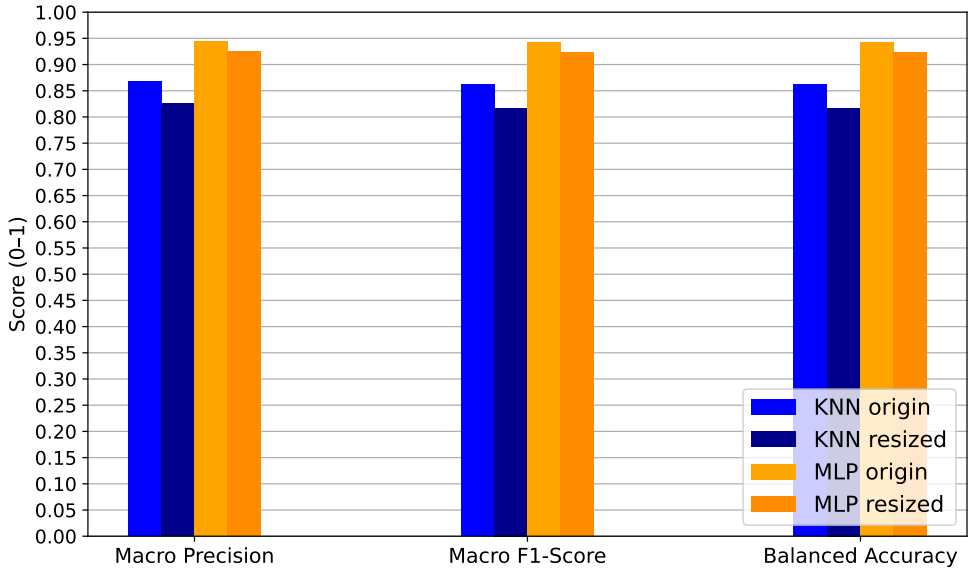


Figure 19: A comparison of Fine-Grained classification results between KNN and MLP models for original and resized datasets. The barchart presents Macro Precision, Macro F1-Score, and Balanced Accuracy.

5.5 Domain Generalization & Fine-Grained Classification

In addition to high species diversity, another significant challenge in automated moth monitoring is the discrepancy between high-resolution training data and the low-resolution images captured by automated traps in the field.

To address this, the KNN and MLP classification methods, based on embeddings, were re-evaluated, comparing their performance on both the original and resized datasets. Unlike the previous fine-grained experiment, the test set used here (Test FG + DG) consists of low-resolution trap imagery covering the same 91 distinct species.

The objective is to assess the robustness of the classifiers against a covariate shift caused by the discrepancy between high-resolution training data and low-resolution test data. As previously outlined, the resized dataset was employed to mitigate information loss during the extraction of embeddings from rectangular images, with the objective of enhancing classification performance.

To ensure comparability, all classifiers and datasets are evaluated using Macro Precision, Macro F1-Score, and Balanced Accuracy.

Classifier Performance Comparison A comparison between the two classifiers demonstrates that the MLP consistently outperforms the KNN classifier, achieving scores up to 8.2 percentage points higher. The MLP attained its peak at Macro

Precision of 46.3% on the original dataset. However, for all other metrics, the MLP classifier yielded its highest results on the resized dataset, reaching 39% in Macro F1-Score, and 43.7% in Balanced Accuracy.

The KNN classifier performed best on the resized dataset, with 42.7% Macro Precision, 32.9% Macro F1-Score, and 35.5% Balanced Accuracy.

Consequently, the MLP classifier surpassed the KNN by 3.6 percentage points in Macro Precision, 6.1 percentage points in Macro F1-Score, and 8.2 percentage points in Balanced Accuracy.

Dataset Performance Comparison A comparison of the dataset variants (original vs. resized) reveals that the resized dataset achieved slightly higher scores for both classifiers, except for the Macro Precision metric.

When using the MLP classifier, the original dataset outperformed the resized version only in Macro Precision by a marginal 0.84 percentage points. In all other metrics, the resized dataset yielded slightly higher values, with differences of 0.79 percentage points in Macro F1-Score, and 0.39 percentage points in Balanced Accuracy.

For the KNN classifier, the benefits of the resized dataset were more pronounced, with increases of up to 2 percentage points across all metrics.

Summary In summary, the MLP classifier outperformed the KNN classifier across all metrics, with improvements ranging from 3.6 to 8.2 percentage points. Regarding the datasets, the resized dataset led to slightly improved results for both classifiers compared to the original dataset. This effect is more pronounced for the KNN classifier (up to 2 percentage points) than for the MLP (up to 0.79 percentage points). A notable exception is the Macro Precision of the MLP, which peaked at 46.35% on the original dataset, surpassing the resized dataset by 0.89 percentage points.

Detailed scores[52] are provided in Table 5, and a visual comparison of these findings is presented in Figure 20.

[52]The confusion matrices presented in Appendix A.3.2 offer a supplementary means of evaluating the classification performance of the applied methods at the class level with greater precision.

	Model	Macro Precision	Macro F1-Score	Balanced Accuracy
•	KNN (original)	41.72%	30.93%	33.51%
•	KNN (resized)	42.72%	32.94%	35.54%
•	MLP (original)	**46.35%**	38.40%	43.39%
•	MLP (resized)	45.51%	**39.19%**	**43.78%**

Table 5: Performance metrics of Domain Generalization classification – including Macro Precision, Macro F1-Score, and Balanced Accuracy – achieved by the KNN and MLP classifiers on both the original and resized datasets. Bold values indicate the best performance for each respective metric.

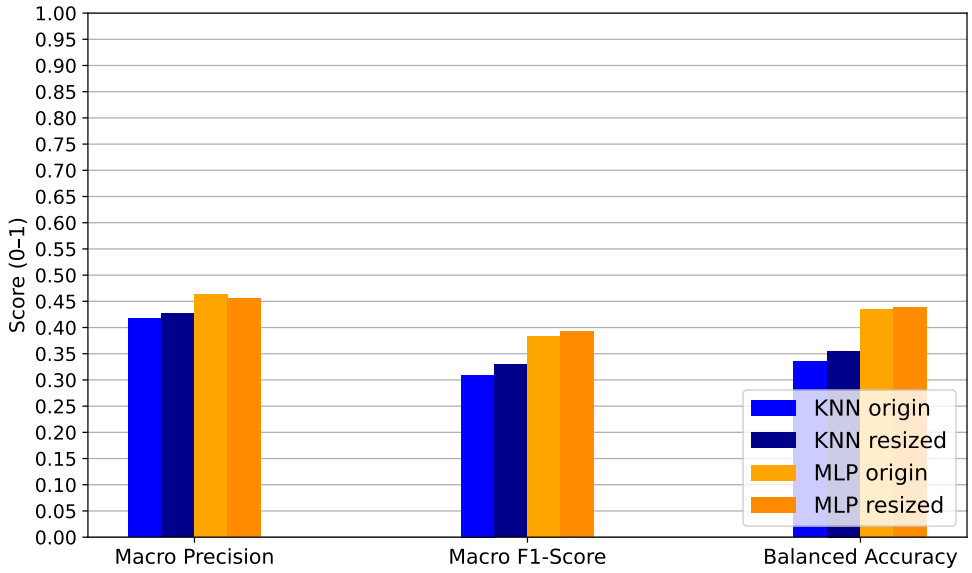


Figure 20: Comparison of Domain Generalization classification results between KNN and MLP models for original and resized datasets. The barchart presents Macro Precision, Macro F1-Score, and Balanced Accuracy.

5.6 Novelty Detection

A third challenge in moth monitoring arises from climate change and global trade, which are shifting the distribution ranges of various insect species. To track these changes via automated monitoring, it is essential to distinguish between regionally known and unknown species.

To address this, a DKNN approach based on DINOv2 embeddings was evaluated for its ability to differentiate between known (ID) and unknown (OOD) moth species using three distinct thresholding variants. As in previous experiments, the method was applied to both the original and the resized datasets. The datasets employed in the testing process comprised the Test FG (ID) and Test ND (OOD) datasets, which encompassed high-resolution images across 91 ID classes and 37 OOD classes.

The objective was to assess the OOD detection performance of these methods within the field of automated moth monitoring. As previously outlined, the resized dataset was employed to mitigate information loss during the extraction of embeddings from rectangular images, with the objective of enhancing classification performance.

For each DKNN variant, ROC curves were plotted, and both AUROC and FPR95 scores were computed. Performance was evaluated and compared with respect to the dataset types (original and resized), the threshold variants, and the classifiers utilised for the class-specific thresholds (MLP and KNN).

To provide an overview of the evaluated DKNN configurations, these are illustrated as a tree diagram in Figure 21.

5.6.1 Original Dataset

The following section presents the performance results of the classifiers trained and tested on the original datasets.

ROC Curves The ROC curves illustrated in Figure 22 are derived from 100 calculated TPR values from the OOD test set (Test ND) and 100 FPR values from the ID test set (Test FG)[53]. The discrete data points are marked on the curves, reflecting the DKNN's detection performance at various thresholds.

All curves exhibit a similar characteristic: They lie above the diagonal, approaching the ideal upper-left corner of the diagram. This indicates that the DKNN classifier consistently achieves a higher TPR than a FPR across all threshold values.

> **Impact of Thresholds on ROC Curves** An analysis of the ROC curves reveals that the *over_all_examples* • threshold exhibits the greatest deviation from the ideal curve (the upper-left corner). Conversely, the class-specific thresholds (*for_each_species* • • and *in_each_species* • •) achieve a closer proximity, with *for_each_species* slightly closer to the optimal curve than *in_each_species* along its trajectory.

[53]The y-axis represents the True Positive Rate, indicating the proportion of OOD samples correctly identified by the classifier. The False Negative Rate is defined as $1 - TPR$, representing OOD samples misclassified as ID. The x-axis shows the False Positive Rate, which is the proportion of ID samples incorrectly flagged as OOD. Consequently, the True Negative Rate is $1 - FPR$, denoting correctly identified ID samples.

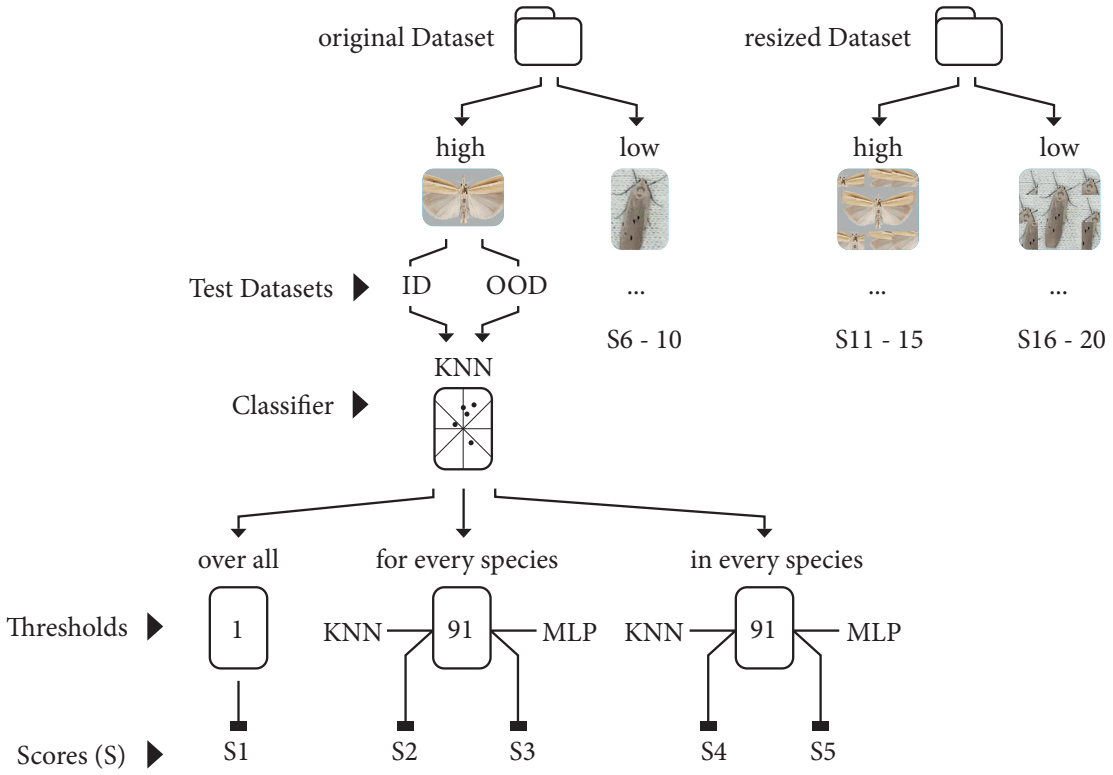


Figure 21: Overview of the evaluated DKNN variants. The experimental setup compares the original dataset (S1–S10) and the resized dataset (S11–S20) across two test sets (test FG and test ND). The variants include a global thresholding strategy (*over_all_examples*) and class-specific strategies (*for_each_species*, *in_each_species*), with the latter being implemented using both KNN and MLP classifiers.

Impact of Classifier on ROC Curves However, the choice of classifier for the class-specific *for_each_species* and *in_each_species* thresholds leads to visible performance gaps. Specifically, the ROC curve using the KNN classifier • • exhibits a stronger curvature toward the ideal ROC curve compared to the curves produced by the MLP classifier • •.

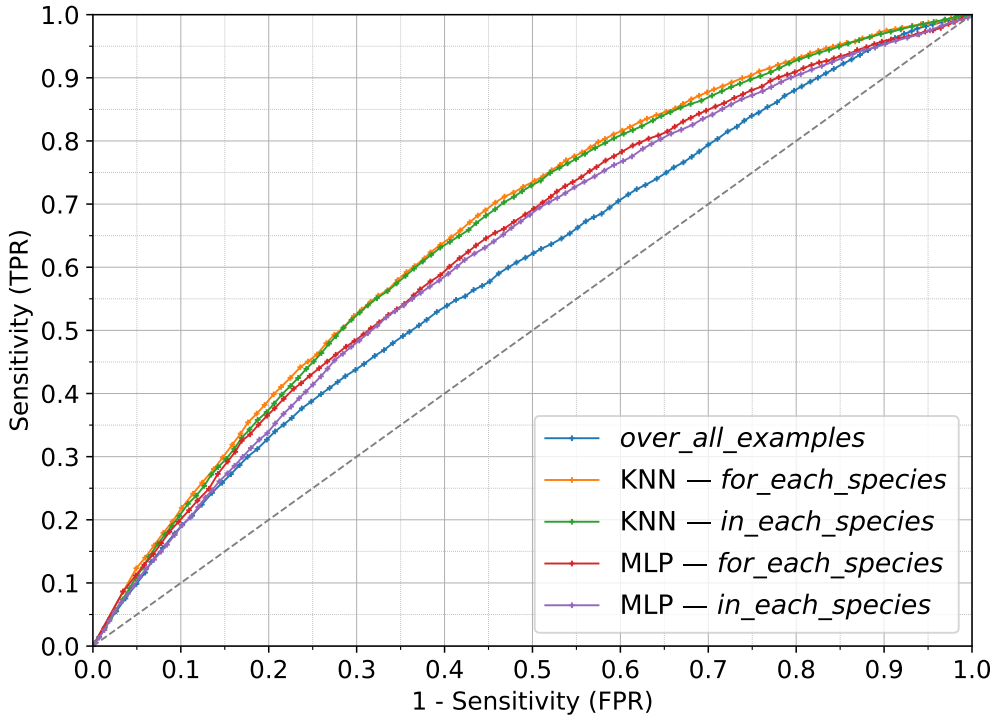


Figure 22: ROC curves for the original dataset using the DKNN method. The plot compares the global *over_all_examples* threshold with the class-specific thresholds (*for_each_species* and *in_each_species*), with the latter two being implemented using both KNN and MLP classifiers. The diagonal dashed line represents the baseline for random classification.

AUROC Scores The AUROC score is employed to quantify classification performance. It represents the area under the ROC curve, effectively condensing the discriminative power of the model into a single scalar value.

The class-specific *for_each_species* • and *in_each_species* • thresholds achieve the highest AUROC scores of 0.6523 and 0.6525, respectively, when paired with the KNN classifier.

Utilizing the MLP classifier results in a performance decrease of 0.028 • and 0.0313 •, respectively. The *over_all_examples* • method yields the lowest AUROC score at 0.5963.

The span of 0,0562 between the highest and lowest scores highlights the varying efficacy of the different thresholding strategies. A comprehensive overview of all AUROC scores is provided in Table 6.

FPR95 Scores The analysis of the FPR95 scores across the different DKNN variants reveals that the class-specific thresholding variants, when combined with the KNN classifier, yield the most favorable results.

Specifically, the *for_each_species* • (KNN) threshold achieved the lowest error rates of 84.32%.

When used with the *in_each_species* • (KNN) threshold, a decrease of 0.71 percentage points was observed. In contrast, utilizing the MLP classifier or the global thresholding strategy resulted in higher error rates. The *for_each_species* • (MLP) and *over_all_examples* • thresholds exhibited performance drops of 3.93 and 4.71 percentage points.

The highest error rate was observed with the *in_each_species* • threshold using the MLP classifier at 89.41%.

The maximum span between the highest and lowest performing configurations is 5.09 percentage points, emphasizing the impact of the thresholding strategy and classifier choice. A complete list of all FPR95 scores is provided in Table 6.

Summary of Results (Original Dataset) The experimental results demonstrate that all tested DKNN variants consistently outperform random classification. Depending on the specific variant, the AUROC scores exceed the 0.5 baseline by a margin ranging from 0.0963 to 0.1525. The findings are summarized below:

Top Performance: The combination of DKNN with a KNN classifier emerged as the superior configuration. It achieved the highest AUROC of 0.6525 (*for_each_species* •) and the lowest FPR95 of 84.32% (*in_each_species* •).

Weakest Configuration: The *over_all_examples* • threshold strategy yielded the lowest overall performance, with an AUROC of 0.5963. The highest error rate in terms of FPR95 (89.41%) was observed when using the MLP classifier with the *in_each_species* • threshold. This results in a deviation from the top performer of 0,0562 in AUROC and 5.09 percentage points in FPR95.

Threshold Comparison: A direct comparison between the *for_each_species* • • and *in_each_species* • • strategies reveals marginal higher scores for the *for_each_species* threshold. The mean performance gap is amounting to 0.0015 in AUROC and 0.94 percentage points in FPR95.

Classifier Comparison: The KNN classifier consistently outperformed the MLP across all evaluations. The mean performance discrepancy between the two classifiers was 0.0296 for the AUROC and 4.15 percentage points for the FPR95.

5.6.2 Resized Dataset

The following section presents the performance results of the classifiers trained and tested on the resized dataset.

ROC Curves The ROC curves illustrated in Figure 23 are derived from 100 calculated TPR values from the OOD test set (Test ND) and 100 FPR values from the ID test set (Test FG). The discrete data points are marked on the curves, reflecting the DKNN classifier's performance across various thresholds.

All curves exhibit a similar characteristic: They lie above the diagonal, approaching the ideal ROC curve. This demonstrates that the DKNN classifier consistently achieves a higher TPR than FPR across all threshold variants.

> **Impact of Thresholds on ROC Curves** An analysis of the ROC curves for the different thresholding variants reveals that the class-specific thresholds, *for_each_species* • • and *in_each_species* • •, exhibit the highest proximity to the ideal ROC curve (the upper-left corner). Among these, *for_each_species* aligns more closely with the optimal curve than *in_each_species*. In contrast, the *over_all_examples* • threshold demonstrates the greatest deviation from the ideal curve.
>
> **Impact of Classifier on ROC Curves** The choice of classifier for the class-specific *for_each_species* and *in_each_species* methods also results in visible performance gaps. Specifically, the ROC curve using the KNN classifier • • exhibits a significantly more pronounced curvature toward the ideal ROC curve than the one produced by the MLP classifier • •.

AUROC Scores Regarding the AUROC scores, the class-specific threshold *for_each_species* • combined with the KNN classifier achieved the highest AUROC scores of 0.6209.

When combined with the *in_each_species* • (KNN) threshold, a decrease of 0.0037 was observed. In contrast, utilizing the MLP classifier resulted in a greater performance decrease of 0.0295 • and 0.0398 •. The *over_all_examples* • approach yields the lowest AUROC score of 0.5681.

The span of 0,0528 between the highest and lowest scores highlights the varying efficacy of the different thresholding strategies. A complete overview of all AUROC scores is provided in Table 6.

FPR95 Scores Regarding the FPR95 scores the class-specific *for_each_species* • thresholding, when combined with the KNN classifier, yield the lowest error rates of 87.62%. This was followed by the *in_each_species* • (KNN) threshold, which showed a decrease of 0.31 percentage points.

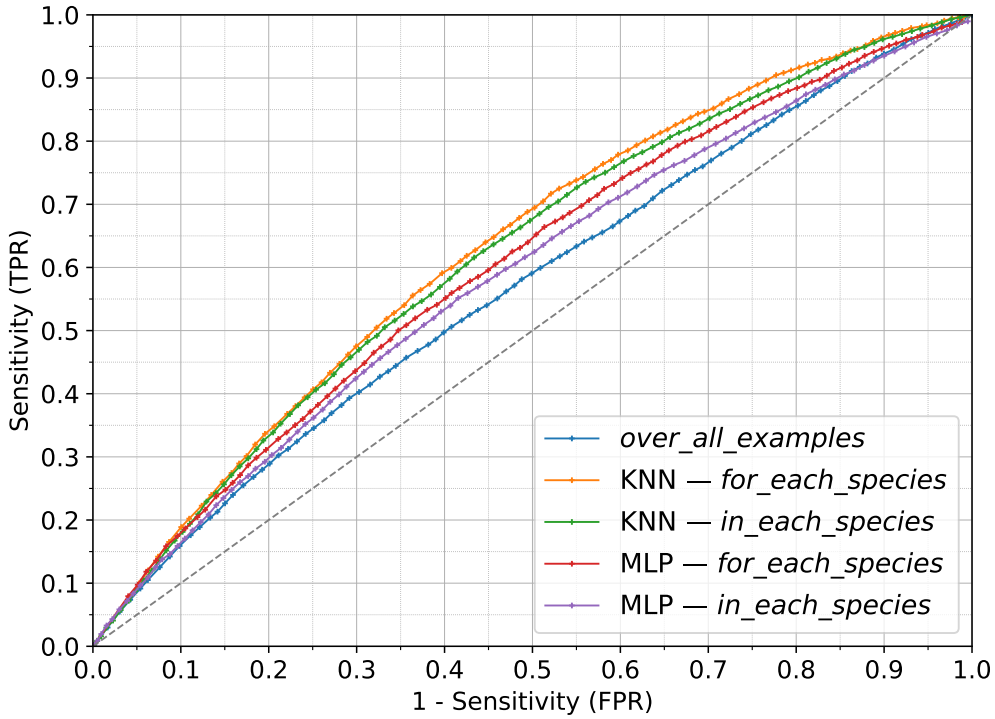


Figure 23: ROC curves for the resized dataset using the DKNN method. The plot compares the global *over_all_examples* threshold with the class-specific thresholds (*for_each_species* and *in_each_species*), with the latter two being implemented using both KNN and MLP classifiers. The diagonal dashed line represents the baseline for random classification.

In contrast, configurations utilizing the MLP classifier or the global thresholding strategy resulted in higher error rates. The *for_each_species* • (MLP) and *over_all_examples* • thresholds results in a performance decrease of 2.76 and 4.07 percentage points.

The highest error rate was observed with the *in_each_species* • (MLP) configuration at 92.45%.

The span between the highest and lowest performing configurations is 4.83 percentage points, emphasizing the impact of the thresholding strategy and classifier choice. A complete list of all FPR95 scores is provided in Table 6.

Summary of Results (Resized Dataset) The experimental results demonstrate that all tested DKNN variants consistently outperform random classification. Depending on the specific variant, the AUROC scores exceed the 0.5 baseline by a margin ranging from 0.0681 to 0.1209. The findings are summarized below:

	Threshold	Dataset	AUROC ↑	FPR95 ↓
•	*over_all_examples*	original	0.5963	89.03%
•	KNN – *for_each_species*	original	0.6523	**84.32%**
•	KNN – *in_each_species*	original	**0.6525**	85.03%
•	MLP – *for_each_species*	original	0.6243	88.25%
•	MLP – *in_each_species*	original	0.6212	89.41%
•	*over_all_examples*	resized	0.5681	91.69%
•	KNN – *for_each_species*	resized	**0.6209**	**87.62%**
•	KNN – *in_each_species*	resized	0.6172	87.93%
•	MLP – *for_each_species*	resized	0.5914	90.38%
•	MLP – *in_each_species*	resized	0.5811	92.45%

Table 6: Novelty Detection – OOD detection performance. Comparison of AUROC and FPR95 scores for various threshold variants across original and resized datasets. Bold values indicate the best performance for each metric within the respective dataset sections.

Top Performance: The DKNN configuration utilizing the *for_each_species* • threshold and the KNN classifier emerged as the superior variant. It achieved the highest AUROC of 0.6209 and the lowest FPR95 of 87.62%.

Weakest Configuration: The *over_all_examples* • strategy yielded the lowest overall performance with an AUROC of 0.5681. The highest error rate in terms of FPR95 (92.45%) was observed when using the MLP classifier with the *in_each_species* • threshold. This results in a deviation from the top performer of 0,0528 in AUROC and 4.83 percentage points in FPR95.

Threshold Comparison: A direct comparison between the *for_each_species* • • and *in_each_species* • • thresholds reveals marginally higher scores for the former. However, the average performance gap remains narrow, with a difference of 0.0119 in AUROC and 1.19 percentage points in FPR95.

Classifier Comparison: The KNN classifier • • consistently outperformed the MLP • • across all evaluations. The mean performance discrepancy between the two classifiers was 0.0328 for the AUROC and 3.64 percentage points for the FPR95.

5.6.3 Comparison of Datasets

The following section examines the impact of the resized dataset on OOD classification performance in comparison to the original dataset. The comparison reveals that

the resized dataset consistently leads to lower OOD detection performance across all evaluated metrics.

Impact on ROC Curves A comparison of the ROC curve progressions for the original and resized datasets (see Figures 22 and 23) reveals similar general characteristics. However, a notable distinction is that the original dataset consistently exhibits a more pronounced curvature toward the ideal ROC curve (the upper-left corner) across all variants compared to the resized dataset.

> **Impact of Thresholds on ROC Curves** The progression of the ROC curves for individual thresholds is visibly influenced by the choice of dataset. Specifically, the curves for *for_each_species* • • and *in_each_species* • • differ more significantly when using the resized dataset. In this case, the effect for the *for_each_species* threshold exhibits a noticeably closer approximation to the ideal ROC curve than the *in_each_species* threshold. In contrast, this discrepancy in progression is minimal or nearly imperceptible when using the original dataset.
>
> **Impact of Classifiers on ROC Curves** The discrepancy in performance between the KNN and MLP classifiers is consistent in both datasets, thus indicating no observed effect.

Impact on AUROC Scores Comparing the datasets regarding their AUROC scores reveals that the original dataset consistently yields higher values than the resized dataset across all thresholding variants.

The original dataset achieves a peak AUROC score of 0.6525, whereas the resized version reaches 0.6209. This represents a discrepancy of 0,0316.

The spread between the minimum and maximum AUROC scores for the different threshold variants is 0,0562 for the original dataset and 0,0528 for the resized version. This indicates that the original dataset exhibits marginal higher variance in detection performance across the different thresholds variants.

In summary, the original dataset yields AUROC scores that are 0,0316 higher than those of the resized dataset. This indicates an opposite effect to what was expected for the resized data.

Impact on FPR95 Scores The previously described discrepancy between the original and resized datasets is also reflected in the FPR95 scores. Across all thresholding methods, the original dataset achieves lower (superior) FPR95 scores compared to the resized dataset.

The original dataset achieves its lowest FPR95 of 84.32%, while the resized dataset reaches a minimum of 87.62%. This represents a discrepancy of 3.3 percentage points.

The spread between the minimum and maximum FPR95 scores for the different threshold variants amounts to 5.09 percentage points for the original dataset and 4.83 percentage points for the resized dataset. This indicates that the resized dataset exhibits a lower variance in detection performance across the different thresholding variants.

5.7 Domain Generalisation & Novelty Detection

In a final experiment, two challenges of automated moth monitoring are combined: domain generalization and novelty detection. The first, domain generalization, addresses the classification challenge of generalizing from high-resolution training data to low-resolution images captured by moth traps. The second, novelty detection, tackles the challenge of shifting distribution range of insects, requiring the system to distinguish between regionally known species (ID) and unknown species (OOD).

To address both challenges, the previously employed DKNN approach based on embeddings was evaluated for its ability to differentiate between ID and OOD moth species. Consistent with previous experiments, the method was applied to both the original and resized datasets. The test datasets employed for this experiment were Test FG + DG (ID) with 91 classes and the Test ND + DG (OOD) with 37 classes. Unlike in previous experiments, these datasets consist of low-resolution trap images.

The objective is to assess the robustness of the OOD classifiers against a covariate shift caused by the discrepancy between high-resolution training data and low-resolution test data. As previously outlined, the resized dataset was employed to mitigate information loss during the extraction of embeddings from rectangular images, with the objective of enhancing classification performance.

For each DKNN variant, ROC curves were plotted, and both AUROC and FPR95 scores were computed. Performance was evaluated and compared with respect to the dataset types (original and resized), the threshold variants, and the classifiers utilised for the class-specific thresholds (MLP and KNN).

5.7.1 Original Dataset

The following section presents the performance results of the classifiers trained and tested on the original dataset.

ROC Curves The ROC curves illustrated in Figure 24 are derived from 100 calculated TPR values from the OOD dataset (Test ND + DG) and 100 calculated FPR values from the ID dataset (Test FG + DG). The discrete data points are marked with crosses on the plotted curves, representing the DKNN classifier's performance across various thresholds.

All ROC curves follow the diagonal line, rising only slightly above the random classification threshold toward the end of the curve. Consequently the curve trajectories do not achieve the typical convex characteristics of an ideal classifier.

Furthermore, the middle and lower-left regions are characterized by a sparse distribution of data points with larger intervals, concentrated with a high density of measurement points towards the upper-right region of the plot.

Impact of Thresholds and Classifiers on ROC Curves An analysis of the ROC curves for the various threshold strategies proves difficult. The sparse measurement points at the beginning of the curve and the high concentration of points toward the end result in partly jumps in the trajectory. These patterns are consistently visible across all tested DKNN variants, with only minor, short-lived deviations between them. Consequently, a differentiation between the individual thresholding variants and the classifiers used is hardly feasible in this context.

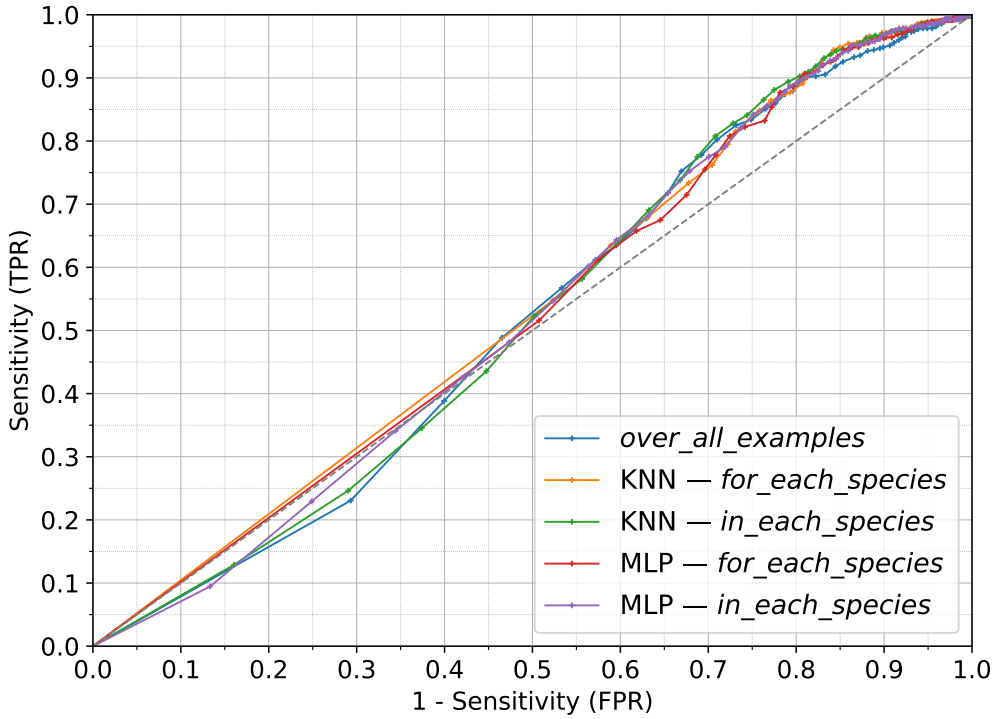


Figure 24: ROC curves for the original dataset using the DKNN classifier. The plot illustrates the performance of various threshold variants – *over_all_examples*, *for_each_species*, and *in_each_species* – implemented with both KNN and MLP classifier. The diagonal dashed line represents the baseline for random classification.

AUROC Scores Regarding the AUROC scores, the class-specific *for_each_species*
- threshold paired with KNN classifier achieve the highest AUROC scores of 0.5350.

Intermediate results followed by a difference of 0,0068 and 0,0117 combined with *for_each_species* • (MLP) and *in_each_species* • (MLP) thresholds, and by a discrepancy of 0,0149 with the *in_each_species* threshold • (KNN).

The *over_all_examples* • threshold yields the lowest performance with a score of 0.5169.

The 0.0181 span between the highest and lowest scores indicates only marginal effects of different thresholding strategies. A comprehensive summary of all AUROC scores is provided in Table 7.

FPR95 Scores Regarding the FPR95 scores the class-specific *for_each_species* threshold •, when paired with the KNN classifier, achieves the lowest FPR95 error rates of 85.21%.

Intermediate results followed by the *in_each_species* threshold, which yields a difference of 0.99 percentage points • (MLP) and 1.17 percentage points • (KNN). Using the *for_each_species* threshold the difference amount to 2.05 percentage points • (MLP).

The highest error rate is produced by the *over_all_examples* • threshold at 90.31%.

Within the FPR95 metric, the discrepancy between the minimum and maximum scores spans a range of 5.1 percentage points. A complete list of all FPR95 scores is provided in Table 7.

Summary of Results (Original Dataset) Overall, the tested DKNN variants exhibit minimal variation in their ROC curves. The trajectories primarily follow the diagonal line, rising only slightly above the random classification threshold toward the end of the curve. This indicates that all variants perform only marginally better than a random classifier, with a maximum AUROC improvement of 0.0350 over the 0.5 baseline. The findings are summarized below:

Top Performance: The combination of the *for_each_species* • threshold and the KNN classifier achieved the highest AUROC of 0.5350 and the lowest (favorable) FPR95 of 85.21%.

Weakest Configuration: The *over_all_examples* • threshold strategy yielded the lowest overall performance, with an AUROC of 0.5169 and the highest FPR95 error rate of 90.31%. This results in a deviation from the top performer of 0.0181 in AUROC and 5.10 percentage points in FPR95.

Threshold Comparison: A direct comparison between the class-specific variants reveals marginally higher AUROC scores for the *for_each_species* • • threshold, with an average lead of 0.0099 over *in_each_species* • •. Regarding the FPR95, no clear advantage for either threshold was observed.

Classifier Comparison: No distinct difference in performance between the KNN and MLP classifiers is evident for this dataset, although the KNN classifier achieved slightly higher peak scores.

5.7.2 Resized Dataset

The following section presents the performance results of the classifiers trained and tested on the resized dataset.

ROC Curves The ROC curves illustrated in Figure 25 are derived from 100 calculated TPR values from the OOD dataset (Test ND + DG) and 100 calculated FPR values from the ID dataset (Test FG + DG). These discrete data points are marked with crosses on the curves, representing the DKNN classifier's performance across various thresholds.

All ROC curves exhibit slightly wave-like curve characteristics along the diagonal. In the lower threshold range, the curves initially fall below the line of random performance. In the mid-range, the trend reverses and exceeds the random classification threshold. A subsequent increase above this threshold is observed as the curve approaches its end. Consequently, the trajectories lack the typical convex curvature associated with an ideal classifier.

Furthermore, the middle and lower-left regions are characterized by a sparse distribution of data points with larger intervals, concentrated with a high density of measurement points towards the upper-right region of the plot.

Impact of Thresholds and Classifiers on ROC Curves An examination of the ROC curves for the various threshold strategies proves difficult. The sparse measurement points at the beginning of the curve and the high concentration of points toward the end result in partly jumps in the trajectory. These patterns are consistently visible across all tested DKNN variants, with only minor, short-lived deviations between them. Consequently, a differentiation between the individual thresholding strategies and the classifiers used is hardly feasible in this context. However, one difference is more clearly visible. The *over_all_species* • variant stands out somewhat and shows a more even wave shape than the class-specific thresholds.

AUROC Scores Regarding the AUROC scores, the class-specific *for_each_species* threshold yields the highest score paired with the MLP classifer of 0.5243 •.

Intermediate results followed by a difference of 0.0015 with *for_each_species* • (KNN) and 0.0078 *in_each_species* • (KNN) thresholds. Using the *in_each_species* threshold combined with a MLP Klassifier, the difference amount to 0.0121 • (MLP).

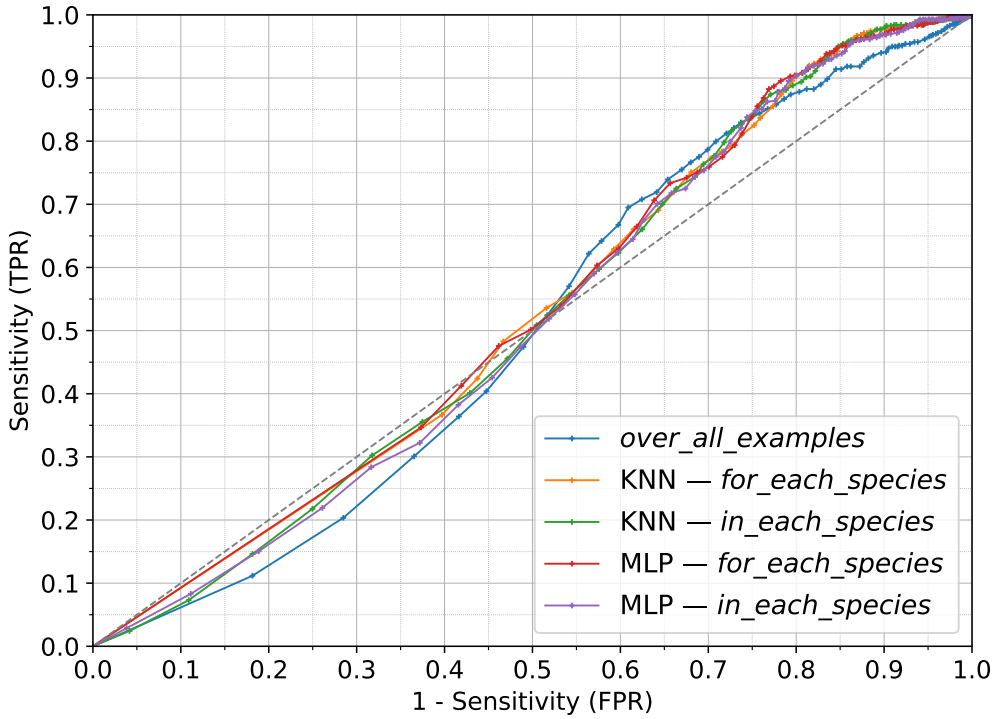


Figure 25: ROC curves for the resized dataset using the DKNN method. The plot compares the global *over_all_examples* threshold with the class-specific thresholds (*for_each_species* and *in_each_species*), with the latter two being implemented using both KNN and MLP classifiers. The diagonal dashed line represents the baseline for random classification.

The lowest results yield the *over_all_examples* • threshold with a score of 0.5025.

Overall, the difference between the highest and lowest variants is 0.0218. A complete overview of all AUROC scores is provided in Table 7.

FPR95 Scores The FPR (ID) at a TPR (OOD) of 0.95 exhibits marginal variations across class-specific variants, with the most significant discrepancy observed between the class-specific and the global thresholding strategies.

The *for_each_species* • threshold, when paired with the KNN classifier, achieves the lowest (favorable) FPR95 value of 84.82% and marginal difference of 0.09 percentage points with the *in_each_species* • (KNN) threshold.

Intermediate results followed by the *for_each_species* • (MLP) threshold, with a difference of 00.22 percentage points and the *in_each_species* • (MLP) threshold with a difference of 1.1 percentage points.

The lowest performance was recorded by the *over_all_examples* • threshold, which lagged behind the other methods by a considerable margin of 6.73 percentage points, yielding an FPR95 score of 91.55%.

This represents a substantial performance gap of 6.73 percentage points compared to the best-performing variant. A complete overview of all FPR95 scores is provided in Table 7.

	Threshold	Dataset	AUROC ↑	FPR95 ↓
•	*over_all_examples*	original	0.5169	90.31%
•	KNN – *for_each_species*	original	**0.5350**	**85.21%**
•	KNN – *in_each_species*	original	0.5201	86.38%
•	MLP – *for_each_species*	original	0.5282	87.26%
•	MLP – *in_each_species*	original	0.5233	86.20%
•	*over_all_examples*	resized	0.5025	91.55%
•	KNN – *for_each_species*	resized	0.5228	84.91%
•	KNN – *in_each_species*	resized	0.5165	**84.82%**
•	MLP – *for_each_species*	resized	**0.5243**	85.04%
•	MLP – *in_each_species*	resized	0.5122	85.92%

Table 7: Domain Generalisation (Novelty Detection) – OOD detection performance. Comparison of AUROC and FPR95 scores for various threshold variants across original and resized datasets. Bold values indicate the best performance for each metric within the respective dataset sections.

Summary of Results (Resized Dataset) The evaluated DKNN variants exhibit minimal variation between in their ROC curves. The trajectories primarily follow the diagonal in a slight wave-like pattern. Towards the upper-right quadrant, the curves rise slightly above the diagonal. Consequently, all variants perform only marginally better than a random classifier, achieving a maximum AUROC improvement of 0.0243 over the 0.5 baseline. The findings are summarized below:

Top Performance: The combination of the *for_each_species* • threshold and the MLP classifier achieved the highest AUROC of 0.5243. Meanwhile, the *in_each_species* • threshold combined with a KNN classifier reached the lowest (favorable) FPR95 of 84.82%.

Weakest Configuration: The *over_all_examples* • threshold strategy yielded the lowest overall performance, with an AUROC of 0.5025 and the highest FPR95 error rate of 91.55%. This results in a deviation from the top performers of 0.0218 in AUROC and 6.73 percentage points in FPR95.

Threshold Comparison: The global *over_all_examples* • threshold consistently achieves the lowest scores across both metrics.

A direct comparison between the class-specific methods reveals marginally higher AUROC scores for the *for_each_species* • • threshold, with an average lead of 0.0092 over *in_each_species* • •. Regarding the FPR95, no clear advantage for either threshold was observed.

Classifier Comparison: No distinct difference in performance between the KNN • • and MLP • • classifiers was found regarding the AUROC, although the MLP classifier achieved slightly higher peak scores. In terms of the FPR95, the KNN reached marginally lower (favorable) scores with an average difference of 0.615 percentage points compared to the KNN.

5.7.3 Comparison of Datasets

The following section examines the impact of the resized dataset on OOD classification performance compared to the original dataset. The comparison reveals no significant difference between the datasets. Overall, both datasets exhibit performance levels close to the random classification baseline, reaching peak AUROC scores of 0.5350 for the original and 0.5243 for the resized dataset. While the original dataset achieves higher AUROC scores, the resized dataset yields lower FPR95 scores.

Impact on ROC Curves The ROC curves for both datasets (see Figures 24 and 25) exhibit wave-like trajectory along the diagonal. This pattern is more pronounced in the resized dataset, which also exhibit more measurement points in the initial segments of the curves compared to the original dataset.

Impact of ROC Curve Thresholds Regarding the thresholding strategies, the *over_all_examples* • is more visually distinct from the class-specific variants in the resized dataset than in the original.

Impact of Classifiers No clear impact of the dataset choice on the performance of the utilized classifiers is distinguishable within the ROC curves. Both classifiers maintain their respective characteristics regardless of whether the original or resized data is used.

Impact on AUROC Comparing the datasets regarding their AUROC scores reveals that the original dataset consistently yields higher values than the resized dataset across all thresholding variants.

The original dataset achieves a peak AUROC score of 0.5350, whereas the resized version reaches 0.5243. This represents a discrepancy of only 0.0107.

The spread between the minimum and maximum AUROC scores for the different threshold variants is 0.0181 for the original dataset and 0.0218 for the resized

dataset. This indicates that the resized dataset exhibits higher variance in detection performance across the different thresholds.

In conclusion, both datasets demonstrate AUROC scores of marginally higher than 0.5, thus indicating only a slight improvement in performance compared to random detection.

Impact on FPR95 Scores Comparing the datasets with respect to their FPR95 scores reveals that the resized dataset yields lower error rates for nearly all thresholding variants, with the exception of the *over_all_species* • variante.

The resized dataset achieves its lowest FPR95 of 84.82%, while the original dataset reaches a minimum of 85.21%. This represents a difference of 0.39 percentage points.

The spread between the minimum and maximum FPR95 scores for the different threshold variants amounts to 6.73 percentage points for the resized dataset and 5.10 percentage points for the original dataset. Similar to the AUROC results, this indicates that the resized dataset exhibits a higher variance in detection performance across the different threshold variants.

6 Discussion and Outlook

This chapter contextualizes the findings and provides recommendations for future research in the field of automated moth classification.

6.1 AMI Dataset Availability

One objective of this work was to verify the continued availability of the AMI dataset, following its publication as a benchmark dataset. Unlike other popular datasets such as ImageNetK1, the AMI dataset relies on a decentralized distribution where individual image files must be downloaded from various original providers. This specifically applies to the GBIF Fine-Grained and Binary subsets, which were aggregated from multiple databases via the GBIF platform. Because the images are accessed individually via third-party URLs, these subsets are susceptible to data incompleteness caused by link rot or other technical issues on the providers' side.

This assumption was partially confirmed, as 51,169 images from the AMI-GBIF subsets could not be successfully retrieved. The primary reasons for these failures were broken URLs or failed server connections. Specifically, 49,806 downloads failed due to non-existent URLs (404 Client Error), and 1,069 failed due to connection timeouts with the data provider's server. Additionally, some image files were found to be corrupted. A comprehensive list of all failed downloads and associated error codes is provided in Appendix A.1.

Notwithstanding these 51,169 failures, the retrieval rate remains high. Approximately seven months after the release of the second AMI dataset version (June 26, 2024), 98.3% of the AMI-GBIF Fine-Grained subset and 98.9% of the AMI-GBIF Binary subset were successfully downloaded.

Given that the data loss is minimal (at most 1.7%) and evenly distributed across all data splits of the cross-regional dataset (all regions), this thesis concludes that the dataset remains highly suitable for the development and testing of moth trap applications.

When evaluated as a benchmark, the loss of image data inherently compromises comparability, even if the current percentage is marginal. It is anticipated that this attrition rate will rise over time. In order to undertake rigorous benchmarking, it is advisable to document the precise retrieval date and to specify which images are no longer available. This approach is intended to ensure a more transparent assessment of potential limitations regarding the comparability of results.

6.1.1 Outlook

The current assessment of the AMI dataset's availability as a benchmark has primarily focused on the distribution of data splits across all regions. Consequently, it would be valuable to also investigate the availability of individual sub-regions within the dataset.

Additionally, an evaluation of data providers would be conceivable to identify those with particularly high failure rates. In such cases, removing or filtering these providers from the dataset could contribute to more stable availability and, thus, enhance long-term comparability.

These recommendations are already feasible using the download results provided as CSV files[54], which documents the success or failure of every individual image download.

6.2 Generated Datasets

To address the research questions regarding fine-grained classification, domain generalization (fine-grained), novelty detection, and domain generalization (novelty detection), six subsets were extracted from the AMI dataset. The creation of these subsets reduced the overall data volume, thereby lowering the computational requirements. Simultaneously, this selection was intended to provide an initial assessment of the performance of the investigated classification methods within the context of the AMI dataset.

6.2.1 Generalizability

Generalizing the performance results of the classification methods to the entire AMI dataset is highly constrained due to the dataset splitting and the associated data reduction.

The datasets were limited to the species level, and a minimum/maximum number of samples per class was established. These restrictions led to a significant decrease in the number of classes: the GBIF Fine-Grained subset was reduced from 5,364 to 128 species, and the Trap Fine-Grained subset from 516 to 128 species. Consequently, the long-tail distribution characteristics typical of the AMI dataset were removed.

Although the inherent complexity of insect classification represented by 91 fine-grained (ID) and 37 OOD classes, remains. It must also be assumed that applying these methods to the full dataset would present different challenges. This is due to the substantially larger data volume, the increased complexity resulting from a larger number of classes, and more pronounced class imbalance inherent in the complete AMI dataset.

6.2.2 Anomalies in Image Distribution

The following section describes observations regarding the image data. These observations were identified as noteworthy during the conduct of this work.

[54]See the following folder for AMI-GBIF Binary and AMI-GBIF Fine-Grained CSV Files:
/data/datasets/ami/binary/download_check
/data/datasets/ami/fine-grain/download_check

Observation Redundancy As introduced in Chapter 3, AMI-GBIF datasets are characterized by the fact, that a single insect observation may contain multiple images. To prevent bias (data leakage), the AMI datasets ensure, that all images from one observation are assigned to the same data split. Therefore, it is noteworthy that all created datasets may contain multiple high-resolution images of an insect observation, but only within a single datasplit.

The property that multiple images of the same moth can be included in the AMI-GBIF dataset, was also observed during work on the AMI-Trap dataset. In contrast to the GBIF observations, the trap data revealed a high frequency of images within a single class that were strikingly similar or nearly identical. For instance, images of the species *Automeris io*[55] suggest that the same individual was likely photographed multiple times (see Figure 26), resulting in significant data redundancy.

A plausible explanation for this observation lies in the triggering mechanisms of the moth traps. According to Jain et al., 2024, the traps are activated by two distinct methods: a fixed ten-minute interval and motion detection. Given that the AMI dataset comprises 2,893 full-frame images from 22 monitoring stations, it is highly probable that the same specimen was captured repeatedly, leading to the inclusion of highly redundant samples in the final AMI-Trap dataset (Jain et al., 2024, pp. 6-7).

In conclusion, multiple captures of the same specimen can occur within the low-resolution Test DG and Test ND + DG datasets, leading to high intra-class redundancy – as exemplified by the species *Automeris io*. While this phenomenon is also present in the high-resolution datasets (Training, Validation, Test FG, and Test ND), it was observed to a much lesser extent. This redundancy in individual species can distort the results of the classifiers and should be taken into account when interpreting the results. It also makes comparison with other data sets more difficult.

Identical Image Leakage The GBIF dataset also revealed embeddings that exhibited extremely high cosine similarity during KNN classification. The corresponding ten images, shown in Figure 27, possess identical content. This confirms that identical images were present in both the training and Test FG datasets (data leakage). While this circumstance distorts the classification results, it can be assumed that these are isolated cases.

6.2.3 Outlook

Looking ahead, it is recommended to conduct a systematic audit of the AMI dataset for duplicate images. This is prompted by the observed redundancies within the *Automeris io* species in the Trap dataset and similar issues found in the GBIF data.

[55]The species *Automeris io* corresponds to label 6 in the dataset created for this study and species key 1866085 in the AMI Traps dataset.

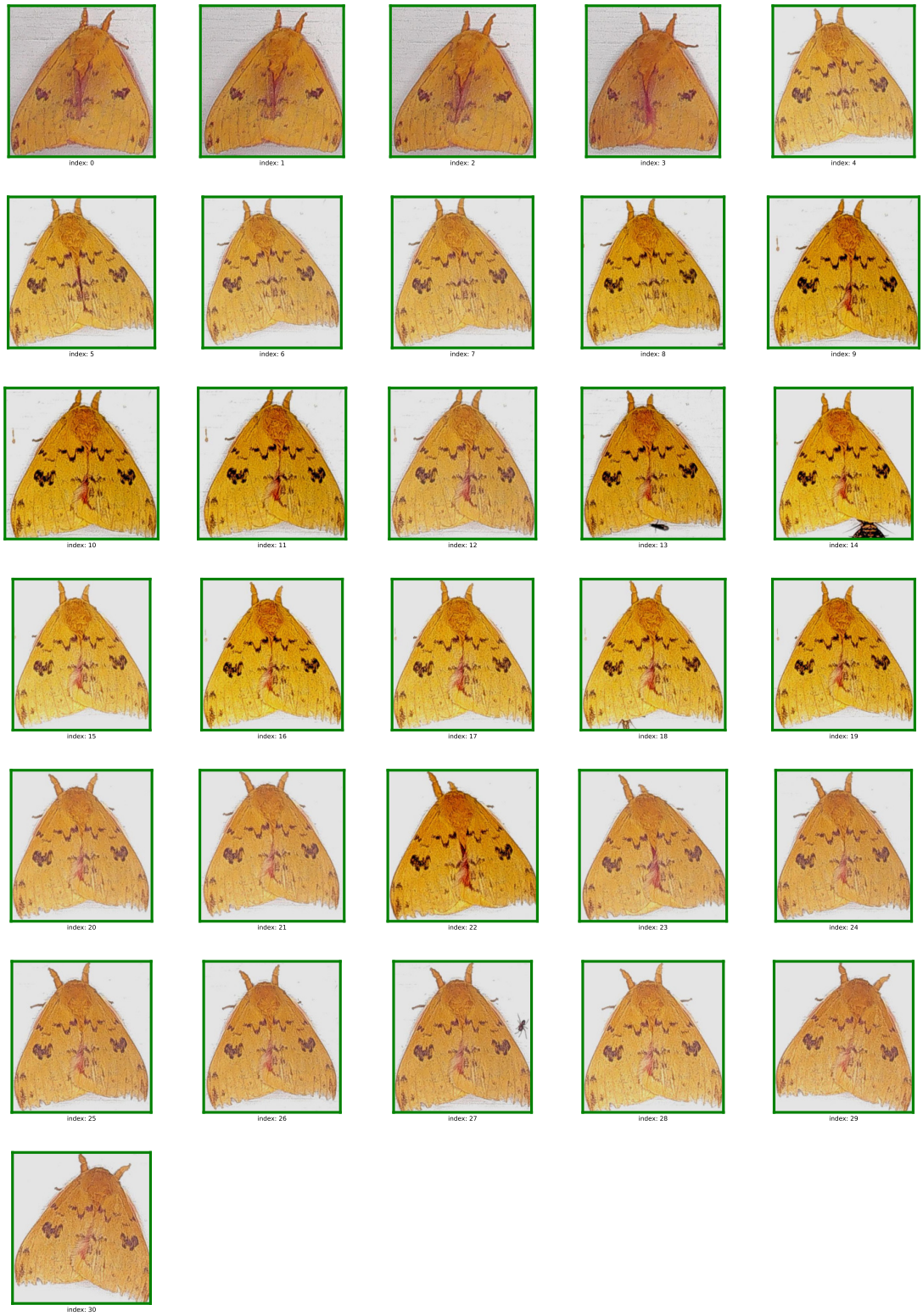


Figure 26: Here you can see all examples of the species Automeris io from the test dataset test FG + DG.

Figure 27: Here you can see one image from the Test FG dataset (2570233081.jpg – bottom right) and images from Training dataset, which stood out during classification due to their small distance.

Implementing a similarity-based filtering process would allow for the identification of identical or near-identical images.

The embeddings and the KNN classification approach developed in this study could be utilized for this purpose by identifying similar images via angular distance metrics (such as the cosine similarity). However, caution is required to ensure that distinct individuals of the same species are not misidentified as duplicates[56].

6.3 Square Padding (Resized Dataset)

The experiment investigating the three padding methods aimed at mitigating information loss during preprocessing is not generalizable due to its qualitative nature and the limited number of specimens. Furthermore, this experiment utilizes local patch embeddings, as opposed to the global embeddings employed in the primary classification methods. Consequently, this study represents a preliminary ad-hoc experiment intended to provide initial insights into the different padding techniques that were tested. The development of the resized datasets was founded upon this, with the subsequent quantitative analysis of the applied padding strategy.

[56]Ideally, this deduplication process should be supervised by taxonomic experts in moth classification.

6.3.1 Outlook

Beyond square padding, additional image processing strategies were considered that appear promising for automated moth monitoring applications. Since moth traps capture insects against a monochromatic background (trap screen), this factor could be leveraged during classifier training. By utilizing the object detector from the experiment - which is based on the first principal component (PCA) of the local patch embeddings - it is possible to accurately segment the primary object from the background.

This would allow for the background to be substituted with the specific color of the trap screen, as demonstrated in Figure 28. In this illustration, the background is set to a colour corresponding to the image's mean color, similar to the Mean Color padding method. The ultimate goal would be to calibrate this colour to precisely match the physical recording screens used in moth traps.

Figure 28: Process and result of resizing the low image on example

This would yield a training dataset that is significantly more aligned with the test dataset. Specifically, the environmental context – which is irrelevant for classification – would be eliminated from the training data. The primary challenge lies in object detection, as moths are not always centrally positioned, as seen in Figure 28. Insects may appear at the edges of the frame or blend into their environment due to camouflage, significantly complicating or even preventing reliable detection.

6.4 Fine-Grained Classification

Fine-Grained classification represents an important step in automated monitoring. To this end, two classifiers (KNN and MLP) were implemented and evaluated using embeddings from the DINOv2 (Small) foundation model for both the original and resized datasets.

6.4.1 Evaluation of Results

The KNN and MLP classifiers achieved Balanced Accuracies of 0.86 and 0.94, respectively, indicating high recognition rates for both approaches.

Comparing these findings with published results for the AMI dataset, similar recognition rates were achieved using the Micro Accuracy score Top 1. However, Micro

Accuracy is more sensitive to imbalanced test sets with high sample counts in individual classes than Balanced Accuracy. Furthermore, the published results were determined on regional subsets of the AMI dataset, which introduces variability in class counts and long-tail characteristics.

Using a ConvNeXt-B model, researchers achieved Micro Accuracy scores of 91.66% for the NE-America region includes 277 species[57], 92.48% for W-Europe includes 244 species, and 96.22% for C-America includes 6 species (Jain et al., 2024, pp. 12-13). Consequently, the classification methods employed in this work demonstrate recognition rates comparable to other approaches in this field. Nevertheless, it should be noted that this comparison is limited, as used metrics and the underlying datasets differ.

Embeddings The results demonstrate that DINOv2 embeddings effectively capture the subtle morphological differences between the 97 moth classes in high-resolution images. This highly differentiated representation is the prerequisite for the high performance achieved by the downstream classifiers.

Classifiers The findings indicate that the MLP is capable of achieving a higher degree of class differentiation within the feature vectors when compared to the KNN method. The MLP is capable of highlighting specific dimensions of the 384-dimensional vector that are support for species differentiation through its weighted connections, whereas KNN treats all dimensions equally. The 8 percentage point performance disparity between KNN and the MLP classifier is therefore anticipated. As indicated by the findings of the original DINOv2 benchmarks on ImageNet-1k, analogous trends were observed, with a 3 percentage point discrepancy reported between KNN and a linear classifier (Oquab et al., 2024, p. 11).

Resized Dataset Comparing the classification performance of the resized dataset with the original dataset reveals that the resized version achieves lower overall accuracy. The theoretical gain in information – resulting from the ability to capture the entire moth during embedding extraction via padding – did not lead to an improvement. Instead, this potential advantage was likely negated by adverse side effects.

Specifically, Balanced Accuracy decreased by approximately 2 percentage points for the MLP classifier and by 4.5 percentage points for the KNN classifier. A possible explanation lies in the nature of the generated padding. The padded regions may introduce unusual spatial relationships between the moth's body parts and the overall frame, which can distort the embedding extraction process. Consequently, this leads to higher variance among embeddings within the same class (intra-class variance), ultimately hindering classification performance.

[57] Includes 277 species in training (GBIF) and test (Trap) set

6.4.2 MLP Anomalies

Unexpected behavior was observed during the training of the MLP classifier on the resized dataset. The model exhibited a Balanced Accuracy on the unseen validation set that was approximately 5 percentage points higher than on the training set. This contradicts the typical expectation that models perform better on their training data. For a more detailed analysis of this phenomenon, please refer to Section 4.2.1 in the methods section.

6.4.3 Outlook

Looking ahead, the high Balanced Accuracy values achieved by both classification methods (MLP and KNN) justify scaling these approaches to larger portions of the AMI dataset and broader domains of insect classification.

A logical next step involves the utilization of regional AMI subsets. These typically encompass a larger number of classes and introduce additional complexities, such as long-tail distributions. Consequently, these datasets impose higher demands on the classification models.

Furthermore, regional datasets provide the opportunity to benchmark these results directly against those published in the AMI publication, allowing for a more precise assessment of the performance of the proposed methods within the current state of research.

Potential for improvement also lies in scaling to larger DINOv2 models. Since this study utilized the DINOv2-Small variant, it can be assumed that larger model sizes – specifically DINOv2-Giant – would yield further performance gains. On the iNat2018 and iNat2021 benchmarks, the Giant model achieves a Top-1 accuracy that is up to 12.6 and 11.5 percentage points higher than the Small variant used here (Oquab et al., 2024, p. 13).

Similar advancements are expected from the latest DINOv3 models. Released in August 2025, the DINOv3 further enhanced visual classification performance. Consequently, future research should evaluate the integration of DINOv3 into the existing classification framework to leverage these improvements (Siméoni et al., 2025, p. 24-25).

The Lepmon project provides further insight into potential improvements to classification. It suggests that, while classification is currently conducted at the pixel level, this should not be the only criterion. Instead, domain and expert knowledge should be integrated to incorporate additional morphological features, thereby enhancing overall recognition performance. For example, moth species can be identified more accurately based on their resting wing silhouettes: Some species exhibit a V-shaped silhouette, while others resemble an inverted letter A. Furthermore, behavioural characteristics, such as whether a species typically spreads its wings or keeps them folded while at rest, serve as vital distinctive features. Additionally, the physical size of the specimen can be taken into consideration and determined precisely via

a reference scale on the trap's recording screen. All this information could be incorporated into classification models to improve their performance. (Analyse des Wandels der Biodiversität (LIB), 2025)

6.5 Domain Generalization

Beyond the large number of moth species, another significant challenge in moth monitoring lies in the discrepancy of visual representation between high-resolution training data and the low-resolution test data captured by moth traps (i.e., the domain shift). To address this, the previously (Fine-Grained Classification) employed KNN and MLP classification methods were evaluated for their domain generalization performance on both the original and resized datasets.

6.5.1 Evaluation of Results

The classification methods achieved a Balanced Accuracy of 43.78% using the MLP classifier and 35.54% with the KNN classifier.

When comparing these findings with results reported for the AMI dataset, the latter show significantly higher recognition rates. Furthermore, the datasets utilised in the publication feature a larger number of classes and exhibit long-tail characteristics. For instance, in the NE-America region[58], classifiers using a ConvNeXt-B model achieved Balanced Accuracy scores of 68.56%. Similarly, for the W-Europe region[59], a score of 76.41% was reported (Jain et al., 2024, pp. 12-13).

In this context, the classification performance under domain shifts remains limited.

Embeddings The results suggest that embeddings derived from low-resolution images exhibit limited similarity to their high-resolution counterparts. Consequently, when projected into the latent space together, they lack the compactness required for robust classification.

Classifiers The same reason apply here as previously described for the fine-grained classifier as shown in Section 6.4.1.

Resized Dataset Comparing the classification results between the original and resized datasets reveals that the resized dataset generally achieved higher performance, with the sole exception of the Macro Precision score when using the MLP classifier. It appears that preserving the entire moths in images, and thus the entire image information, allowed for higher-quality embedding, which in turn slightly improved the performance of the classifier.

[58]Contains 2,497 species, of which 277 were included in the trap test set
[59]Contains 2,603 species, of which 244 were included in the trap test set

Specifically, Balanced Accuracy increased by a maximum of approximately 2 percentage points for the KNN classifier and by 0.89 percentage points for the MLP.

These results contrast with the previous fine-grained experiment on the GBIF dataset, likely due to the different characteristics of the test datasets. Unlike the high-resolution GBIF images, which often picture moths within a complex context (e.g., museum labels or vegetation), the trap dataset primarily consists of rectangular images without context. The images captured by the moth traps on monochrome trap screens were processed through an object detection pipeline that precisely cropped the images around the moths. Consequently, these images are characterised by an emphasis on the object and a monochromatic background. Specifically, the unchanged original images are subject to more extensive cropping, which in turn results in the loss of valuable information. This effect is potentially more significant in the trap test dataset due to its already tightly cropped images than in the GBIF dataset, which provides a broader context in the images. This provide a explanation for the contrasting performance observed when comparing the resized and original dataset.

6.5.2 Outlook

In conclusion, the moderate balanced accuracy values achieved by both classification methods (KNN and MLP) do not justify scaling the current approach to larger portions of the AMI dataset at this stage. Instead, larger models should be used to evaluate whether a significant improvement in classification performance can be achieved.

Given that this study utilized the DINOv2 (Small) model, it is highly probable that larger variants – specifically DINOv2 Giant – would demonstrate superior domain generalization due to higher-quality embedding representations. In domain generalization benchmarks such as ImageNet-A, -R, -C, and Sketch, which were evaluated using linear probing, performance gaps of up to 42.4% were observed between the Small and Giant Model. This highlights a substantial potential for increasing classification accuracy by employing larger model variants.

Similarly, DINOv3 – released in August 2025 – presents a promising opportunity to improve classification performance over DINOv2 (Siméoni et al., 2025, pp. 23–25).

Another potential avenue involves transitioning the classification methods toward Domain Adaptation. Unlike domain generalization, this approach incorporates data from the target domain into the training phase. This could be implemented by downscaling high-resolution images from the training set to simulate low-resolution trap data – a technique already explored in the original AMI dataset publication (Jain et al., 2024, pp. 9–15). Consequently, the foundation model would no longer be solely responsible for bridging the resolution gap via its embeddings; instead, it could leverage an augmented training set that includes synthetically generated low-resolution samples.

6.6 Novelty Detection

A third challenge in moth monitoring arises from shifting population ranges, necessitating the differentiation between regionally known ID species and unkown OOD (novel or invasive) species. To address this, a DKNN classification method utilizing three distinct threshold variants was implemented and evaluated for both the original and resized datasets.

6.6.1 Evaluation of Results

The top-performing variants of the DKNN achieved a peak AUROC score of 0.6525 and a minimum FPR95 value of 84.32% (where lower values indicate better performance).

Contextualizing these results remains challenging due to the lack of comparable experiments in the literature. This observation aligns with (Chen and Rolnick, 2023, p. 2), who conducted OOD classification on a specific moth family[60] and noted the general absence of OOD detection studies in the field of fine-grained wildlife classification.

Even without a direct comparison to other publications, it is evident that an AUROC score of 0.6525 represents only a modest improvement over the random baseline of 0.5. Similarly, an FPR95 value of 84.32% indicates that while 95% of the OOD samples are correctly identified, 84.32% of the ID samples are still misclassified. This confirms that the model's ability to distinguish between ID and OOD classes remains highly limited.

Embeddings The limited differentiation between ID and OOD samples by the DKNN classifier suggests that the DINOv2 embeddings (in 384 dimensions) do not form sufficiently compact representations for individual moth species. Consequently, a reliable density-based separation between these species is not feasible within the created embeddings.

Threshold Among the threshold methods used, the class-specific *for_each_species* and *in_each_species* approaches achieved the highest scores. These offer an advantage over the global *for_all_species* threshold by allowing for individual calibration per class, thus enabling more precise boundaries. However, these thresholds depend on the preceding fine-grained classification stage.

Resized Dataset The resized dataset results in lower performance, with a decrease up to 0.0401 in the AUROC score and up to 3.3% in FPR95 compared to the original dataset. The same assumptions apply here as previously discussed for the fine-grained classifier as shown in Section 6.4.1.

[60] This represents a different taxonomic level compared to the present study.

6.6.2 Outlook

Extending this method to larger portions of the AMI dataset is currently not recommended due to the limited classification performance.

As noted in the outlook for fine-grained classification and domain generalization, utilizing larger models (DINOv2-Large or -Giant) or more recent models (DINOv3) could significantly enhance recognition rates. For instance, while DINOv2-Small uses 384-dimensional feature vectors, DINOv2-Giant produces vectors of length 1536. This increased dimensionality may allow for a more nuanced representation of visual characteristics across different species, may leading to better separability in the latent space. (Oquab et al., 2024, p. 13; Siméoni et al., 2025, p. 24–25)

6.7 Domain Generalization & Novelty Detection

In the final experiment, the two main challenges of automated moth monitoring are addressed together: Domain Generalisation and Novelty Detection, representing a more realistic application scenario. Both factors were evaluated using the previously established DKNN classification method on both the original and resized datasets.

6.7.1 Evaluation of Results

The DKNN classifier exhibits a maximum AUROC score of 0.5350 and an FPR95 of 84.82%. The overall performance, as reflected by the AUROC, is only slightly above random-level (0.5). While the FPR95 score indicates the model's performance at a fixed True Positive Rate (TPR) of 95%, it also reveals that 84.82% of ID samples are false classified as OOD. These results demonstrate that the DKNN classifier fails to effectively differentiate between ID and OOD classes when tested to simultaneous on Domain Generalization and Novelty Detection – a finding that remains consistent across both the original and resized datasets. The ROC curve further illustrates these findings, exhibiting a near-diagonal trajectory (see Figure 24 and 25).

6.7.2 Embeddings

The novelty detection experiment revealed that knowledge storage contains a compact representation of individual species, resulting in high recognition rates in the fine-grained experiment. However, this representation is limited by the spatial distance to other classes. A K-nearest neighbour (KNN) classifier compensates for this by considering the majority of neighbours, thereby being able to override spatially adjacent classes.

The novelty detection experiment uses a density-based threshold. This means that a compact representation of individual classes, as well as a spatial distance between the classes (ID and OOD), is required to enable successful classification based on

this metric. The experiments showed that the recognition rate is limited, meaning that the spatial separation between the classes in the feature space is also limited.

Domain generalisation experiments also revealed that embeddings from low-resolution case images significantly deviate from those of high-resolution images. This was accompanied by a significant drop in classification performance, even when the tolerant majority method of the KNN approach was applied. This demonstrates that the embeddings of low-resolution case images differ spatially from the clusters of high-resolution case images in the knowledge storage.

Applying this finding to the final experiment, we can deduce that there is only limited spatial separation of classes in the knowledge storage. Adding the low-resolution embedding, which deviates further from its associated cluster due to its low resolution, moves it further away from this cluster. This makes it impossible to correctly separate the classes based on density.

7 Conclusion

This master's thesis investigates and evaluates modern computer vision methods for the automated monitoring of moths. Given the context of global biodiversity loss and the need for scalable monitoring systems, this work assesses the suitability of self-supervised embeddings – extracted from the DINOv2 (Small) visual foundation model – to address challenges in Fine-Grained Visual Classification (FGVC), Domain Generalization (DG), and Novelty Detection (ND).

This thesis utilizes the AMI (Automated Monitoring of Insects) dataset as a foundation to simulate the realistic challenges encountered by automated field traps: taxonomic diversity, varying image quality, and the emergence of novel, invasive species. To address the seven established research questions, extensive experiments were conducted using K-Nearest Neighbor (KNN), Multi-Layer Perceptron (MLP), and Deep K-Nearest Neighbor (DKNN) classifiers.

This concluding Chapter summarizes the findings, provides a critical review of the methodological approaches, and answers the initial research questions. Furthermore, it offers an outlook on potential future research directions.

7.1 Summary of Research Objectives and Methodology

A central aspect of the research questions under investigation was the quality of the extracted embeddings. The efficacy with which these feature vectors mapped subtle morphological differences between moth species, while maintaining compact representations within the latent space, was a primary determinant of the classification performance. Consequently, the results serve as a direct reflection of the capability of the DINOv2 (Small) vision foundation model within the specialized domain of moths (entomology).

The seven research questions and associated experiments are categorized into two primary focus areas. The first area addresses the data foundation of the thesis:

- Assessment of the AMI dataset's suitability and availability as a benchmark for insect monitoring.
- Generation of the specific sub-datasets required for the subsequent experiments.
- Evaluation of padding methods for converting rectangular images into square formats, resulting in the creation of a secondary "resized" dataset.
- Extraction of feature embeddings from the generated datasets to serve as the basis for classification.

The second area focuses on the classification tasks (FGVC, DG, and ND), utilizing both the original and resized datasets:

- Application of KNN, MLP, and DKNN classifiers utilizing embeddings derived from high-resolution imagery.
- Performance evaluation of these classifiers using embeddings from both high-resolution (GBIF) and low-resolution trap images.

7.2 Answering the Research Questions

7.2.1 1: Availability and Integrity of the AMI Dataset

How completely can the AMI-GBIF subset be reconstructed, and to what extent is it suitable as a standardized benchmark in the long term?

The integrity of the database is a basic prerequisite for reproducible research. Since the AMI-GBIF sub-dataset is organized in a decentralized manner – metadata is provided, but image data must be loaded from the different sources (museums, citizen science platforms) – there is a risk of data loss.

The availability analysis carried out yielded the following findings:

GBIF Fine-Grained

- Of the 2,564,392 referenced image URLs in the fine-grained dataset, 2,520,912 were successfully retrieved.
- This corresponds to a success rate of 98.31% and a failure rate of 1.69% (43,480 images).

GBIF Binary

- Of the 700,000 referenced image URLs in the binary dataset, 692,311 were successfully retrieved.
- This yields a success rate of 98.91% and a failure rate of 1.09% (7,689 images).

Analysis of the error codes (see Appendix A.1) revealed that the majority of failures (totaling 49,806 cases) were due to HTTP error code 404 (Not Found). This confirms the hypothesis that external links constitute a unstable basis for reliable benchmarks.

Conclusion Although availability exceeding 98% (fine-grained) and 98% (binary) approximately seven months after is considered high, the loss of over 50,000 records indicates a substantial erosion of the dataset. This poses a challenge for its use as a strict benchmark, where models are compared in small percentage increments, as future research will be conducted on a slightly altered data foundation.

It is recommended that future studies precisely document the specific image data utilized. This ensures transparency and addresses the limitations regarding the

comparability of results within the AMI dataset. Furthermore, such documentation helps monitor dataset erosion and allows for the identification of data providers with high failure rates. Excluding these unreliable providers could result in a more temporally robust version of the dataset.

7.2.2 2: Influence of Padding Strategies on Embeddings

Which padding strategy is most suitable for converting rectangular moth images into the square input format of DINOv2 without adversely impacting embedding extraction?

The DINOv2 (Small) model requires square input images (518×518 pixels) and applies center cropping by default. Since cropped moth images are often rectangular, this approach potentially leads to the loss of relevant peripheral information (e.g., wing tips). To mitigate this, three padding methods were investigated: Mean Color (filling with average color), Small Blur (filling with noisy image fragments), and Large Patches (filling with large image sections).

Qualitative analysis, performed by visualizing the first principal component of the local patch embeddings, exhibited distinct differences:

1. Mean Color & 2. Small Blur

Depending on the background structure of the image, these methods produced sharp edges between the original image and the padded area. The PCA visualization (see Figure 31 and Appendix A.2) illustrated that the model interpreted these edges as dominant structures, which can disrupt the segmentation of the actual object (the moth).

3. Large Patches

This method achieved the best structural integration of the original image into the padded area. The segmentation of the primary object remained largely intact.

Conclusion Based on the segmentation quality, the Large Patches (3.) variant was identified as the most suitable. It effectively minimizes the introduction of high-contrast artifacts, which are most prominent in the Mean Color approach. However – and this is a critical point for the subsequent interpretation of the classification results – it introduces semantic redundancy. Duplicating parts of the moth in the background may provide additional information but simultaneously creates unnatural spatial relationships. This, in turn, could lead to a higher variance in the embedding encoding.

7.2.3 3: Creation of a Resized Dataset

Can the selected padding strategy be used to create a dataset that minimizes information loss, thereby yielding higher-quality embeddings and improving classification performance?

Based on the findings of Research Question 2, a resized dataset was created in parallel to the original dataset. All images were expanded to a square format using the Large Patches method before being processed by the embedding extractor (DINOv2).

Conclusion This implementation facilitated a direct comparison between 'cropped images' (original) and 'complete but padded images' (resized) across all subsequent experiments. It allows for isolating whether information loss due to cropping or the introduction of padded regions has a more adverse effect on the classifiers.

7.2.4 4: Performance in Fine-Grained Classification

How effectively can KNN and MLP classifiers, leveraging DINOv2 embeddings, distinguish between 91 moth species?

Regarding the Fine-Grained Visual Classification (FGVC) task on the high-resolution test set (Test FG), the experiments demonstrated high performance for the DINOv2 embeddings, albeit with distinct variations across different classifiers and datasets.

The results (summarized in Table 4) are as follows:

MLP vs. KNN

The MLP classifier (original dataset) achieved a Balanced Accuracy of 94.33% and a Macro Precision of 94.44%. It significantly outperformed the KNN approach (86.21% Balanced Accuracy) by approximately 8 percentage points.

Effect of the Resized Dataset

For the MLP, the Balanced Accuracy dropped to 92.32% (-2.01 percentage points).

For the KNN, it decreased to 81.67% (-4.54 percentage points).

Conclusion First, the high Accuracy of the MLP (94%) confirms that DINOv2 (Small) yields robust embeddings without fine-tuning, encoding even subtle differences between moth species.

Second, the superiority of the MLP over KNN demonstrates that raw cosine distances in the latent space are suboptimal for taxonomic separation. The MLP is

capable of highlighting specific dimensions of the 384-dimensional vector that are support for species differentiation through its weighted connections, whereas KNN treats all dimensions equally.

Third, the square padding strategy proved counterproductive for high-resolution images. The redundant information introduced by the large-patches method caused greater noise in the embedding than the gain in information could compensate for. DINOv2 appears sufficiently robust to correctly classify species even based on partial views (center crops).

7.2.5 5: Performance in Domain Generalization

How robustly do the classifiers respond to the covariate shift between high-resolution training imagery and low-resolution trap images?

This represents the primary challenge for real-world deployment. The models were trained on high-resolution image data and subsequently tested on actual field data from traps (Test FG+DG). The results (see Table 5) indicate a substantial performance degradation:

Performance Drop

The Balanced Accuracy of the MLP plummeted from 94.33% (GBIF) to 43.39% (Trap images). For KNN, it fell from 86.21% to 33.51%.

Effect of the Resized Dataset

Interestingly, an inverse trend was observed regarding the resized dataset compared to the fine-grained classification experiment. While it performed worse on high-resolution images, it yielded slightly improved (or comparable) results under domain generalization:

- KNN (resized) achieved 35.54% Balanced Accuracy (+2.03 percentage points compared to the original).
- MLP (resized) reached 43.78% (+0.39 percentage points).

Conclusion The decline in Accuracy to below 45% demonstrates that DINOv2 (Small) embeddings lack invariance to image quality (resolution) regarding the subtle differences between moth species. The domain gap is too extensive, leading to varying embeddings. In this context, the resized dataset provides advantages. This is likely due to the fact that trap images are frequently tightly cropped around the moth and predominantly rectangular, making information retention through padding more critical than in the original datasets. Nevertheless, the overall performance remains insufficient for real-world deployment.

7.2.6 6: Suitability for Novelty Detection (OOD)

How reliably can the DKNN method discriminate between 91 known species (ID) and 37 unknown species (OOD), leveraging DINOv2 embeddings?

This section investigates whether the cosine distance within the latent space can serve as a metric for novelty. Three thresholding variants were tested (*over_all_species*, *for_each_species*, *in_each_species*). When evaluated on the high-resolution test sets (Test FG and Test ND), the results (see Table 6) indicate moderate discriminatory power:

Best Configuration

The class-specific threshold approach using a KNN classifier yielded the best results on the original dataset:

- AUROC: 0.6525 (*in_each_species*)
- FPR95: 84.32% (*for_each_species*)

Comparison of Variants

Class-specific thresholds outperformed global thresholds (*over_all_species* AUROC 0.5963). This suggests that different moth species form clusters of varying density and compactness within the latent space.

Comparison of Datasets

For the resized dataset the class-specific threshold *for_each_species* combiend with KNN classifier yielded the best results:

- AUROC: 0.6209 ($-0,0316$ compared to the original)
- FPR95: 87.62% (+3.3 percentage points)

Conclusion An AUROC of 0.65 outperforms chance levels (0.5) but remains far below the threshold for reliable detection of 0.9. The FPR95 metric indicates that achieving a TPR of 95% for unknown (OOD) species results in 87.58% of known (ID) species being misclassified as unknown. While DINOv2 embeddings successfully cluster species, the distance to a novel or unseen species is often not significantly greater than the intra-class variance of known species.

7.2.7 7: Novelty Detection under Domain Shift

Can the DKNN method detect unknown species even when they are captured in low-quality trap imagery?

This represents the most challenging scenario – a combination of covariate shift and semantic shift – reflecting the real-world conditions of moth monitoring in the field.

When evaluated on the low-resolution test sets (Test FG + DG and Test ND + DG), The results (see Table 7) are sobering:

Near-random Performance

Out-of-Distribution (OOD) detection proved largely ineffective, yielding results comparable to random chance. The "highest" performance utilized the class-specific *for_each_species* thresholds in combination with the KNN classifier:

- AUROC: 0.5350 ($\Delta = 0.035$ above random chance)
- FPR95: 85.21%

Comparison of Datasets

Resized dataset archived compareable results as with the original dataset.

- AUROC: 0.5243 (*in_each_species*, KNN shows -0.011 compared to the original)
- FPR95: 84.82% (*for_each_species*, MLP shows $-0,39$ percentage points)

Conclusion In this scenario, the DKNN classifier performs only slightly better than random guessing. This is best illustrated by the ROC curve; see Figure 24 and 25. While high-resolution imagery facilitates class-specific clustering, inter-class separation in the latent space remains constrained. The drift exhibited by low-resolution trap embeddings away from these established clusters further blurs the decision boundaries. Consequently, the resulting overlap inhibits density-based differentiation approaches like DKNN.

7.3 Critical Discussion and Anomalies

Two observations are relevant for interpreting the results:

Data Redundancy Duplicates were identified within the dataset, specifically for the *Automeris io* species in the trap dataset and some for the GBIF datasets. The former stems from motion-triggered captures in the moth traps. For future iterations of the dataset, data deduplication using similarity analysis (e.g. cosine similarity of embeddings) is recommended.

Validation Anomaly During MLP training on the resized dataset, validation Accuracy exceeded training Accuracy by approximately 5 percentage points. This suggests that the potentially inconsistent embeddings generated by padding, combined with the WeightedRandomSampler (oversampling rare classes), hindered the learning process on the training data, whereas the validation set reflected the natural data distribution.

7.4 Limitations of the Work

The findings of this study are subject to specific limitations that must be considered during interpretation.

First, the evaluation was restricted to the DINOv2 (Small 22M parameters) model. While larger models, such as DINOv2 (Giant 1.1B parameters), demonstrate significantly better generalization in benchmarks, they were not evaluated due to computational resource constraints. It is highly probable that larger models would bridge the domain gap more effectively.

Second, to manage resource consumption and facilitate novelty detection experiments, the AMI dataset was subsampled to 128 species (91 ID / 37 OOD), and the long-tail distribution was truncated (minimum of 500 training images per class). Consequently, the results should be viewed as optimistic and do not reflect the performance that would be achieved on the full AMI dataset, particularly regarding few-shot learning challenges.

Finally, this study did not employ domain adaptation techniques; the focus remained exclusively on thetransfer capabilities of the foundation model.

7.5 Outlook and Recommendations

The following recommendations are proposed for the further development of automated monitoring systems in visual moth classification:

Utilization of Regional AMI Subsets These subsets provide a realistic deployment scenario, as they reflect the requirements of actual moth monitoring systems. Furthermore, regional subsets are more limited in size compared to the full AMI dataset. Their use also facilitates a direct comparison with previously published classification benchmarks.

Utilization of Advanced Models (DINOv3) Since the conclusion of these experiments, DINOv3 has been released. Future research should evaluate whether newer model or larger model variants (e.g., DINOv2-Giant) offer enhanced robustness for DG or ND. Increasing the embedding dimensionality (from 384 to 1536 or higher) may facilitate better cluster separation within the latent space.

Background Replacement instead of Square Padding Square padding showed limited to negative effects in this study. A more promising approach would be background replacement (see Figure 28). Given that trap images typically feature a homogeneous background, segmentation based on the PCA of DINOv2 embeddings could be used to remove complex backgrounds from GBIF imagery. Replacing these with a neutral color consistent with the trap

screen could mitigate the covariate shift between GBIF training and trap images. This method is also suitable for converting rectangular images to a square format while maintaining focus on the relevant features.

Moving away from a strictly DG approach Results regarding domain generalization indicate that high-resolution training data combined with DINOv2 alone are insufficient. Future systems should incorporate domain adaptation strategies. One promising method is the synthetic degradation of GBIF imagery: By applying blurring and downsampling, the model can be trained to extract features that remain robust against low-quality image conditions.

Hierarchical Classification The inherent difficulties of FGVC, DG and ND could be mitigated through a hierarchical approach. Instead of attempting immediate species-level identification, the system could first classify the specimen at genus level.

7.6 Final Conclusion

This work demonstrates that DINOv2 is a robust tool for moth classification, with promising applicability to other insect taxa. However, it also highlights that performance on low-resolution field imagery remains limited due to the covariate shift.

Under ideal conditions with high-resolution data, the classification methods achieve outstanding results. Utilizing simple MLP heads, a Balanced Accuracy of over 94% was reached, which has the potential to significantly accelerate taxonomic workflows. Furthermore, the approach is highly efficient as it avoids full backbone retraining. Despite an 8% performance trade-off, KNN models remain a viable alternative, offering user-friendly features such as interpretability and continuous learning.

Significant challenges remain for deployment in the field (trap imagery). The sensitivity of embeddings to image quality results in a performance drop that, in its current state, limits the feasibility of the system for autonomous operation. Furthermore, novelty detection lacks the necessary robustness to consistently identify invasive species.

A Appendix

A.1 Reasons for Failed GBIF Image Downloads

The following section summarizes the reasons for failed image downloads within the GBIF Fine-Grained and GBIF Binary datasets.

By far the most prevalent issue, accounting for 49,806 cases, was the *404 Client Error*. This indicates that the referenced image is no longer available at the specified URL. A similar but significantly rarer issue was the *403 Client Error* (68 cases), which points to insufficient access permissions rather than a non-existent resource.

The second most common error was a connection failure of the type *Max retries exceeded with URL*, occurring in 1,120 cases (1,069 + 51). This indicates that the server referenced in the URL could not be reached or that the connection attempts failed repeatedly.

In third place, with 156 instances, was the *file corruption* error previously discussed in Subsection 3.1.1. In these cases, the image was successfully retrieved via the URL, but the resulting file was corrupted or unreadable.

Another cause of failure is categorized under the error message *connection aborted*. In these instances, the connection was terminated by the server 12 times during the image file retrieval.

All reasons for failed or unsuccessful image downloads from the fine-grain and binary datasets are summarized in the Table 8 below. Similar error types have been grouped together for clarity. For instance, the errors *HTTP Max retries exceeded* and *HTTPS Max retries exceeded* differ primarily in the underlying protocol rather than the nature of the error itself.

	Download Error Description	Sum
1.	404 Client Error: Not Found for url	49806
2.	403 Client Error: Forbidden for url	68
3.	HTTPSConnectionPool(host='*',port=*): Max retries exceeded with url	1069
4.	HTTPConnectionPool(host='*',port=*): Max retries exceeded with url	51
5.	HTTPSConnectionPool(host='*',port=*): Read timed out. (read timeout=300)	7
6.	Connection aborted, RemoteDisconnected('Remote end closed connection without response')	9
7.	Connection aborted, ConnectionResetError(104, 'Connection reset by peer')	3
8.	invalid image files	156
		51.169

Table 8: Summary of error types and their frequency during image downloads. The table divide the errors into HTTP status codes, network connection problems, and file integrity errors.

A.2 Result of all Padding Samples

The following figures (high-resolution samples: Figures 29, 30, 31 and low-resolution samples: Figures 32, 33, 34) illustrate the results of all samples utilized in the padding experiment.

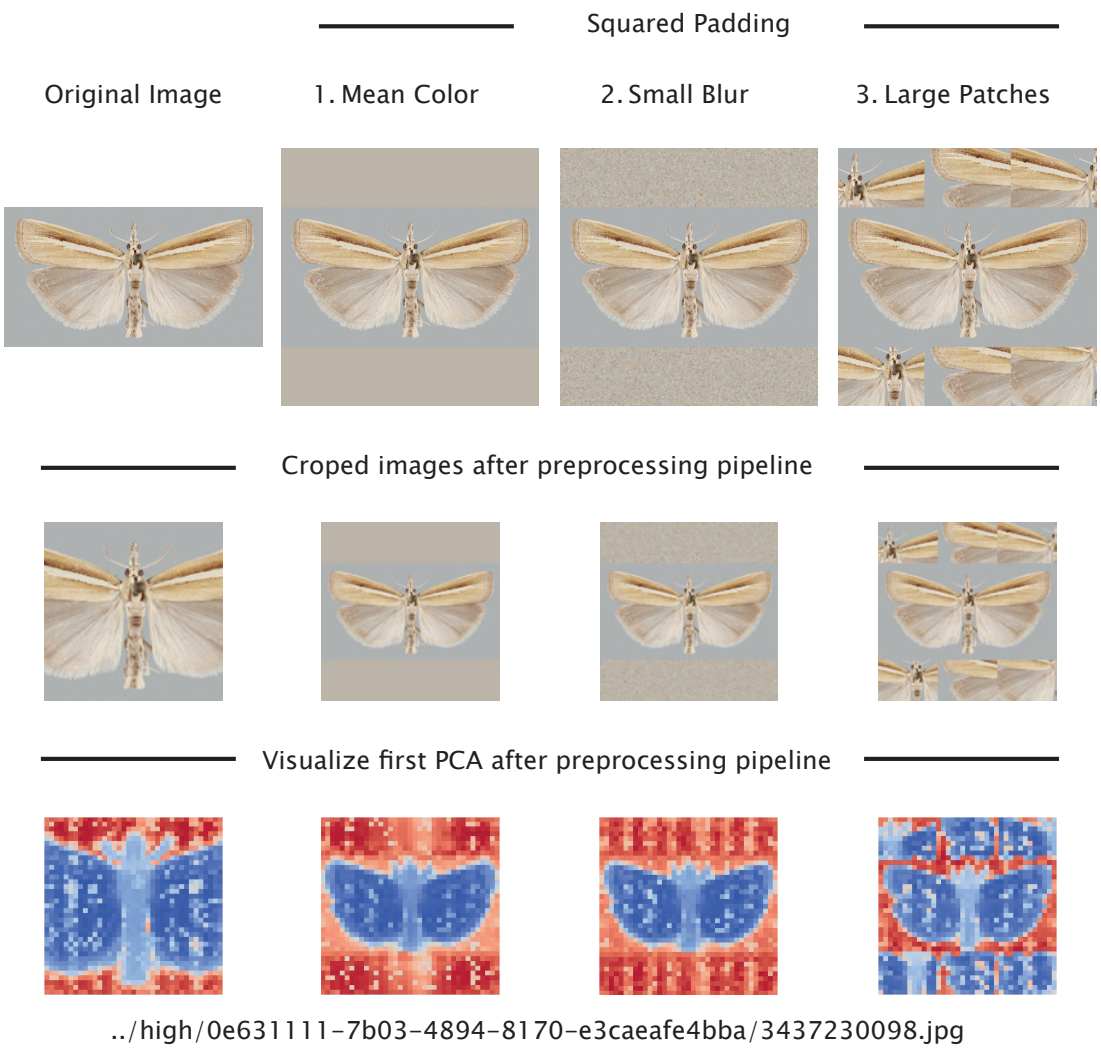


Figure 29: Overview of padding strategies sample one: The first row illustrates the different padding strategies compared to the original image. The second row displays the images after being processed by the preprocessing pipeline, and the third row presents the corresponding segmentation results.

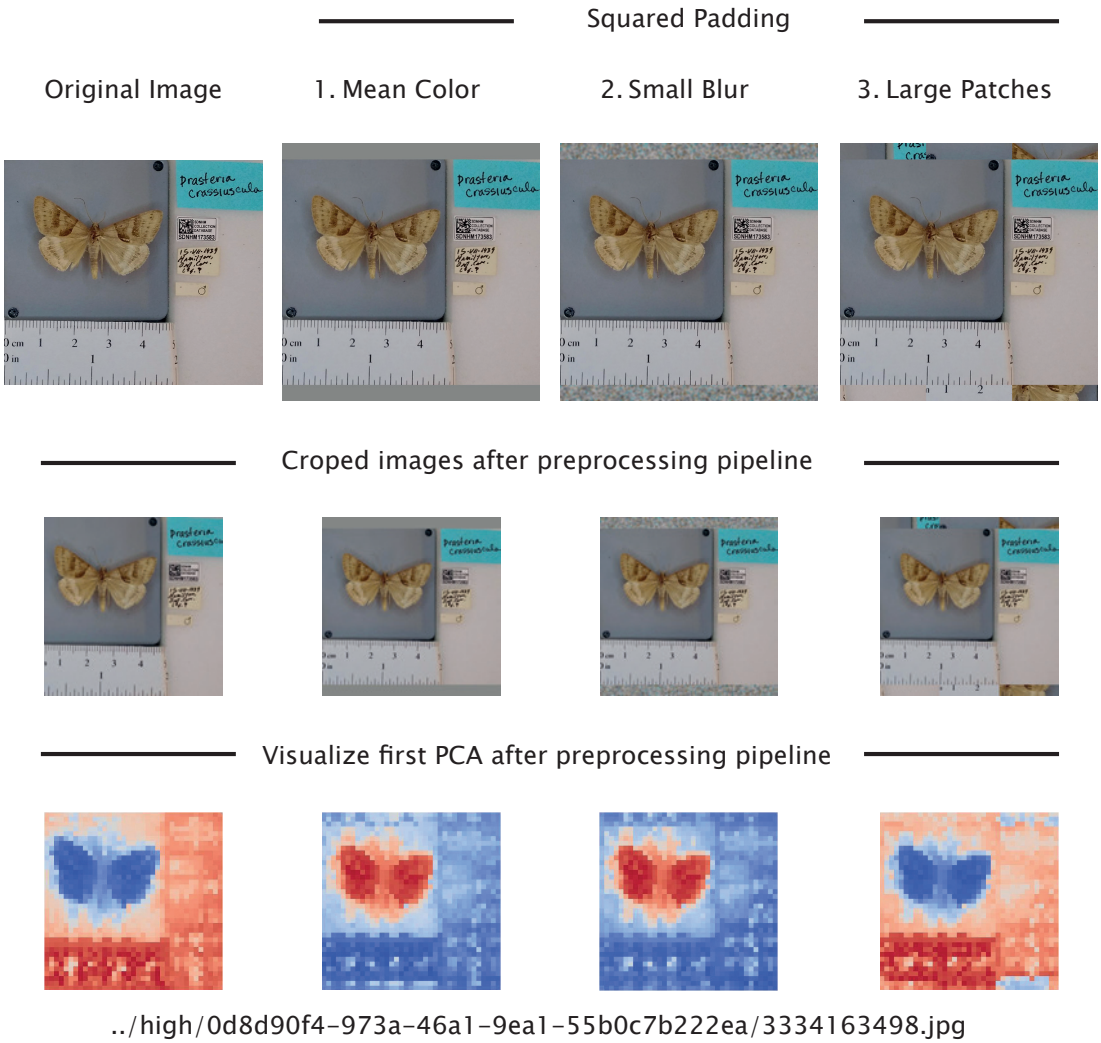


Figure 30: Overview of padding strategies sample two: The first row illustrates the different padding strategies compared to the original image. The second row displays the images after being processed by the preprocessing pipeline, and the third row presents the corresponding segmentation results.

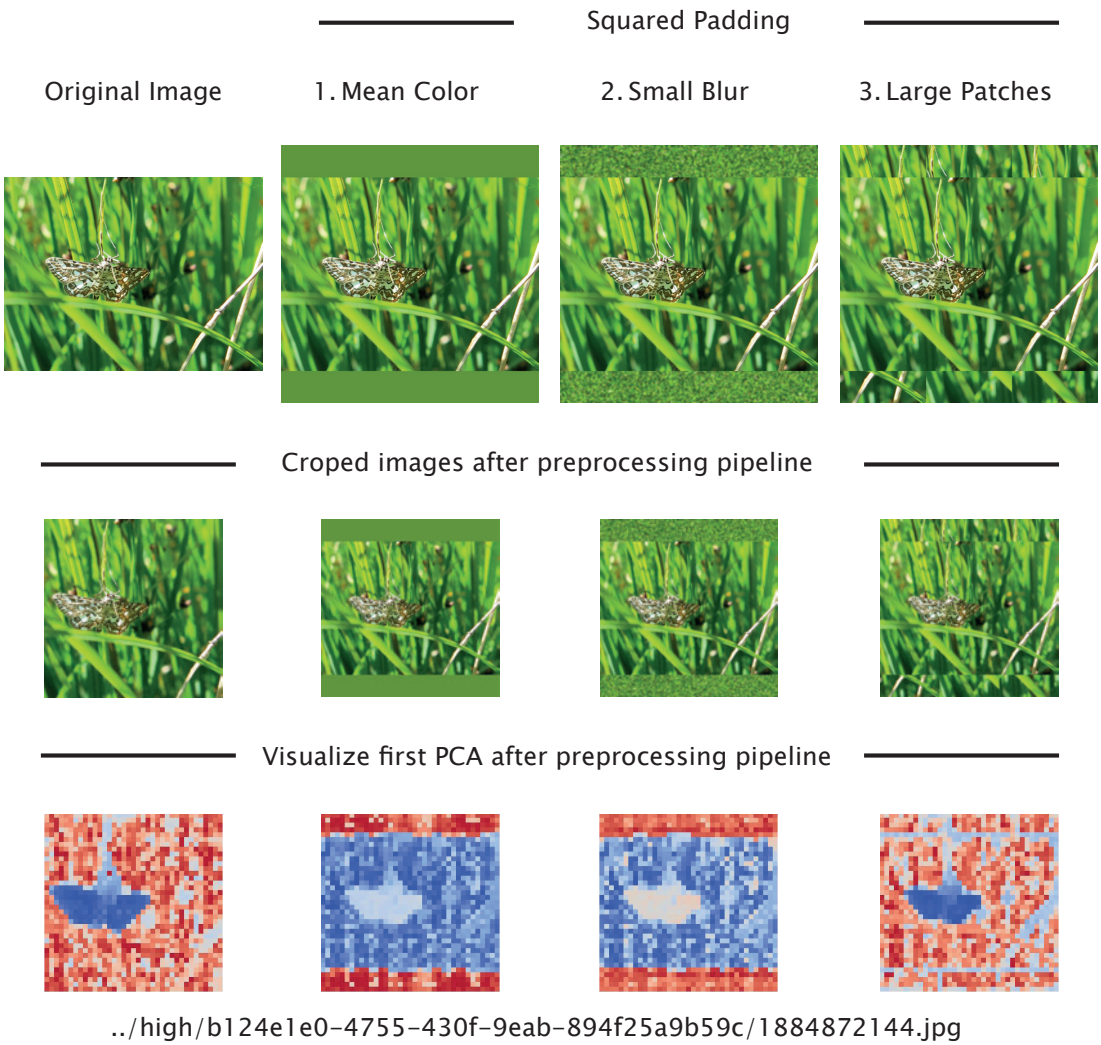


Figure 31: Overview of padding strategies sample three: The first row illustrates the different padding strategies compared to the original image. The second row displays the images after being processed by the preprocessing pipeline, and the third row presents the corresponding segmentation results.

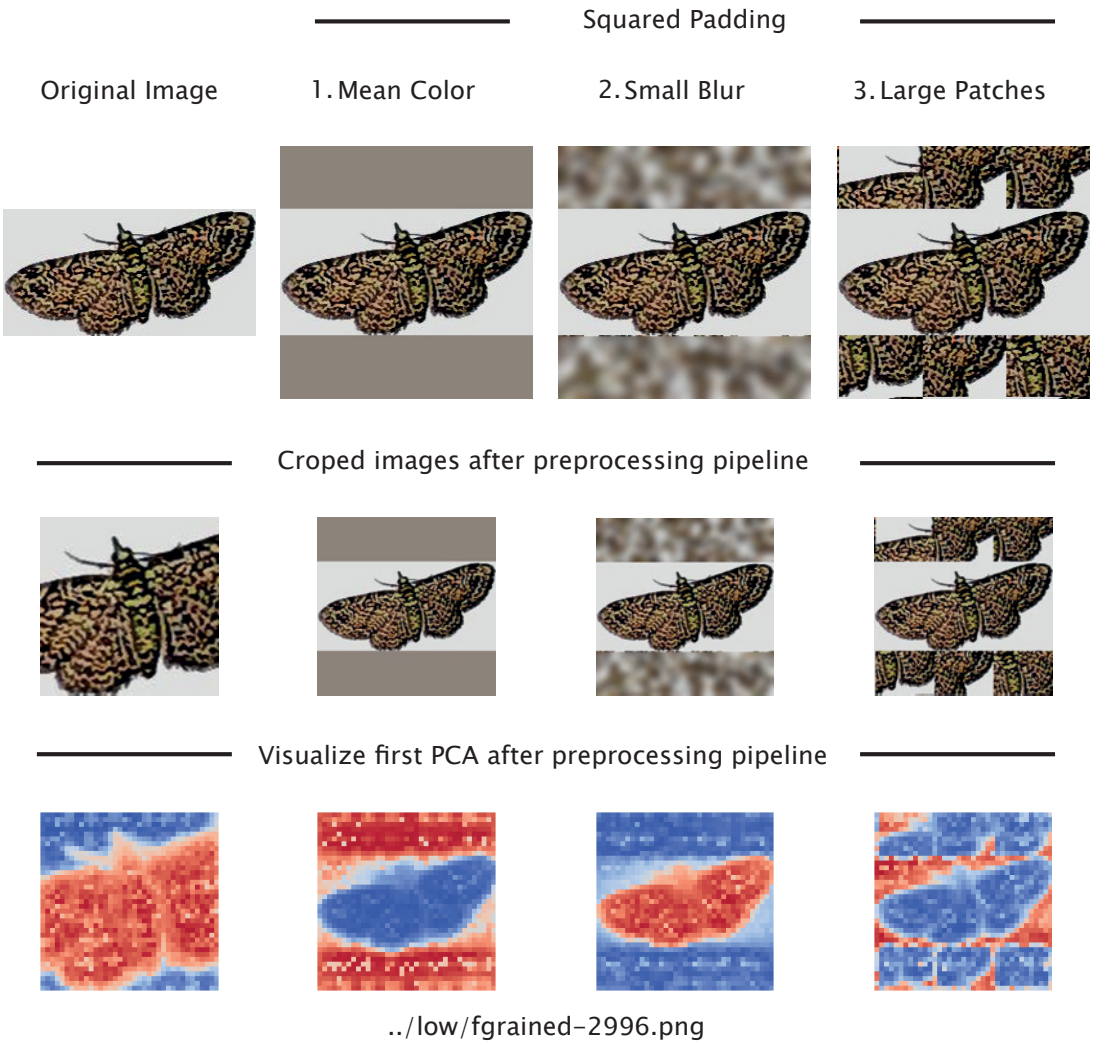


Figure 32: Overview of padding strategies sample four: The first row illustrates the different padding strategies compared to the original image. The second row displays the images after being processed by the preprocessing pipeline, and the third row presents the corresponding segmentation results.

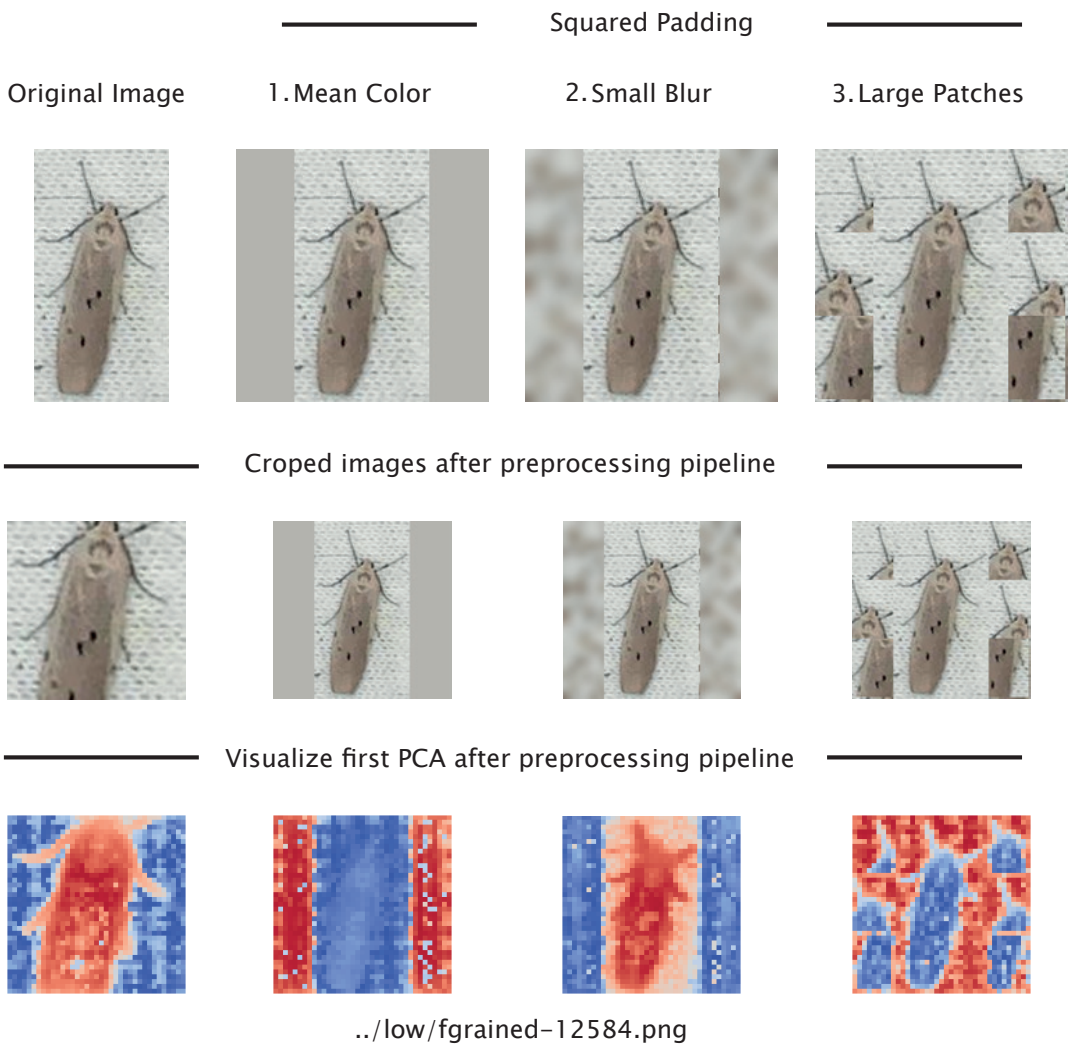


Figure 33: Overview of padding strategies sample five: The first row illustrates the different padding strategies compared to the original image. The second row displays the images after being processed by the preprocessing pipeline, and the third row presents the corresponding segmentation results.

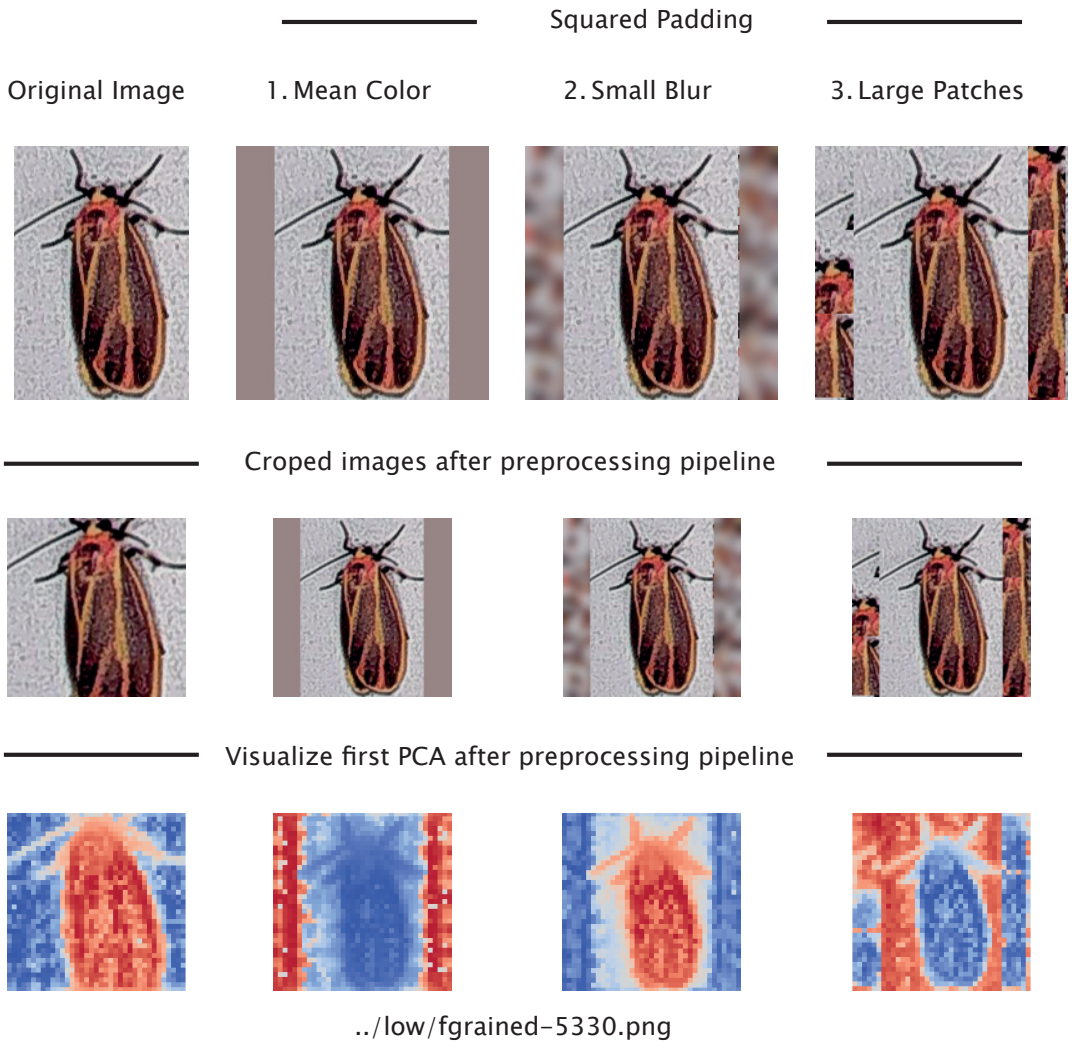


Figure 34: Overview of padding strategies sample six: The first row illustrates the different padding strategies compared to the original image. The second row displays the images after being processed by the preprocessing pipeline, and the third row presents the corresponding segmentation results.

A.3 Confusion Matrix

The confusion matrix shows, for each test sample, both the predicted class and the corresponding ground truth. This enables a detailed analysis of typical misclassifications between individual moth species.

The confusion matrices generated for this study are visualized as heatmaps. Due to the large number of classes and the resulting high resolution, their display within this document is limited and intended primarily as an overview. For a more detailed examination, the full confusion matrices are provided as high-resolution PDFs[61].

A.3.1 Fine-Grained Classification

Original Dataset Figures 35 and 36 present the confusion matrices for the fine-grain classifiers KNN and MLP, using the original dataset.

[61]The confusion matrices can be found via the following paths:
/masterArbeit/models/knn/origin/scores/visualisation
/masterArbeit/models/knn/resized/scores/visualisation

/masterArbeit/models/mlp/origin/scores/visualisation
/masterArbeit/models/mlp/resized/scores/visualisation

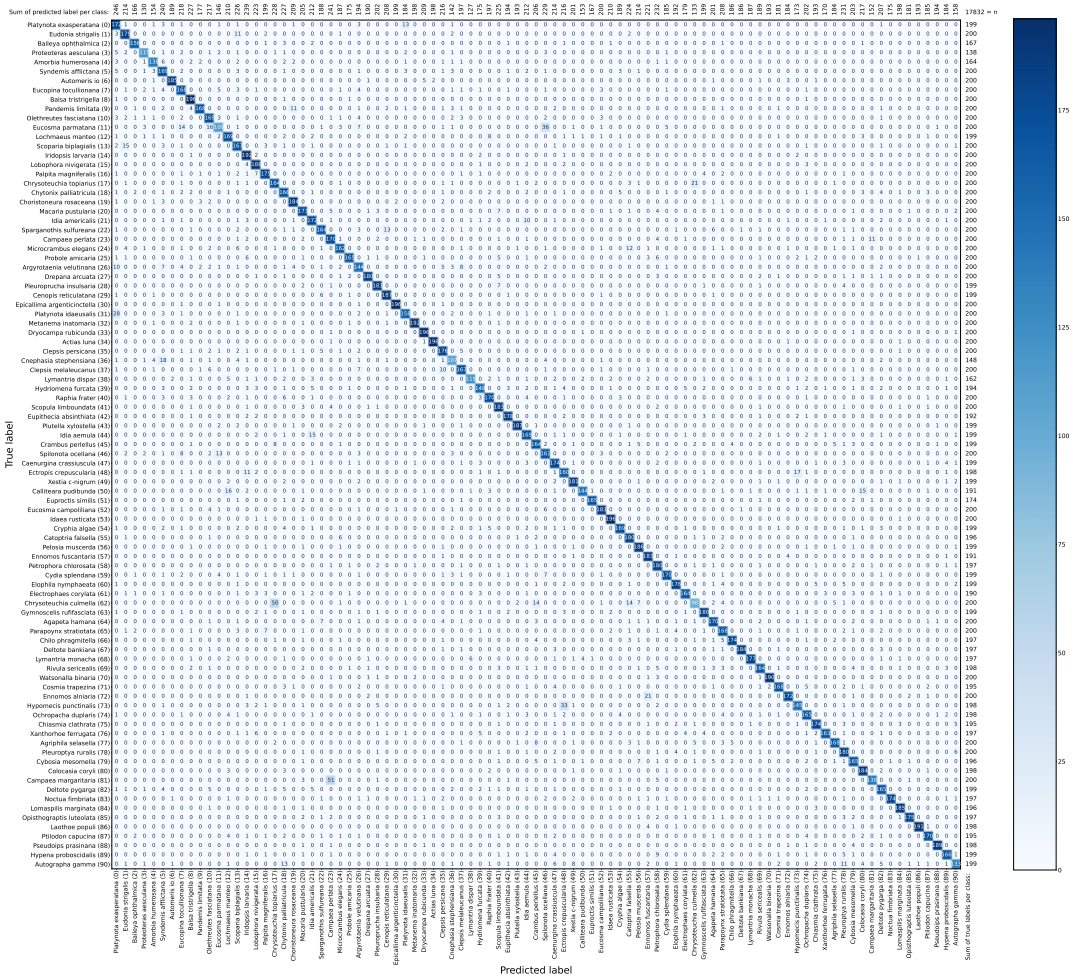


Figure 35: The confusion matrix illustrates the fine-grained classification results obtained with the KNN classifier on the original dataset. It contrasts the true labels (rows) with the predicted labels (columns), highlighting the distribution of correct classifications along the main diagonal.

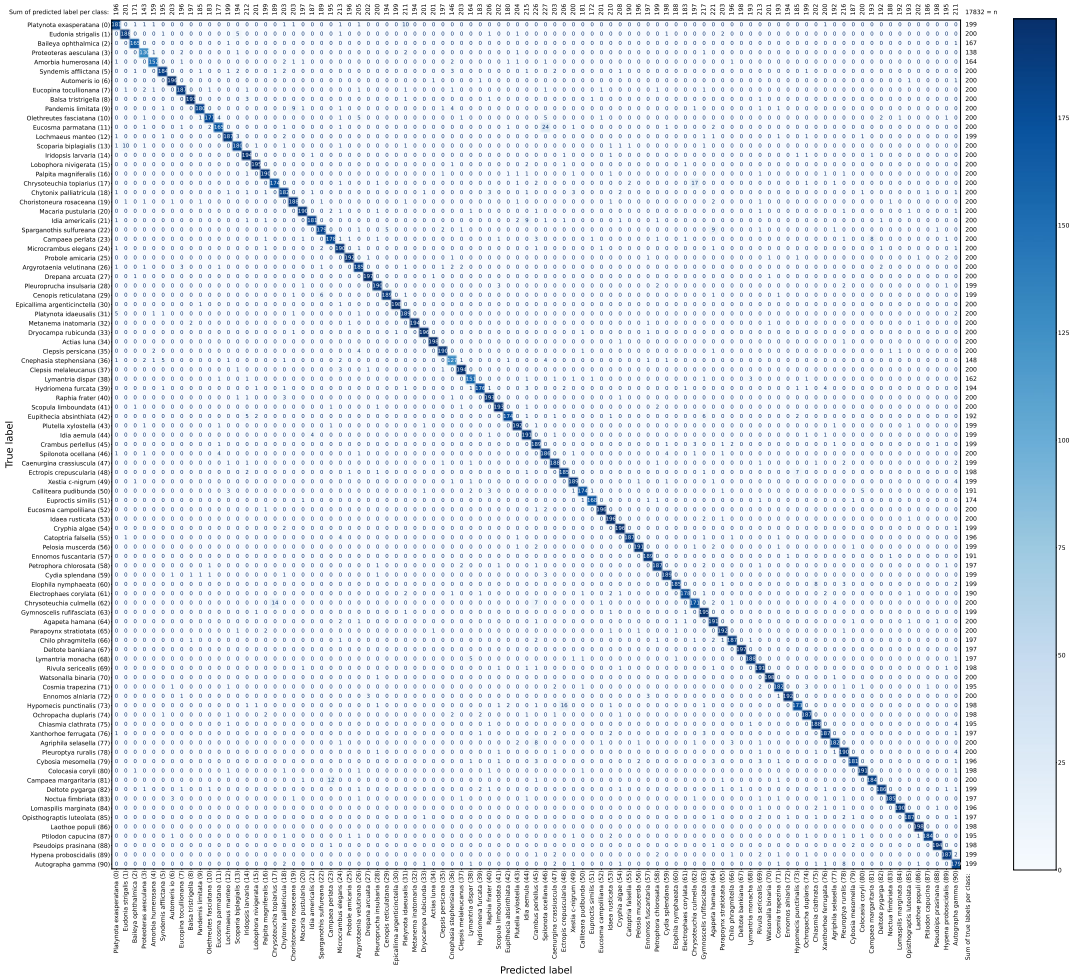


Figure 36: The confusion matrix illustrates the fine-grained classification results obtained with the MLP classifier on the original dataset. It contrasts the true labels (rows) with the predicted labels (columns), highlighting the distribution of correct classifications along the main diagonal.

Resized Dataset Figures 37 and 38 present the confusion matrices for the fine-grain classifiers KNN and MLP, using the resized dataset.

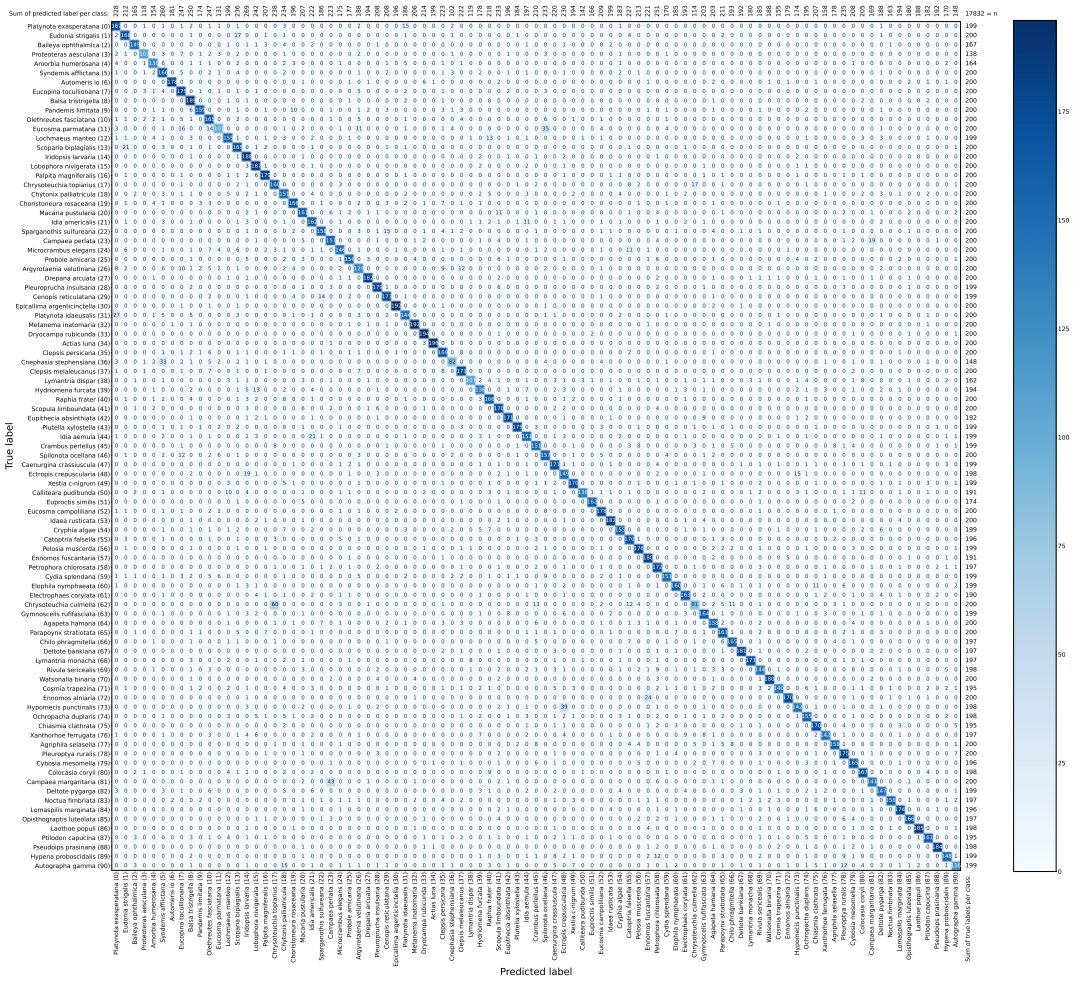


Figure 37: The confusion matrix illustrates the fine-grained classification results obtained with the KNN classifier on the resized dataset. It contrasts the true labels (rows) with the predicted labels (columns), highlighting the distribution of correct classifications along the main diagonal.

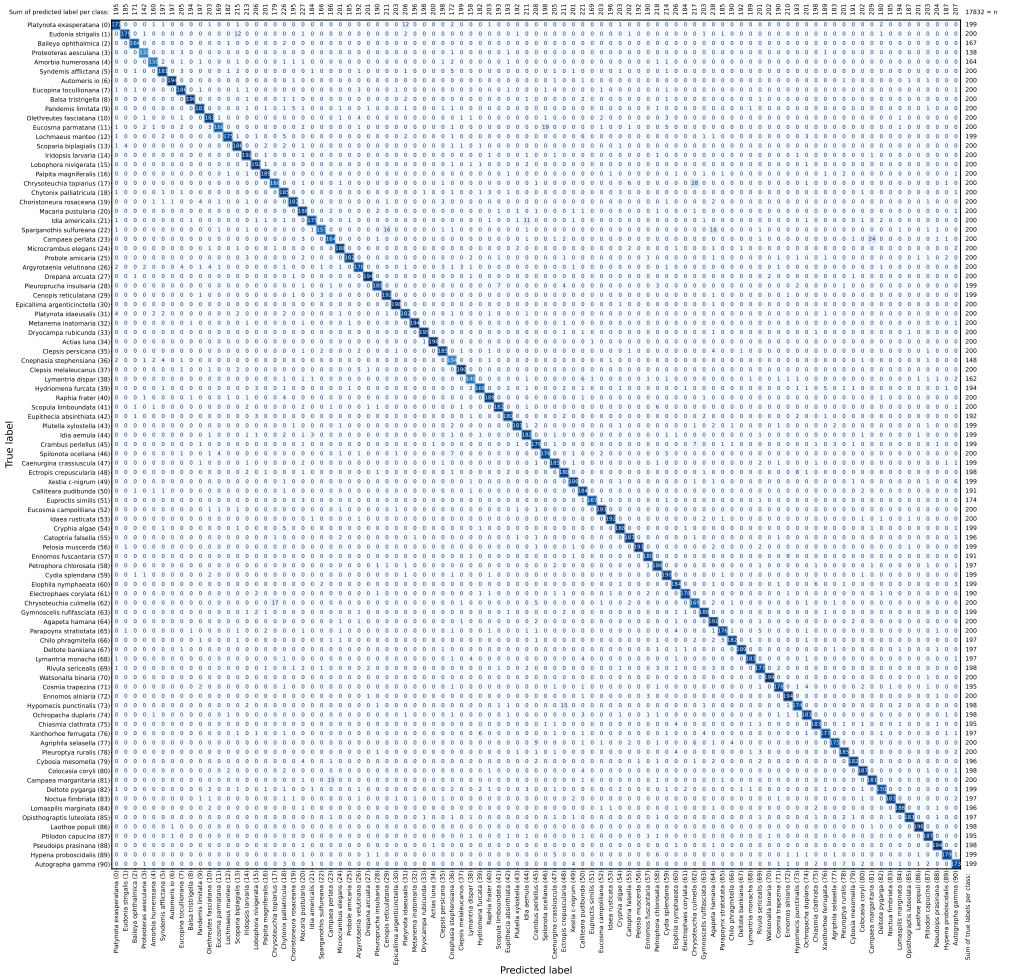


Figure 38: The confusion matrix illustrates the fine-grained classification results obtained with the MLP classifier on the resized dataset. It contrasts the true labels (rows) with the predicted labels (columns), highlighting the distribution of correct classifications along the main diagonal.

A.3.2 Domain Generalisation

Original Dataset Figures 39 and 40 present the confusion matrices of the domain generalization experiments for the KNN and MLP classifiers using the original dataset.

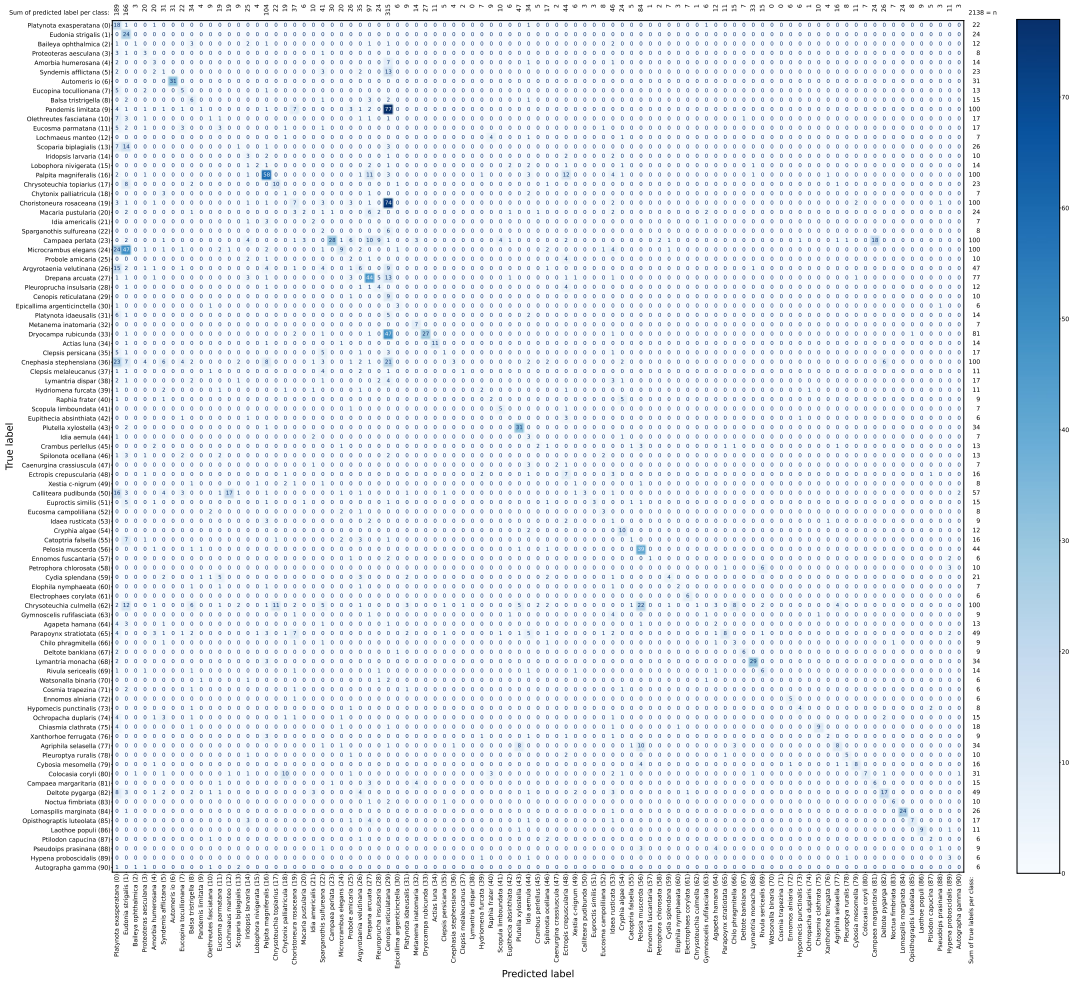


Figure 39: The confusion matrix illustrates the domain generalisation results obtained with the KNN classifier on the original dataset. It contrasts the true labels (rows) with the predicted labels (columns), highlighting the distribution of correct classifications along the main diagonal.

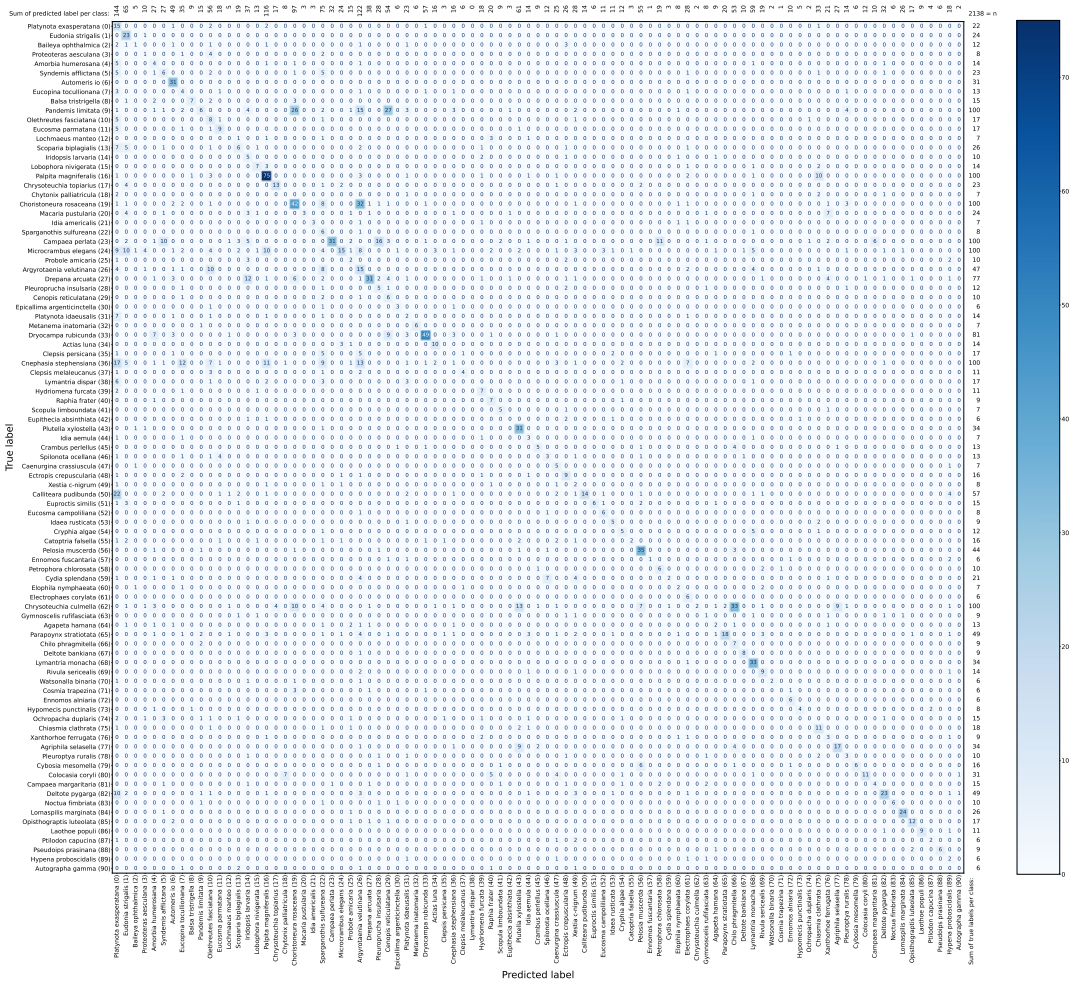


Figure 40: The confusion matrix illustrates the domain generalisation results obtained with the MLP classifier on the original dataset. It contrasts the true labels (rows) with the predicted labels (columns), highlighting the distribution of correct classifications along the main diagonal.

Resized Dataset Figures 41 and 42 present the confusion matrices of the domain generalization experiments for the KNN and MLP classifiers using the resized dataset.

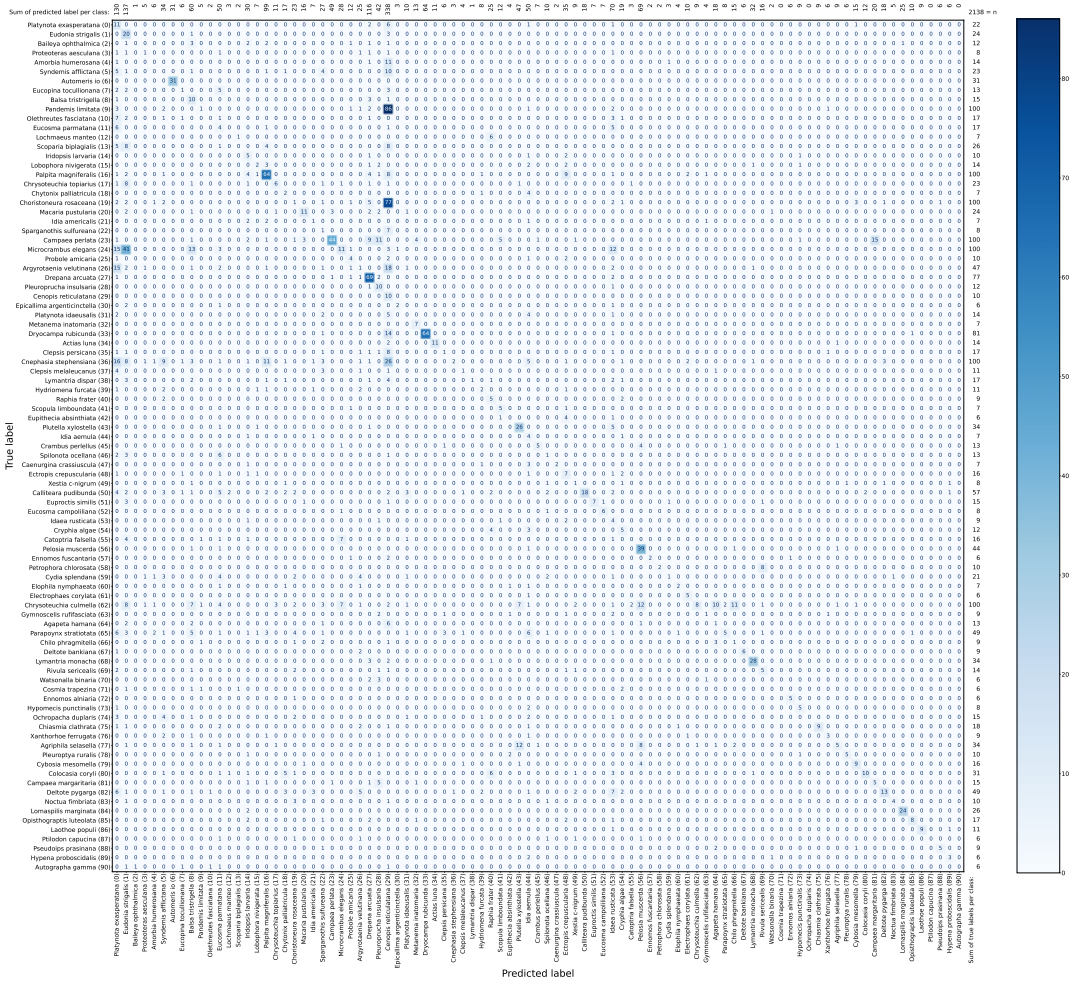


Figure 41: The confusion matrix illustrates the domain generalisation results obtained with the KNN classifier on the resized dataset. It contrasts the true labels (rows) with the predicted labels (columns), highlighting the distribution of correct classifications along the main diagonal.

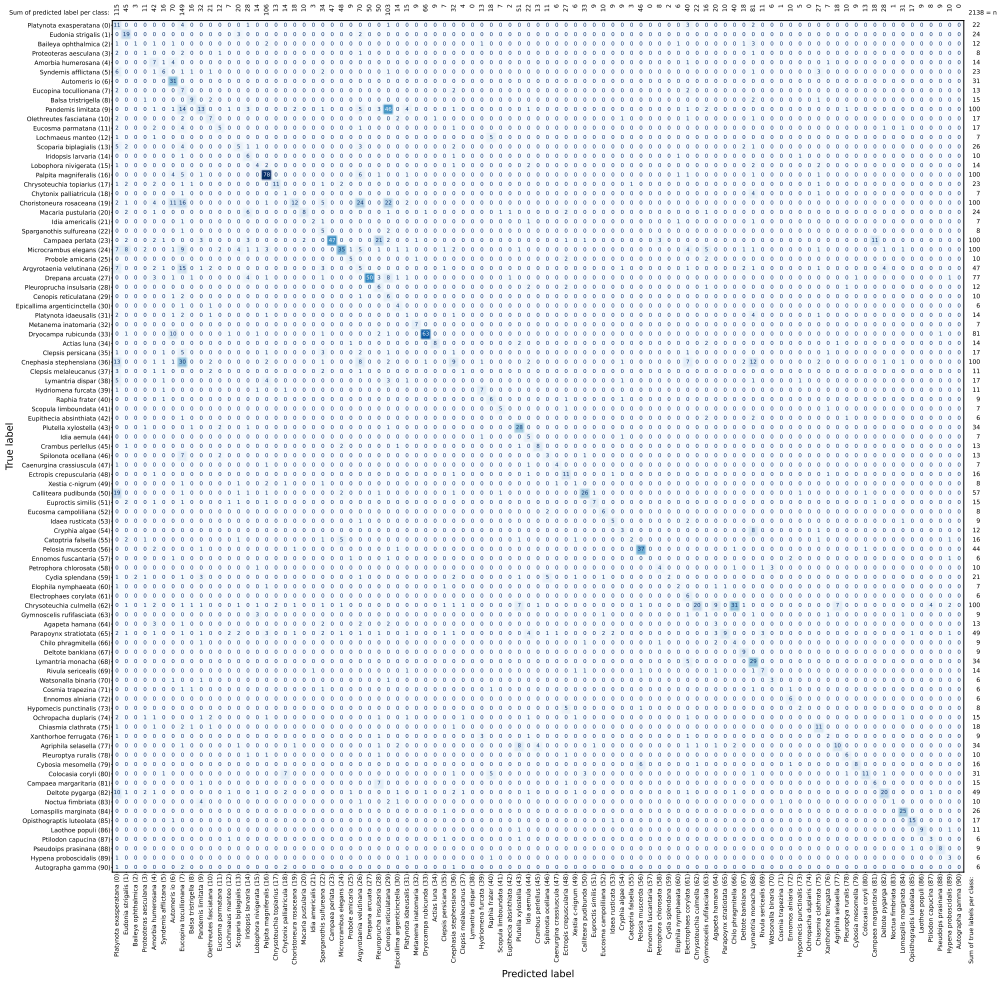


Figure 42: The confusion matrix illustrates the domain generalisation results obtained with the MLP classifier on the resized dataset. It contrasts the true labels (rows) with the predicted labels (columns), highlighting the distribution of correct classifications along the main diagonal.

Bibliography

Analyse des Wandels der Biodiversität (LIB), Leibniz-Institut zur (2025). *LEPMON – Erfassung Der Biodiversität von Nachtfaltern (Lepidoptera) Mit Automatisierten Kamerafallen Und Künstlicher Intelligenz*. LEPMON. URL: https://lepmon.de/ (visited on 07/07/2025).

Bjerge, Kim, Hjalte M. R. Mann, and Toke Thomas Høye (2022). "Real-Time Insect Tracking and Monitoring with Computer Vision and Deep Learning". In: *Remote Sensing in Ecology and Conservation* 8.3, pp. 315–327. ISSN: 2056-3485. DOI: 10.1002/rse2.245. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/rse2.245 (visited on 09/29/2025).

Bjerge, Kim, Jakob Bonde Nielsen, Martin Videbæk Sepstrup, Flemming Helsing-Nielsen, and Toke Thomas Høye (Jan. 2021). "An Automated Light Trap to Monitor Moths (Lepidoptera) Using Computer Vision-Based Tracking and Deep Learning". In: *Sensors* 21.2, p. 343. ISSN: 1424-8220. DOI: 10.3390/s21020343. URL: https://www.mdpi.com/1424-8220/21/2/343 (visited on 09/30/2025).

BMFTR (Jan. 4, 2023). *Richtlinie zur Förderung von Projekten zum Thema „Methoden der Künstlichen Intelligenz als Instrument der Biodiversitätsforschung"*. Bekanntmachung / Förderrichtlinie. Bundesministerium für Forschung, Technologie und Raumfahrt (BMFTR). URL: https://www.bmftr.bund.de/SharedDocs/Bekanntmachungen/DE/2023/01/2023-01-04-Bekanntmachung-Biodiversit%C3%A4tsforschung.html (visited on 09/29/2025).

Chen, Yuyan and David Rolnick (Dec. 16, 2023). "Understanding Insect Range Shifts with Out-of-Distribution Detection". In: *Climate Change AI*. NeurIPS 2023 Workshop on Tackling Climate Change with Machine Learning. Climate Change AI. URL: https://www.climatechange.ai/papers/neurips2023/130 (visited on 10/25/2024).

Chu, Yang, Minchao Ye, and Yuntao Qian (2024). "Fine-Grained Image Recognition Methods and Their Applications in Remote Sensing Images: A Review". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 17, pp. 19640–19667. ISSN: 2151-1535. DOI: 10.1109/JSTARS.2024.3482348. URL: https://ieeexplore.ieee.org/document/10720642 (visited on 10/06/2025).

Clark, Alex (2026). *Pillow (PIL Fork) 12.1.0 Documentation*. Pillow (PIL Fork). URL: https://pillow.readthedocs.io/ (visited on 01/31/2026).

Consortium-Steering-Committee (2025). *Automated Monitoring of Insects (AMI)*. The AMI Initiatives. URL: https://www.insectmonitoring.org/initiatives (visited on 09/30/2025).

Cunningham, Pádraig and Sarah Jane Delany (July 13, 2021). "K-Nearest Neighbour Classifiers - A Tutorial". In: *ACM Comput. Surv.* 54.6, 128:1–128:25. ISSN: 0360-0300. DOI: 10.1145/3459665. URL: https://dl.acm.org/doi/10.1145/3459665 (visited on 10/02/2024).

Doerrich, Sebastian, Tobias Archut, Francesco Di Salvo, and Christian Ledig (May 2024). "Integrating kNN with Foundation Models for Adaptable and Privacy-Aware Image Classification". In: *2024 IEEE International Symposium on Biomedical Imaging (ISBI)*. 2024 IEEE International Symposium on Biomedical Imaging (ISBI), pp. 1–5. DOI: 10.1109/ISBI56570.2024.10635560. URL: https://ieeexplore.ieee.org/document/10635560/?arnumber=10635560 (visited on 10/18/2024).

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby (Oct. 2, 2020). "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: International Conference on Learning Representations. URL: https://openreview.net/forum?id=YicbFdNTTy (visited on 01/31/2026).

FEdA (2025). *Forschungsprojekte der Richtlinie „Methoden der Künstlichen Intelligenz als Instrument der Biodiversitätsforschung" (BiodivKI)*. BiodivKi-Projekte. URL: https://www.feda.bio/de/projekte/biodivki/ (visited on 07/07/2025).

GBIF (2024). *Global Biodiversity Information Facility (GBIF)*. URL: https://www.gbif.org/ (visited on 10/25/2024).

General Data Protection Regulation (Apr. 27, 2016). URL: https://gdpr-info.eu/art-17-gdpr/ (visited on 10/21/2025).

Grandini, Margherita, Enrico Bagli, and Giorgio Visani (Aug. 13, 2020). *Metrics for Multi-Class Classification: An Overview*. DOI: 10.48550/arXiv.2008.05756. arXiv: 2008.05756 [stat]. URL: http://arxiv.org/abs/2008.05756 (visited on 07/22/2025). Pre-published.

Heppner, John B. (2008). "Moths (Lepidoptera: Heterocera)". In: *Encyclopedia of Entomology*. Springer, Dordrecht, pp. 2491–2494. ISBN: 978-1-4020-6359-6. DOI: 10.1007/978-1-4020-6359-6_4705. URL: https://link.springer.com/rwe/10.1007/978-1-4020-6359-6_4705 (visited on 12/28/2025).

Jain, Aditya, Fagner Cunha, Michael Bunsen, Léonard Pasi, Anna Viklund, Maxim Larrivee, and David Rolnick (Dec. 16, 2023). "A Machine Learning Pipeline for Automated Insect Monitoring". In: *Climate Change AI*. NeurIPS 2023 Workshop on Tackling Climate Change with Machine Learning. Climate Change AI. URL: https://www.climatechange.ai/papers/neurips2023/47 (visited on 10/02/2024).

Jain, Aditya, Fagner Cunha, Michael James Bunsen, Juan Sebastián Cañas, Léonard Pasi, Nathan Pinoy, Flemming Helsing, JoAnne Russo, Marc Botham, Michael Sabourin, Jonathan Fréchette, Alexandre Anctil, Yacksecari Lopez, Eduardo Navarro, Filonila Perez Pimentel, Ana Cecilia Zamora, José Alejandro Ramirez Silva, Jonathan Gagnon, Tom August, Kim Bjerge, Alba Gomez Segura, Marc Bélisle, Yves Basset, Kent P. McFarland, David Roy, Toke Thomas Høye, Maxim Larrivée, and David Rolnick (2025). "Insect Identification in the Wild: The AMI Dataset". In: *Computer Vision – ECCV 2024 - 18th European Conference, Proceedings*. Lecture Notes in Computer Science. Ed. by Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, pp. 55–73. ISSN: 9783031729126. DOI: 10.1007/978-3-031-72913-3_4. URL: https://www.scopus.com/pages/publications/85211768804 (visited on 01/30/2026).

Jain, Aditya, Fagner Cunha, Michael James Bunsen, Juan Sebastián Cañas, Léonard Pasi, Nathan Pinoy, Flemming Helsing, JoAnne Russo, Marc Botham, Michael Sabourin, Jonathan Fréchette, Alexandre Anctil, Yacksecari Lopez, Eduardo Navarro, Filonila Perez Pimentel, Ana Cecilia Zamora, José Alejandro Ramirez Silva, Jonathan Gagnon, Tom August, Kim Bjerge, Alba Gomez Segura, Marc Bélisle, Yves Basset, Kent P. McFarland, David Roy, Toke Thomas Høye, Maxim Larrivée, and David Rolnick (2024). "Insect Identification in the Wild: The AMI Dataset". Version 2. In: DOI: 10.48550/ARXIV.2406.12452. URL: https://arxiv.org/abs/2406.12452 (visited on 04/11/2025).

Kundu, Rohit (2026). *Confusion Matrix: A Complete Guide*. V7 Labs Blog. URL: https://www.v7labs.com/blog/confusion-matrix-guide (visited on 02/07/2026).

McKinney, Wes (May 1, 2010). "Data Structures for Statistical Computing in Python". In: *SciPy 2010*. DOI: 10.25080/Majora-92bf1922-00a. URL: https://proceedings.scipy.org/articles/Majora-92bf1922-00a (visited on 01/31/2026).

Nakata, Kengo, Youyang Ng, Daisuke Miyashita, Asuka Maki, Yu-Chieh Lin, and Jun Deguchi (2022). "Revisiting a kNN-Based Image Classification System with High-Capacity Storage". In: *Computer Vision – ECCV 2022*. Ed. by Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner. Cham: Springer Nature Switzerland, pp. 457–474. ISBN: 978-3-031-19836-6. DOI: 10.1007/978-3-031-19836-6_26.

Oquab, Maxime, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski (Feb. 2, 2024). "DINOv2: Learning Robust Visual Features without Supervision". In: *Transactions on Machine Learning Research (TMLR)*,

p. 32. ISSN: 2835-8856. DOI: 10.48550/arXiv.2304.07193. arXiv: 2304.07193. URL: https://openreview.net/forum?id=a68SUt6zFt (visited on 10/25/2024).

Oubabas, Sarah, Hassina Ait Issad, Karima Chemoun, Rachida Aoudjit, Malika Belkadi, and Mhammed Daoui (Nov. 2024). "Pests Classification Based on Convolutional Neural Networks in Smart Agriculture". In: *2024 International Conference of the African Federation of Operational Research Societies (AFROS)*. 2024 International Conference of the African Federation of Operational Research Societies (AFROS), pp. 1–6. DOI: 10.1109/AFROS62115.2024.11037078. URL: https://ieeexplore.ieee.org/document/11037078 (visited on 07/08/2025).

Passias, Athanasios, Karolos-Alexandros Tsakalos, Nick Rigogiannis, Dionisis Voglitsis, Nick Papanikolaou, Maria Michalopoulou, George Broufas, and Georgios Ch. Sirakoulis (Sept. 2024). "Insect Pest Trap Development and DL-Based Pest Detection: A Comprehensive Review". In: *IEEE Transactions on AgriFood Electronics* 2.2, pp. 323–334. ISSN: 2771-9529. DOI: 10.1109/TAFE.2024.3436470. URL: https://ieeexplore.ieee.org/document/10637440 (visited on 07/08/2025).

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay (2011). "Scikit-Learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12.85, pp. 2825–2830. ISSN: 1533-7928. URL: http://jmlr.org/papers/v12/pedregosa11a.html (visited on 01/31/2026).

Siméoni, Oriane, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski (Aug. 13, 2025). *DINOv3*. DOI: 10.48550/arXiv.2508.10104. arXiv: 2508.10104 [cs]. URL: http://arxiv.org/abs/2508.10104 (visited on 12/10/2025). Pre-published.

Skendžić, Sandra, Monika Zovko, Ivana Pajač Živković, Vinko Lešić, and Darija Lemić (2021). "The Impact of Climate Change on Agricultural Insect Pests". In: *Insects* 12.5. DOI: 10.3390/insects12050440. URL: https://www.proquest.com/docview/2532403367/abstract/8AF22CEBD71E4645PQ/1 (visited on 07/09/2025).

Stevens, Eli, Luca Antiga, and Thomas Viehmann (2020). *Deep Learning with PyTorch*. Shelter Island, NY: Manning Publications Co. 490 pp. ISBN: 978-1-61729-526-3.

Sun, Yiyou, Yifei Ming, Xiaojin Zhu, and Yixuan Li (June 28, 2022). "Out-of-Distribution Detection with Deep Nearest Neighbors". In: *Proceedings of the 39th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, pp. 20827–20840. URL: https://proceedings.mlr.press/v162/sun22d.html (visited on 06/10/2025).

Wirth, Christian, Helge Bruelheide, Nina Farwig, Jori Marx, and Josef Settele, eds. (2024a). *Faktencheck Artenvielfalt: Bestandsaufnahme und Perspektiven für den Erhalt der biologischen Vielfalt in Deutschland*. München: oekom verlag. 1256 pp. ISBN: 978-3-98726-095-7.

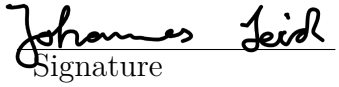
Wirth, Christian, Helge Bruelheide, Nina Farwig, Jori Maylin Marx, and Josef Settele, eds. (2024b). *Faktencheck Artenvielfalt: Bestandsaufnahme und Perspektiven für den Erhalt der biologischen Vielfalt in Deutschland: Zusammenfassung für die gesellschaftliche Entscheidungsfindung*. München: oekom verlag. 95 pp. ISBN: 978-3-98726-096-4.

Yang, Jingkang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu (Dec. 1, 2024). "Generalized Out-of-Distribution Detection: A Survey". In: *International Journal of Computer Vision* 132.12, pp. 5635–5662. ISSN: 1573-1405. DOI: 10.1007/s11263-024-02117-4. URL: https://doi.org/10.1007/s11263-024-02117-4 (visited on 04/25/2025).

Declaration of Authorship

Ich erkläre hiermit gemäß §9 Abs. 12 APO, dass ich die vorstehende Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Des Weiteren erkläre ich, dass die digitale Fassung der gedruckten Ausfertigung der Abschlussarbeit ausnahmslos in Inhalt und Wortlaut entspricht und zur Kenntnis genommen wurde, dass diese digitale Fassung einer durch Software unterstützten, anonymisierten Prüfung auf Plagiate unterzogen werden kann.

Bamberg, 12 February 2026
Place, Date


Signature