Deep Learning Pipeline for Automated Visual Moth Monitoring: Insect Localization and Species Classification

Dimitri Korsch [1], Paul Bodesheim [2], Joachim Denzler [3] [4] [5]

Abstract: Biodiversity monitoring is crucial for tracking and counteracting adverse trends in population fluctuations. However, automatic recognition systems are rarely applied so far, and experts evaluate the generated data masses manually. Especially the support of deep learning methods for visual monitoring is not yet established in biodiversity research, compared to other areas like advertising or entertainment. In this paper, we present a deep learning pipeline for analyzing images captured by a moth scanner, an automated visual monitoring system of moth species developed within the AMMOD project. We first localize individuals with a moth detector and afterward determine the species of detected insects with a classifier. Our detector achieves up to 99.01 % mean average precision and our classifier distinguishes 200 moth species with an accuracy of 93.13 % on image cutouts depicting single insects. Combining both in our pipeline improves the accuracy for species identification in images of the moth scanner from 79.62 % to 88.05 %.

Keywords: Biodiversity Monitoring; Deep Learning; Convolutional Neural Networks; Insect Detection; Species Classification; Unsupervised Part Estimation

1 Introduction

The discussions and concerns about environmental changes nowadays are both ubiquitous and necessary. We live in times in which ecosystems change drastically in a short time, and we, as humans, play a significant role in this development. One of these negative developments is the dramatic loss of insects [Ha17, Wa21]. One step towards a better understanding of insect die-off is monitoring species populations, which is time-consuming and often requires experts in that field. With about one million named species [St18], insects represent the vast majority of animals on our planet. Hence, it is also clear why manual counting of individuals for abundance estimations is not realistic. Though current developments in big data analysis and computer vision improved a lot, these technologies are not established in insect monitoring as they are, for example, in marketing or entertainment.

[1] Computer Vision Group, Friedrich Schiller University Jena, 07737 Jena, Germany; dimitri.korsch@uni-jena.de
[2] Computer Vision Group, Friedrich Schiller University Jena, 07737 Jena, Germany; paul.bodesheim@uni-jena.de
[3] Computer Vision Group, Friedrich Schiller University Jena, 07737 Jena, Germany; joachim.denzler@uni-jena.de
[4] German Aerospace Center (DLR), Institute for Data Science, Mälzerstraße 3, 07745 Jena, Germany
[5] Michael Stifel Center Jena for Data-Driven and Simulation Science, Ernst-Abbe-Platz 2, 07743 Jena, Germany

Besides others, there are monitoring systems to observe insects [JFR14, Sv20, Bj21], great apes [Fr16, Br17, Kä18, YMB19, SB20], elephants [KBD18, KD19], or sharks [HB17]. Though such monitoring systems are already camera-assisted, a vast amount of data needs to be evaluated. Unfortunately, the daily work of many ecologists nowadays is still the manual inspection of hundreds or thousands of images, which is exhausting and time-consuming.

As a part of the AMMOD project[6], we aim at automated monitoring of species in our immediate vicinity assisted by a computer vision system. In this paper, we cover the task of categorizing moth species (subset of *Lepidoptera*) by a non-invasive approach. Within the project, a so-called *moth scanner* is developed, which consists of an illuminated planar surface and an automated camera system. During the nighttime, special light sources illuminate the planar area to attract different moth species in the surrounding area. The automated camera system captures the attracted individuals that land on the illuminated surface. Finally, our task is to detect and classify the individuals in the taken images. With the detection and classification results, we assist the ecologists in analyzing the insect population trends.

For automatically analyzing the images, we propose a prototype for a deep learning pipeline consisting of two steps: (1) localization of individuals via moth detection and (2) species identification by classification. For the detection, we use a well-established detection model capable of identifying multiple objects in an image, namely the single-shot MultiBox detector (SSD) [Li16]. The mean average precision (mAP) of our moth detector is 88.88 % and 99.01 % for intersection over union (IoU) values above 0.75 and 0.5, respectively.

The subsequent classification of 200 common moth species in Central Europe is performed with the help of a convolutional neural network (CNN). In our experiments, we show the benefits of different design decisions for a classifier trained on copped images, namely images depicting a single insect. The selection of a fine-tuning strategy, the pre-training dataset, and the extension of the classifier with an unsupervised part estimator improve the classification accuracy from 63.28 % to 93.13 %.

Finally, we show that the classification accuracy of our proposed pipeline improves with a preceding moth detector on uncropped images. These are the images captured by the moth scanner, where we cannot ensure that only a single insect has been photographed. Furthermore, in these images, the insects allocate only a small portion of the entire area. Hence, preceding a moth classifier with a moth detector, we can improve the classification performance from 79.62 % to 88.05 %.

[6] AMMOD = **A**utomated **M**ultisensor Station for **M**onitoring **o**f Bio**d**iversity (`https://ammod.de/`)

2 Related Work

2.1 Insect Monitoring

In general, a commonly used method for monitoring insects is the usage of light traps. Jonason et al. [JFR14] presented a survey on the influence of weather, time of the year, and the type of the light source on the richness and abundance of species. While the authors identified the moth species manually, some of the first automated species identification systems were presented by Watson et al. [WOK04], Mayo and Watson [MW07], and Batista et al. [BCK10]. All of these works used the same dataset, namely 35 species with 20 individuals per species. Using support vector machines (SVMs) and nearest neighbor classifiers, they report an accuracy of up to 85 % [MW07] with leave-one-out cross-validation. Ding and Taylor [DT16] presented automated detection and classification of insect pests. They used a sliding window approach coupled with a CNN model. The CNN, followed by a non-maximum suppression as post-processing, performed a binary classification to identify a *codling moth* in the windows. They achieved an area under the precision-recall curve of 0.93. In contrast to these works, we perform the classification of much more classes, namely 200 moth species.

The works of Chang et al. [Ch17] and Xia et al. [Xi18] tackled more challenging classification tasks. Using images from the Internet, they classified 636 and 24 species, respectively. Chang et al. achieved for the 450 butterflies and 186 moths species an accuracy of 71.5 % with a ResNet-18 [He16] architecture. Xia et al. performed a joint detection and classification of individual insects and achieved with their variant of a VGG-19 CNN a mean average precision (mAP) of 89.22 %. First, we gather images in a more controlled environment. As a result, the background is more homogeneous, and the moths are photographed from above in a resting position. It is worth investigating how far the Internet images that do not represent our desired setup domain may enhance the classification performance. Anyway, this is out of the scope of this paper. Furthermore, unlike Xia et al., we aim to separate the detection and classification tasks since they will be performed on different physical devices in our setting.

Zhong et al. [Zh18] and Bjerge et al. [Bj21] presented detection and classification pipelines deployed on embedded systems, namely on Raspberry Pi variants. While Zhong et al. used the YOLO framework [Re16] for moth detection, Bjerge et al. presented a detection-by-thresholding approach. Zhong et al. achieved a classification accuracy of 90.2 % for six species with an SVM and shallow features (texture, shape, color, and HOG features). Bjerge et al. presented their own CNN architecture and report an F1-score of 93.00 % for the classification of nine classes. The authors perform additional counting and tracking of the insects, which is not part of this work. Furthermore, we outperform the classification results presented by Bjerge et al. in our experiments (Sect. 4.2).

2.2 Object Detection

Pre-CNN image-based object detection was dominated by Deformable Part Model (DPM) [FMR08] and Selective Search [Ui13]. The first approach uses a sliding window approach, whereas the latter uses region proposal selection as an object detection strategy. After the rise of CNNs, region proposal methods are dominating the object detection research field. One of the first was the R-CNN [Gi14] that combined selective search region proposals with a CNN-based classification of these regions. Many improvement and adaptations based on R-CNN were developed: SPPNet [He15], Fast R-CNN [Gi15], MultiBox [Er14], or Faster R-CNN [Re15]. Some of them improved the classification of the region proposals in quality and computation time [He15, Gi15]. Others improved the quality of the region proposals directly [Er14, Re15], especially with an integration of a region proposal CNN. Some of the methods skip the proposal step and predict bounding boxes directly with the confidences for multiple categories. Most popular examples are YOLO [Re16], OverFeat [Se13], and SSD [Li16]. While OverFeat implements a deep version of the sliding window approach, YOLO uses CNN features to predict bounding boxes and categories. SSD extracts features from multiple feature maps from multiple stages in the CNN and predicts bounding boxes based on a set of prior locations.

All of them have their advantages and disadvantages. Meanwhile, there are also dozens of adaptations and improvements to these methods. Nevertheless, in our work, we use the single-shot MultiBox detector (SSD) since it allows an exchange of the underlying backbone network and yields one of the best results on standard object detection benchmarks like Pascal VOC [Ev07, Ev12] and MS COCO [Li14].

2.3 Fine-grained Classification

Fine-grained classification is a special classification task, where the categories, which need to be classified, originate from the same object domain (e.g., bird species [Wa11], car models [Kr13], moth species [Ro15], or elephant individuals [KD19]). The challenge is now to distinguish closely related classes that differ only in subtle features. In the age of CNNs, it is common to use the data and let the network figure out what are relevant visual features that distinguish a class from the others. This kind of approach utilizes the input image as it is and performs either smart pre-training strategies [Cu18, Kr16] or advanced feature aggregation techniques [LRM15, Si18]. On the other hand, there are the part- or attention-based approaches [GLY19, HPZ19, Zh19b] that extract relevant regions already at the pixel level and use cropped image regions as additional features for the classification. Both classification strategies have their advantages and drawbacks. We use an unsupervised approach for part estimation proposed by Korsch et al. [KBD19] within our pipeline.

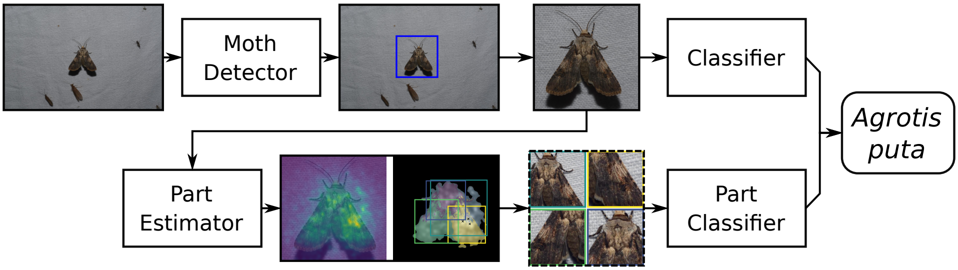


Fig. 1: Our deep learning pipeline for automatically analyzing images of the moth scanner: in the first step we use a moth detector to detect an individual in an image. On the one hand, we use this cropped image for the first species prediction. On the other hand, we estimate additional information in form of parts and perform a second part-based prediction. Finally, both predictions are combined to obtain the final species classification.

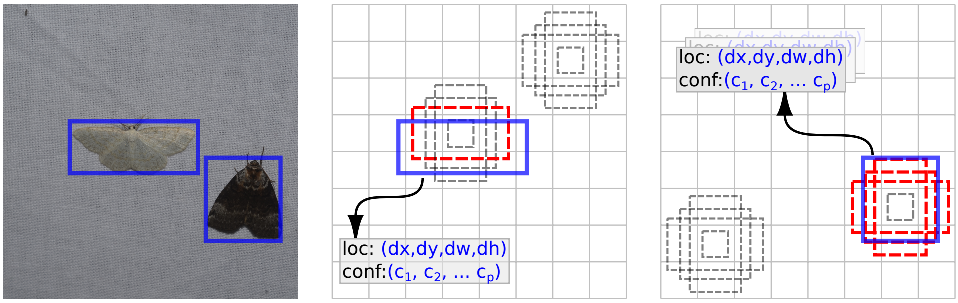
3 Methods

In this paper, we introduce an automated pipeline for moth species detection and classification. As visualized in Figure 1, the moth detector identifies bounding boxes around the insects given an input image. Afterward, the image patches identified by the detected bounding boxes are fed into a CNN classifier. The pipeline performs the classification either only on the input image or extracts informative regions, called parts, which it uses as additional information. In our experiments, we show that this additional information improves the classification performance (see Sect. 4.3).

In the following, we introduce the two stages of the pipeline: (1) moth detection based on the single-shot detector (Sect. 3.1), and (2) part-based classification with the help of classification-specific parts (Sect. 3.2).

3.1 Single-shot Detector

As already mentioned in Sect. 2.2, the main idea of the single-shot MultiBox detector (SSD) proposed by Liu et al. [Li16] is to utilize feature maps from multiple intermediate stages of the backbone CNN to predict location offsets and class confidences for a set of prior locations. More precisely, given a feature map with $F \in \mathbb{R}^{N \times M \times P}$ with P channels, K prior bounding boxes with different scales and aspect ratios are defined for each of the $N \cdot M$ locations. The feature map is transformed by a 3×3 convolution with $(C + 4) \cdot K$ output channels. This results for each of the K prior boxes in C per-class scores and four offset values $\Delta = \{dx, dy, dw, dh\}$. The offsets dx and dy describe the positional offset to the center of the prior box. The change of the width and the height of a prior box is modeled by dw and dh.



(a) Example image with ground truth annotations.

(b) For the first moth a single prior box is selected.

(c) For the second moth multiple prior boxes are selected.

Fig. 2: Example of SSD prior boxes (*dashed gray*) for an 8×8 feature map. The prior boxes have different scales and aspect ratios, which allows for detecting objects of various sizes and orientations. Different ground truth bounding boxes (*solid blue*) may be matched to a single (Figure 2b) or multiple (Figure 2c) prior boxes. For each selected prior box (*dashed red*) the offset values and the class scores are estimated. (*Similar to Figure 1 in [Li16].)*

During training, a prior box is selected as positive when there was a ground truth bounding box with an IoU score higher than 0.5. As visualized in Figure 2, prior boxes at different locations are assigned to different objects. Additionally, for a single object multiple prior boxes may be selected. The training objective for estimated location offsets Δ and class scores c given ground truth bounding boxes g is defined as a sum of the confidence loss and the localization loss:

$$L(\Delta, c, g) = \frac{1}{\mathcal{K}} \left(L_{conf}(c) + L_{loc}(\Delta, g) \right) \tag{1}$$

with $\mathcal{K}$ being the number of matched prior boxes, and if no boxes are matched, the loss is set to 0. The class scores are unnormalized log-likelihoods of a class identified in a certain location. In our case, the detector distinguishes only the general *moth* class from the background. For more details about the loss functions, we refer to the original paper of Liu et al. [Li16].

3.2 Part-based Classification

Nowadays, neural networks like CNNs yield the best results in classification by extracting high-level features from the input image in the form of a high-dimensional feature vector (e.g., $D = 2048$ in case of InceptionV3). In the context of a fine-grained recognition task, the classifier has to focus on a specific feature dimension to distinguish a class from the others. Therefore, we first estimate the most informative features for the current classification task. It is realized by utilizing a linear classifier with a sparsity-inducing

L1-regularization. An optimization with L1-regularization forces the classifier's decisions to perform the classification on a small subset of feature dimensions. This kind of implicit feature selection is classification-specific. Furthermore, it allows identifying for each class the feature dimensions that best distinguish this class from all other classes by selecting dimensions with classifier weights above a certain threshold.

Informative Image Regions: We utilize gradient maps [SVZ13] to estimate the most informative pixels in the image, identified by large gradients. As described previously, we restrict the computation of the gradients only to the feature dimensions used by the L1-regularized classifier. Thus, we incorporate the initial classification in the estimation of the part regions. Like Simonyan et al. [SVZ13] and Simon et al. [SR15], we use back-propagation through the CNN to identify the regions of interest for each selected feature dimension. We compute a saliency map $\vec{M}(\vec{I})$ for an image $\vec{I}$ based on the feature dimension subset $\mathfrak{D} \subset \{1, \dots, D\}$ as follows:

$$M_{x,y}(\vec{I}) = \frac{1}{|\mathfrak{D}|} \sum_{d \in \mathfrak{D}} \left| \frac{\partial}{\partial I_{x,y}} f^{(d)}(\vec{I}) \right| \quad . \tag{2}$$

Part Estimation: Next, we normalize the values of the saliency map to the range $[0 \dots 1]$, and discard regions of low saliency by setting values beneath the mean saliency value to 0. We use the resulting sparse saliency map to estimate the spatial extent of coherent regions. Like Zhang et al. [Zh19a], we achieve this by k-means clustering of pixel coordinates (x, y) and the saliencies $M_{x,y}$ (Eq. 2). Additionally, we also consider the RGB values at the corresponding positions in the input image. The clusters are initialized with k peaks computed by non-maximum suppression, identifying locations with the largest saliencies. Consequently, the number of peaks determines the number of parts to detect. Finally, it is straightforward to identify a bounding box around each estimated cluster, and the resulting bounding boxes serve as parts for the following classification.

Extraction and Aggregation of Part Features: In the final step, we extract image patches with the help of the estimated bounding boxes and treat them as regular images. The neural network should extract different features from these image patches than from the original image because the level of detail varies between these types of input. Therefore, we process the part images by the same CNN architecture as the original image but with a separate set of weights. Afterward, for every part image, the CNN extracts a feature vector, resulting in a set of part features for every single image. There are different ways to aggregate these features to a single feature vector and perform the classification. We have chosen to average over the part features, which results in a single feature vector with the same dimension as for the original image. This aggregation strategy yielded better results in our experiments than, for example, concatenation of part features. Finally, classification is performed based on the global feature and the aggregated part feature. For joint optimization of both CNNs, we average the cross-entropy losses of the global prediction $\vec{p}$ and part prediction $\vec{q}$. It equals to computing the geometrical mean of normalized class probabilities and enforces both classifiers to be certain about the correct class:

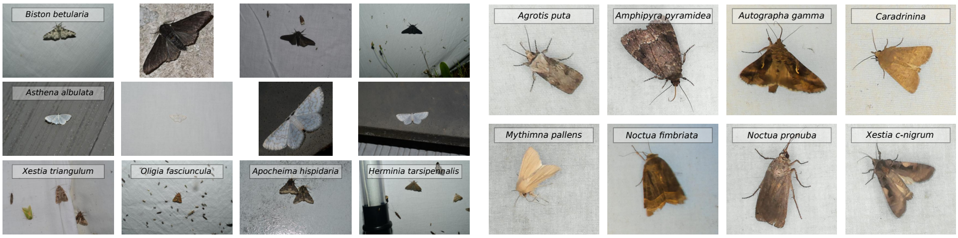
$$\begin{aligned} L_{final}\left(\{\vec{p},\vec{q}\},y\right) &= \frac{1}{2}\left(L\left(\vec{p},y\right)+L\left(\vec{q},y\right)\right) && (3)\\ &= -\frac{1}{2}\left(\sum_{i=1}^{C} y_i \cdot \log(p_i) + \sum_{i=1}^{C} y_i \cdot \log(q_i)\right) && (4)\\ &= -\sum_{i=1}^{C} y_i \cdot \log\left(\sqrt{p_i \cdot q_i}\right) \quad . && (5) \end{aligned}$$

4 Experiments

In the following, we evaluate the two parts of our moth scanner pipeline. We perform each experiment ten times and provide in Sections 4.2, and 4.3 the mean and the standard deviation of the evaluation metrics across the different runs. We fine-tune all CNNs for 60 epochs with the RMSProp [TH12] optimizer and L2-regularization with a weight decay of 5×10^{-4}. The starting learning rate of 1×10^{-4} is reduced by 0.1 after 20 and 40 epochs. Furthermore, we utilize standard image augmentation methods: random cropping, random horizontal and vertical flipping, and color jittering (contrast, brightness, and saturation). In the case of the classification, we further utilize label smoothing regularization [Sz16] with a smoothing factor of 0.1.

First, we evaluate the performance of the moth detector presented in Sect. 3.1. In Table 2, we report the mean average precision (mAP) as the evaluation metric for the detections. The precision is based on predictions of detected objects, where the intersection over union (IoU) of predicted and ground truth bounding boxes is above a certain threshold. The IoU describes how well two bounding boxes match by computing the ratio between the intersection and the union of the areas of the two bounding boxes. The two typical evaluation metrics used in one of the most common object detection benchmarks [Li14] are *mAP@0.5* and *mAP@0.75*, with IoU thresholds of 0.5 and 0.75, respectively. We use a standard version of the detection network with an input size of 300 × 300 and the VGG16 [SZ14] architecture as a backbone that is pre-trained on ImageNet [Ru15]. All additional layers included for the detection task and not initially present in the VGG architecture are initialized randomly.

Second, we evaluate the classification performance. Since the datasets we use have a balanced sample distribution across the classes, we use accuracy as a primary evaluation metric. To be independent of the underlying detector, we use the ground truth bounding boxes and perform the classification on the cropped image patches. Further, we extract additional parts, as described in Sect. 3.2, and combine the predictions on these parts with the predictions on the entire image. For the classification, we use the InceptionV3 CNN



(a) The first two rows show two different moths species, whereas the third row shows images with more than one insect. These examples illustrate the versatility in the appearance of the moths.

(b) Cropped images of the eight MCC classes. Bjerge et al. [Bj21] used an eight-megapixel web camera to capture the images. Hence, the details of the species are barely visible.

Fig. 3: Example images from the *EU-Moths* (a) and *MCC* (b) datasets.

architecture [Sz16]. Here, we also decided on the default input size of 299 × 299 for both the images and parts. Furthermore, we investigate the effect of two different pre-training methods. As the typical choice, we use ImageNet [Ru15] pre-training since most of the deep learning frameworks implementing different CNN architectures also provide weights for these architectures pre-trained on the ImageNet dataset. Additionally, we use pre-training on the iNaturalist [Va18] dataset provided by Cui et al. [Cu18]. Data used in this kind of pre-training is more related to the domain of animals, in our case to the domain of insects, which can also be seen in the improvement of the classification accuracy in Table 3.

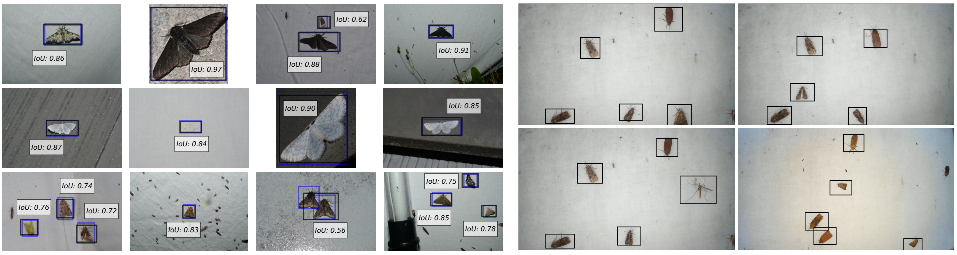
Finally, we evaluate the proposed pipeline as a whole: given an uncropped image, like in Figure 3a, we first apply the moth detector and then perform the classification on the resulting image patches. We compare this setup to a classifier that performs the classification on the initial uncropped images.

We further evaluate the classifier on the dataset provided by Bjerge et al. [Bj21]. Unfortunately, the authors report only F1-scores of their classification and do not provide their training-test split. Nevertheless, we performed five-fold cross-validation with the same train-test ratios as in the original paper. For each of the folds, we repeated the training ten times, like in the previous experiments.

4.1 Datasets

Moth Classification and Counting (MCC) Dataset[7]: Created by Bjerge et al. [Bj21], the subset for the classification consists of eight moth species with 250 images for each species, resulting in a dataset of 2000 images. The authors used an 80:20 training-test split of the data but did not provide their specific split. Additionally, there are two more classes: a background class and a class for a wasp species. We ignore these classes and perform

[7] `https://github.com/kimbjerge/MCC-trap`



(a) Our manual ground truth annotations are visualized in *blue*. Text boxes contain the IoU of each detection with the ground truth.

(b) The MCC dataset does not provide any ground truth annotations. Hence, only the detections are visualized.

Fig. 4: Visualized detection results (*black* bounding boxes) on the *EU-Moths* (a) and *MCC* (b) datasets.

the training and the evaluation only on the eight moths species. An individual from every class is shown in Figure 3b. Compared to the EU-Moths dataset, the images are of lower quality since the authors captured them with an eight-megapixel web camera connected to a Raspberry Pi 4.

European Moths (EU-Moths) Dataset[8]: From roughly 4700 moth species present in Central Europe[9], this dataset consists of 200 species most common in this region. Each of the species is represented by approximately 11 images. We considered a random but balanced training-test split in eight training and three test images per species, resulting in roughly 1600 training and 600 test images in total. To evaluate the detector, we manually annotated the bounding boxes around the insects. Some examples of the images are shown in Figure 3a. The insects are photographed manually and mainly on a relatively homogeneous background. About 92 % of the images contain only a single individual like it is shown in the first two rows of Figure 3a. The last row of the same image depicts images with more than one insect of interest. This scenario may happen in the final moth scanner installation, and it is crucial to test how the detector performs in this case.

This dataset yields different challenges for the detector and the classifier. On the one hand, the detector should be able to detect insects of different sizes. Furthermore, we require a detector with MultiBox support. Both of these properties are fulfilled by the SSD. On the other hand, designing a classifier that can predict species from these raw images is difficult. As proposed in this paper, we decided to use a moth detector to locate single insects and classify these separately. Hence, the classifier is trained on images cropped to the bounding boxes identifying a single individual.

	F1-Score	Accuracy	
		ImageNet	iNaturalist
Bjerge et al. [Bj21]	93.00 %	–	–
Ours	99.69 % (±0.34 %)	99.41 % (±0.79 %)	99.55 % (±0.40 %)

Tab. 1: Comparison of the classification results on cropped images (see Figure 3b) of the MCC dataset provided by Bjerge et al. [Bj21]. Besides the F1-Score that was reported by the authors in their work, we report additionally the accuracy for our trained classifiers.

4.2 Results on the MCC Dataset

Detection: Unfortunately, the authors do not provide any bounding box annotations. Hence, we were not able to evaluate the performance of our moth detector on this dataset. Nevertheless, we provide a qualitative evaluation on some of the images in Figure 4b.

Classification: In Table 1, we compare our classification method with the one proposed by Bjerge et al. [Bj21] on the MCC dataset. One can see that our classifier achieves near-perfect accuracies and F1-scores. We assume the reason for these results is in the composition of the dataset. Since Bjerge et al. do not provide any training-test split, we have used a random split with the same ratio (80:20) as the authors in their work. Nevertheless, one can see in Figure 4b some of the moth individuals do not move in different images captured in short intervals. As a result, extracting image crops from these images would result in near-identical images in different splits after the random splitting. Consequently, one would train and test the classifier on nearly the same data. Nevertheless, to be comparable to the results of Bjerge et al., we chose the same splitting strategy, even though it may not represent the correct evaluation of the model.

4.3 Results on the EU-Moths Dataset

Detection: We split this experiment into two parts: (1) we evaluated the detection performance on the entire dataset, and (2) we split the dataset into two subsets of distinct classes. The first part evaluates the standard performance of the detector. The second part of the experiment evaluates how good the detector performs on classes not seen during the training. This scenario is essential since not all species may be available at training time in the real-world setup. One could train the detector on a dataset captured at one location and deploy it at another one. Furthermore, a detector able to localize moth species not seen at

[8] https://www.inf-cv.uni-jena.de/eu_moths_dataset

[9] http://lepiforum.org/ (accessed on 6th July, 2021)

	mAP@0.75	mAP@0.5
Evaluated on	Trained on entire dataset	
Entire dataset	88.88 % (±0.77 %)	99.01 % (±0.09 %)
Evaluated on	Trained on subset 1	
Subset 1	87.38 % (±1.65 %)	98.53 % (±0.13 %)
Subset 2	78.83 % (±1.11 %)	99.19 % (±0.28 %)
Evaluated on	Trained on subset 2	
Subset 1	85.88 % (±1.04 %)	99.70 % (±0.10 %)
Subset 2	82.45 % (±1.23 %)	98.04 % (±0.29 %)

Tab. 2: Detection results on the EU-Moths dataset. First row contains the evaluation of the detectors trained on the entire dataset (200 classes). The lower part of the table shows the capability of the detector to localize unseen species. For that, we split the dataset in two parts with distinct classes (subset 1: classes 1 to 100, and subset 2: classes 101 to 200) and perform cross-subset evaluations.

training time is beneficial for novelty detection, active learning, and incremental learning algorithms [Br17, BKD20].

Table 2 shows the detection results for both experiments. As previously mentioned, we report the mean average precision for IoU thresholds 0.75 and 0.5 (mAP@0.75 and mAP@0.5). The first row shows the results for the first experiment. The detector seems to be quite precise if we consider the challenges of the dataset. The lower part of Table 2 further shows the cross-subset results. Here we can see that the mAP@0.75 performance drops compared to the previous experiment, and the standard deviation increases. Both are explainable because, for the second experiment, we used only half of the classes for the training. Furthermore, mAP@0.5 performance remains comparable to the first experiment, which shows the moth detector's reliability for unseen classes.

Additionally, Figure 4 depicts qualitative results of the detector. In Figure 4a, we estimated the bounding boxes (in *black*) for some of the images of the EU-Moths dataset. We visualized the ground truth bounding boxes (in *blue*) and the resulting IoU between the prediction and the ground truth. The detector's most significant challenge seems to be insects located too close to each other (second last example in the final row).

Classification: Table 3 shows the results of the classification. We compare different training and pre-training methods and whether the additional information in the form of parts benefits the classification accuracy.

First, one can see that fine-tuning the entire CNN instead of using it only as feature extractor results in an improvement of the recognition rate by roughly 26 % and 4 % for CNNs pre-trained on ImageNet [Ru15] and iNaturalist 2017 [Va18] datasets, respectively.

		Accuracy	
	Fine-tuning	ImageNet	iNaturalist
No Parts	*only FC layer*	63.28 % (±0.45 %)	86.60 % (±0.42 %)
	entire CNN	89.46 % (±0.88 %)	90.54 % (±1.10 %)
With Parts	*only FC layer*	71.82 % (±0.35 %)	87.96 % (±0.38 %)
	entire CNN	91.50 % (±0.61 %)	93.13 % (±0.76 %)

Tab. 3: Classification results on cropped images of the EU-Moths dataset. The results show the effects of the different fine-tuning strategies, the two pre-training datasets, and the usage of additional information in the form of parts.

Composition	Accuracy
Classifier Only	79.62 % (±1.10 %)
Detector + Classifier	88.05 % (±0.58 %)

Tab. 4: Classification results on uncropped images as shown in Figure 3a. The effect of a preceding detector on the classification accuracy is clearly visible.

Though training the entire CNN results in longer training times and is computationally more expensive, the improvements are visible.

Second, the choice of the pre-training dataset is also crucial. Replacing the typical CNN weights provided by almost every deep learning framework pre-trained on the ImageNet dataset with the ones proposed by Cui et al. [Cu18] leads to a further improvement. The later pre-training increases the accuracy by approximately 1 % if the entire CNN is trained. It also yields remarkable benefits if choosing computationally cheaper training of only the final classification layer, namely an improvement of 20 %.

Finally, employing additional information in the form of parts improves the classification accuracies by approximately 2 to 2.6 % depending on the chosen pre-training. We achieved the best results with the part-based setup: 91.50 % and 93.13 % with ImagenNet and iNaturalist pre-training, respectively.

Entire Pipeline: In this experiment, we evaluate the entire pipeline as presented in Sect. 3. For this purpose, we couple every trained detector with every trained classifier and observe the resulting classification performance. This way, we report the mean accuracy of 100 detector-classifier combinations in the last row of Table 4. As a baseline method, we trained ten CNN classifiers on the original uncropped images. One can see that the preceding detector improves the classification accuracy by approximately 9 %.

5 Conclusions

In this paper, we presented an automated detection and classification pipeline for 200 moth species. We plan to deploy this pipeline in the visual monitoring system of the AMMOD project, the so-called moth scanner, which will help ecologists observe the population trends of the insects. Since light sources easily attract moths, the moth scanner consists of an illuminated white surface and a camera that automatically captures images of insects resting on this surface. We first localized the moths with a single-shot MultiBox detector (SSD) in the recorded images and then classified the resulting detections using a CNN classifier. We also showed the effect of different training configurations on the final classification accuracy: the choice of fine-tuning strategy, the selection of the pre-training dataset, and the extension of the classification with an unsupervised part estimator. In our experiments, each part of the pipeline achieved promising results: a detection rate of up to 99.01 % (mAP@50) and classification accuracy on images depicting a single insect of up to 93.13 %. Finally, we evaluated both parts of the pipeline together and improved the classification accuracy on original images captured by the moth scanner from 79.62 % to 88.05 % compared to a setup without a preceding moth detector.

Acknowledgments

This work has been funded by the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung, BMBF, Deutschland) via the project “Development of an Automated Multisensor Station for Monitoring of Biodiversity (AMMOD) - Subproject 5: Automated Visual Monitoring and Analysis” (FKZ: 01LC1903E).

References

[BCK10] Batista, Gustavo EAPA; Campana, Bilson; Keogh, Eamonn: Classification of live moths combining texture, color and shape primitives. In: 2010 Ninth International Conference on Machine Learning and Applications. IEEE, pp. 903–906, 2010.

[Bj21] Bjerge, Kim; Nielsen, Jakob Bonde; Sepstrup, Martin Videbaek; Helsing-Nielsen, Flemming; Høye, Toke Thomas: An automated light trap to monitor moths (Lepidoptera) using computer vision-based tracking and deep learning. Sensors, 21(2):343, 2021.

[BKD20] Brust, Clemens-Alexander; Käding, Christoph; Denzler, Joachim: Active and Incremental Learning with Weak Supervision. Künstliche Intelligenz (KI), 2020.

[Br17] Brust, Clemens-Alexander; Burghardt, Tilo; Groenenberg, Milou; Käding, Christoph; Kühl, Hjalmar; Manguette, Marie; Denzler, Joachim: Towards Automated Visual Monitoring of Individual Gorillas in the Wild. In: ICCV Workshop on Visual Wildlife Monitoring (ICCV-WS). pp. 2820–2830, 2017.

[Ch17] Chang, Qi; Qu, Hui; Wu, Pengxiang; Yi, Jingru: Fine-Grained Butterfly and Moth Classification Using Deep Convolutional Neural Networks. Machine Learning course project report, submitted to the Department of Computer Science, Rutgers University, 2017.

[Cu18] Cui, Yin; Song, Yang; Sun, Chen; Howard, Andrew; Belongie, Serge: Large Scale Fine-Grained Categorization and Domain-Specific Transfer Learning. In: Proceedings of CVPR. 6 2018.

[DT16] Ding, Weiguang; Taylor, Graham: Automatic moth detection from trap images for pest management. Computers and Electronics in Agriculture, 123:17–28, 2016.

[Er14] Erhan, Dumitru; Szegedy, Christian; Toshev, Alexander; Anguelov, Dragomir: Scalable object detection using deep neural networks. In: Proceedings of CVPR. pp. 2147–2154, 2014.

[Ev07] Everingham, Mark; Van Gool, Luc; Williams, Christopher K.I.; Winn, John; Zisserman, Andrew: The PASCAL visual object classes challenge 2007 (VOC2007) results. 2007.

[Ev12] Everingham, Mark; Van Gool, Luc; Williams, Christopher K.I.; Winn, John; Zisserman, Andrew: The PASCAL visual object classes challenge 2012 (VOC2012) results. 2012.

[FMR08] Felzenszwalb, Pedro; McAllester, David; Ramanan, Deva: A discriminatively trained, multiscale, deformable part model. In: Proceedings of CVPR. IEEE, pp. 1–8, 2008.

[Fr16] Freytag, Alexander; Rodner, Erik; Simon, Marcel; Loos, Alexander; Kühl, Hjalmar; Denzler, Joachim: Chimpanzee Faces in the Wild: Log-Euclidean CNNs for Predicting Identities and Attributes of Primates. In: German Conference on Pattern Recognition (GCPR). pp. 51–63, 2016.

[Gi14] Girshick, Ross; Donahue, Jeff; Darrell, Trevor; Malik, Jitendra: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of CVPR. pp. 580–587, 2014.

[Gi15] Girshick, Ross: Fast R-CNN. In: Proceedings of ICCV. pp. 1440–1448, 2015.

[GLY19] Ge, Weifeng; Lin, Xiangru; Yu, Yizhou: Weakly supervised complementary parts models for fine-grained image classification from the bottom up. In: Proceedings of CVPR. pp. 3034–3043, 2019.

[Ha17] Hallmann, Caspar A; Sorg, Martin; Jongejans, Eelke; Siepel, Henk; Hofland, Nick; Schwan, Heinz; Stenmans, Werner; Müller, Andreas; Sumser, Hubert; Hörren, Thomas et al.: More than 75 percent decline over 27 years in total flying insect biomass in protected areas. PloS one, 12(10):e0185809, 2017.

[HB17] Hughes, Benjamin; Burghardt, Tilo: Automated visual fin identification of individual great white sharks. International Journal of Computer Vision, 122(3):542–557, 2017.

[He15] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian: Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, 37(9):1904–1916, 2015.

[He16] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778, 2016.

[HPZ19] He, Xiangteng; Peng, Yuxin; Zhao, Junjie: Which and How Many Regions to Gaze: Focus Discriminative Regions for Fine-Grained Visual Categorization. IJCV, pp. 1–21, 2019.

[JFR14] Jonason, Dennis; Franzén, Markus; Ranius, Thomas: Surveying Moths Using Light Traps: Effects of Weather and Time of Year. PLOS ONE, 9(3):1–7, 03 2014.

[Kä18] Käding, Christoph; Rodner, Erik; Freytag, Alexander; Mothes, Oliver; Barz, Björn; Denzler, Joachim: Active Learning for Regression Tasks with Expected Model Output Changes. In: British Machine Vision Conference (BMVC). 2018.

[KBD18] Körschens, Matthias; Barz, Björn; Denzler, Joachim: Towards Automatic Identification of Elephants in the Wild. In: AI for Wildlife Conservation Workshop (AIWC). 2018.

[KBD19] Korsch, Dimitri; Bodesheim, Paul; Denzler, Joachim: Classification-Specific Parts for Improving Fine-Grained Visual Categorization. In: Proceedings of the German Conference on Pattern Recognition. pp. 62–75, 2019.

[KD19] Körschens, Matthias; Denzler, Joachim: ELPephants: A Fine-Grained Dataset for Elephant Re-Identification. In: ICCV Workshop on Computer Vision for Wildlife Conservation (ICCV-WS). 2019.

[Kr13] Krause, Jonathan; Stark, Michael; Deng, Jia; Fei-Fei, Li: 3D Object Representations for Fine-Grained Categorization. In: 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13). 2013.

[Kr16] Krause, Jonathan; Sapp, Benjamin; Howard, Andrew; Zhou, Howard; Toshev, Alexander; Duerig, Tom; Philbin, James; Fei-Fei, Li: The unreasonable effectiveness of noisy data for fine-grained recognition. In: ECCV. Springer, pp. 301–320, 2016.

[Li14] Lin, Tsung-Yi; Maire, Michael; Belongie, Serge; Hays, James; Perona, Pietro; Ramanan, Deva; Dollár, Piotr; Zitnick, C Lawrence: Microsoft COCO: Common objects in context. In: ECCV. Springer, pp. 740–755, 2014.

[Li16] Liu, Wei; Anguelov, Dragomir; Erhan, Dumitru; Szegedy, Christian; Reed, Scott; Fu, Cheng-Yang; Berg, Alexander C: SSD: Single shot multibox detector. In: ECCV. Springer, pp. 21–37, 2016.

[LRM15] Lin, Tsung-Yu; RoyChowdhury, Aruni; Maji, Subhransu: Bilinear cnn models for fine-grained visual recognition. In: Proceedings of ICCV. pp. 1449–1457, 2015.

[MW07] Mayo, Michael; Watson, Anna T: Automatic species identification of live moths. Knowledge-Based Systems, 20(2):195–202, 2007.

[Re15] Ren, Shaoqing; He, Kaiming; Girshick, Ross; Sun, Jian: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS. pp. 91–99, 2015.

[Re16] Redmon, Joseph; Divvala, Santosh; Girshick, Ross; Farhadi, Ali: You only look once: Unified, real-time object detection. In: Proceedings of CVPR. pp. 779–788, 2016.

[Ro15] Rodner, Erik; Simon, Marcel; Brehm, Gunnar; Pietsch, Stephanie; Wägele, J. Wolfgang; Denzler, Joachim: Fine-grained Recognition Datasets for Biodiversity Analysis. In: CVPR Workshop on Fine-grained Visual Classification (CVPR-WS). 2015.

[Ru15] Russakovsky, Olga; Deng, Jia; Su, Hao; Krause, Jonathan; Satheesh, Sanjeev; Ma, Sean; Huang, Zhiheng; Karpathy, Andrej; Khosla, Aditya; Bernstein, Michael et al.: Imagenet large scale visual recognition challenge. International journal of computer vision, 115(3):211–252, 2015.

[SB20] Sakib, Faizaan; Burghardt, Tilo: Visual Recognition of Great Ape Behaviours in the Wild. arXiv preprint arXiv:2011.10759, 2020.

[Se13] Sermanet, Pierre; Eigen, David; Zhang, Xiang; Mathieu, Michaël; Fergus, Rob; LeCun, Yann: Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229, 2013.

[Si18] Simon, Marcel; Rodner, Erik; Darell, Trevor; Denzler, Joachim: The whole is more than its parts? From explicit to implicit pose normalization. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1–13, 2018.

[SR15] Simon, Marcel; Rodner, Erik: Neural Activation Constellations: Unsupervised Part Model Discovery With Convolutional Networks. In: The IEEE International Conference on Computer Vision (ICCV). December 2015.

[St18] Stork, Nigel E: How many species of insects and other terrestrial arthropods are there on Earth? Annual review of entomology, 63:31–45, 2018.

[Sv20] Svenningsen, Cecilie S; Bowler, Diana E; Hecker, Susanne; Bladt, Jesper; Grescho, Volker; van Dam, Nicole M; Dauber, Jens; Eichenberg, David; Ejrnæs, Rasmus; Fløjgaard, Camilla et al.: Contrasting impacts of urban and farmland cover on flying insect biomass. bioRxiv, 2020.

[SVZ13] Simonyan, Karen; Vedaldi, Andrea; Zisserman, Andrew: Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034, 2013.

[SZ14] Simonyan, Karen; Zisserman, Andrew: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[Sz16] Szegedy, Christian; Vanhoucke, Vincent; Ioffe, Sergey; Shlens, Jon; Wojna, Zbigniew: Rethinking the Inception Architecture for Computer Vision. In: Proceedings of CVPR. June 2016.

[TH12] Tieleman, Tijmen; Hinton, Geoffrey: Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, 4(2):26–31, 2012.

[Ui13] Uijlings, Jasper RR; Van De Sande, Koen EA; Gevers, Theo; Smeulders, Arnold WM: Selective search for object recognition. IJCV, 104(2):154–171, 2013.

[Va18] Van Horn, Grant; Mac Aodha, Oisin; Song, Yang; Cui, Yin; Sun, Chen; Shepard, Alex; Adam, Hartwig; Perona, Pietro; Belongie, Serge: The iNaturalist species classification and detection dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8769–8778, 2018.

[Wa11] Wah, C.; Branson, S.; Welinder, P.; Perona, P.; Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[Wa21] Wagner, David L; Grames, Eliza M; Forister, Matthew L; Berenbaum, May R; Stopak, David: Insect decline in the Anthropocene: Death by a thousand cuts. Proceedings of the National Academy of Sciences, 118(2), 2021.

[WOK04] Watson, Anna T; O'Neill, Mark A; Kitching, Ian J: Automated identification of live moths (Macrolepidoptera) using digital automated identification System (DAISY). Systematics and Biodiversity, 1(3):287–300, 2004.

[Xi18] Xia, Denan; Chen, Peng; Wang, Bing; Zhang, Jun; Xie, Chengjun: Insect detection and classification based on an improved convolutional neural network. Sensors, 18(12):4169, 2018.

[YMB19] Yang, Xinyu; Mirmehdi, Majid; Burghardt, Tilo: Great Ape Detection in Challenging Jungle Camera Trap Footage via Attention-Based Spatial and Temporal Feature Blending. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. pp. 0–0, 2019.

[Zh18] Zhong, Yuanhong; Gao, Junyuan; Lei, Qilun; Zhou, Yao: A vision-based counting and recognition system for flying insects in intelligent agriculture. Sensors, 18(5):1489, 2018.

[Zh19a] Zhang, Jian; Zhang, Runsheng; Huang, Yaping; Zou, Qi: Unsupervised Part Mining for Fine-grained Image Classification. arXiv preprint arXiv:1902.09941, 2019.

[Zh19b] Zhang, Lianbo; Huang, Shaoli; Liu, Wei; Tao, Dacheng: Learning a Mixture of Granularity-Specific Experts for Fine-Grained Categorization. In: Proceedings of ICCV. pp. 8331–8340, 2019.