Improving image recognition for species identification by modeling ecological context

A Master Thesis by

Wouter Ubbink
(email woutertwo@gmail.com) ,
with supervision by Dr. Ronald Poppe[a],
daily supervision by Dr. Laurens Hogeweg[b], and
second assessment by Prof. Dr. Albert Salah[a]

[a]Utrecht University, Utrecht, The Netherlands
[b]Naturalis Biodiversity Center, Leiden, The Netherlands

May 14, 2024

Master's programme: Artificial Intelligence
Utrecht University
Student number: 5812453

Abstract

Fine-grained image recognition can be used to identify species from images on the (sub)species level. One of the key challenges for improving the accuracy of species identification models are geographical bias and class imbalance: some species and some areas are overrepresented in the training data. Providing a model with contextual information such as location coordinates, date, environmental variables and neighboring species may help to overcome these problems by creating context-aware predictions.

We combined 22 million images of 31 thousand species with information on location and date of observation, habitat variables and neighboring observations to train a new context-aware model. We employed a transformer architecture that enriches the image representation created by a convolutional neural network, using information from 800 nearby species. Transforming image representations using neighbouring observations is a novel approach to modeling ecological context. This model was compared with a baseline image-only model and ablation models, using existing and new metrics that measure how well the model is able to deal with data biases.

The new context-aware model showed a significant performance improvement on all metrics. The overall accuracy improvement was 1.5 percent point, reducing the error rate by 9.5 percent. Enriching the image representations using a transformer architecture improved the model for most taxonomic groups. Species with few observation records profited more strongly from including contextual ecological information than species with many observations. Rare species that are only present locally could be correctly identified because the model had access to contextual information about the local ecology. Areas with few data points profited more from the new model than areas with a lot of data. The local accuracy in different areas became more equally distributed.

In summary, the new model was better able to deal with geographical bias and class imbalance in the data. Image recognition for species identification thus profits from including contextual ecological information in the model, either as direct input or as a means to transform image representations.

Contents

1 Introduction

In this chapter, we provide an introduction to our proposal to research modeling ecological context in fine-grained image recognition models using a transformer architecture. We start by providing thematic context and setting out the scope of themes and methods we cover in this work. Then, we discuss a problem in current models with the purpose of species identification, after which we propose a solution to the formulated problem. We also motivate our choice to use transformers, although the technical details of the transformer architecture are only described in chapters **2** and **3**. We finish by formulating research questions and summarizing the contributions of this work.

1.1 The context: using citizen science data for geo-aware, fine-grained image recognition

1.1.1 Fine-grained image recognition

Since the advance of high-performing image recognition models, one application of this technology has been to identify the species of plant or animal present in an image. Typically, models with this purpose employ a convolutional neural network and solve a multiclass classification task, where the different classes represent different species to be identified. First, the image is distilled to a one-dimensional image representation vector by a convolutional neural network. Then, the representation is classified by a multiclass neural network. The models are trained to make the right classifications in a supervised machine learning approach, using large numbers of images with labeled species as data. Image recognition models that aim to classify species are a type of fine-grained image recognition, because the goal is to classify particular species and not just broader categories like 'bird' (Yao et al., 2011). Due to the high diversity of wild species, these fine-grained image recognition models typically have thousands of different label classes. Since the species that need to be distinguished are often visually similar, classification based purely on subtle or absent visual differences can be hard, and in some cases impossible.

1.1.2 Citizen science data

In the last few years, a number of publicly available tools that can identify species from images have become publicly available in the form of mobile apps or in-browser applications. Among the most frequently downloaded apps are PlantNet for identifying plants (PlantNet, 2024), Merlin for identifying birds (Cornell Lab of Ornithology, 2023), ObsIdentify for identifying any species within Europe (Observation International, 2024) and iNaturalist for identifying any species worldwide (iNaturalist, 2024). These are used primarily by citizens out of curiosity for what they observed in the wild or to keep a record of their observations, but can also be used for educational or nature conservation purposes. These applications have become increasingly popular, leading to more people uploading their image data, which can be used in training even better image recognition models. When images contain information about the location and date of the observation, the data can also be used for large-scale citizen science projects. This is done in modeling species distributions geographically and researching species abundance trends and spatial distribution trends over time (Feldman et al., 2021; Johnston et al., 2020; Matutini et al., 2021).

1.1.3 Looking beyond visual features

Identifying species that are visually hard to distinguish is a problem for both humans and image recognition models, because of visual similarity between classes and visual diversity within classes. As an example, the problem of visually distinguishing snakes from the genus *Natrix* is shown in **Figure 1**. Sometimes, identification of visually similar species in the wild can be quite easy for experts. This is the case when experts can use other sources of information, like the location, date, or local habitat of the observation for identification (Terry et al., 2020). Indeed, some species occur only in very particular locations or are only observed in certain months. Snakes from the genus *Natrix* have almost completely separated occurrence ranges, shown in **Figure 2**. Identifying them based on location is a lot easier than on visual clues. So, a way to improve accuracy without relying on small visual differences is by using additional, non-visual information in the recognition pipeline to infer which species are more likely to be in the photograph. For species recognition specifically, using location information may be advantageous as the distribution of a single species is generally more concentrated than that of coarse-grained image objects like 'bird' (Chu et al., 2019). Until some years ago, the integration of multimodal information in fine-grained image recognition (instead of using only images) has been underused and understudied (Aodha et al., 2019; Chu et al., 2019). Using spatial and temporal information for enhancing visual categorization of wild species was first done in 2014 (Berg et al., 2014). Later, photographer ID (Aodha et al., 2019) and temperature and habitat variables (Terry et al., 2020) were used as input to image recognition models as well. Location being the most important source of information beside the images themselves, these

developments led to so-called 'geo-aware' image recognition, which is image recognition where the location of the observation is used for improved species identification.

(a) *Natrix helvetica*

(b) *Natrix natrix*

(c) *Natrix tessellata*

(d) *Natrix tessellata*

Figure 1: Snakes from species within the genus *Natrix*. Some snakes look similar (e.g. **(a)** and **(b)**), but belong to different species. Some snakes look different, but belong to the same species (e.g. **(c)** and **(d)**). This makes distinction on visual differences difficult. Pictures by sylvainm_53 (a), lugachev_vitaly (b), andrea_chemello (c) and urusovaalina (d) on iNaturalist.

1.2 The scope of this work

1.2.1 Our work in a nutshell

In this work, we focus on increasing the quality of the image recognition models. The overall goal is to increase the performance of image recognition models by making them better suited to deal with biased data. Existing biases in available data include overrepresentation of some classes and of data from some areas. Currently, many image recognition models only make use of image features. This limits the amount of information the model can harness to make a prediction. Data biases may result in performance biases, like overrepresentation in the predictions of some classes. If image models would have 'access' to more context, the performance of classification may increase because the prediction is tailored to a specific context. This context can be spatial (location), temporal (date and/or time), abiotic (environmental variables) or ecological (nearby species) in nature. Some current species identification models already use location, date and habitat variables to make more accurate predictions. The observation records from citizen science projects often facilitate this, because they usually contain information on the location and date an observation was made. Our work aims to further improve on these models by modeling the ecological context of a query observation using a transformer architecture. Transformer models have been shown to be able to adjust object representations based on context around the object (Azad et al., 2022; Ferrando et al., 2022; Li et al., 2020). We will use the local species distribution around an observation to provide our transformer model with the contextual information it needs to create better representations of the query image. With this architecture, we expect that the model performance increases because the model is more flexible to perform according to the local ecological context. In particular, we expect the performance to increase on data from areas that are underrepresented in the data, and on data containing rare species with a limited distribution range.

1.2.2 Integration of non-visual information about species distribution

We will consider different strategies to integrate non-visual information within image-based classification models and distinguish different types of information fusion. We will limit the types of non-visual information to

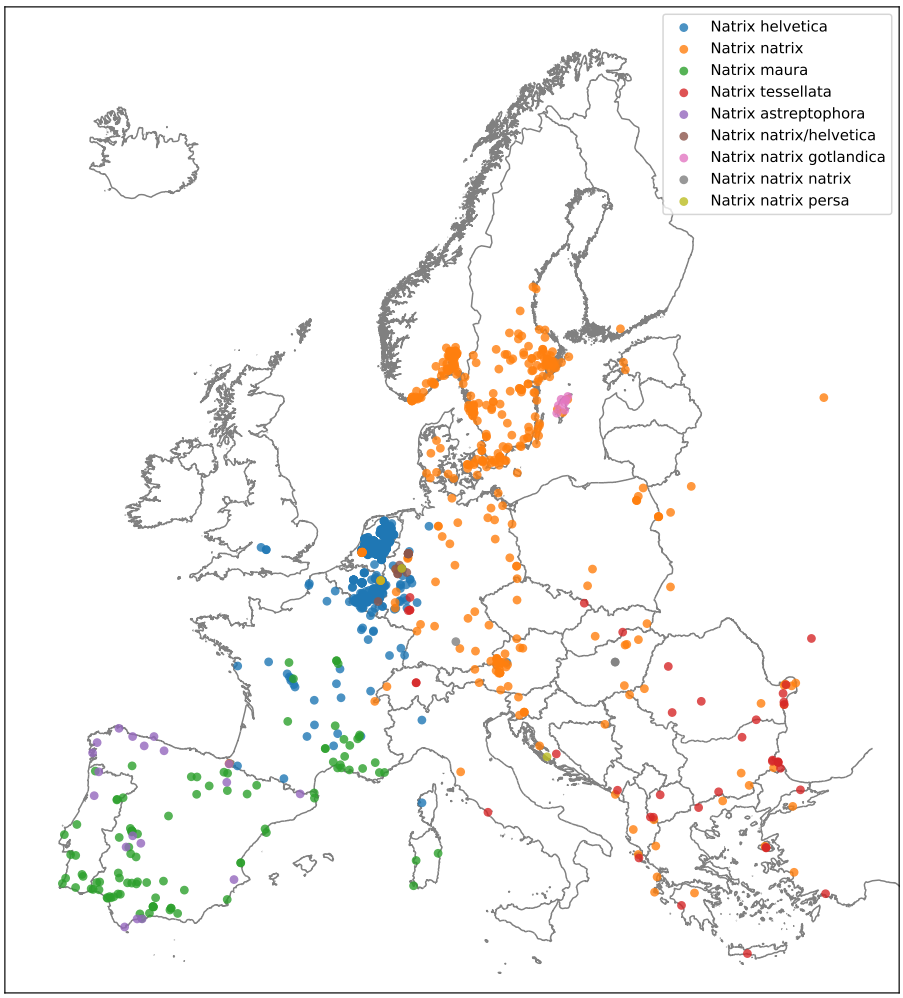


Figure 2: Map of snake observations in the genus *Natrix* (data from the test set). Observations of different (sub)species are indicated by dots of different colors.

information regarding occurrence records, the location and date of the observation and habitat variables. Some models use additional sources of data on species distribution such as those from existing species distribution models (Berg et al., 2014; Wittich et al., 2018). However, recently, citizen science-based species occurrence datasets have become one of the most important sources of raw data able to be used for species distribution modeling (Feldman et al., 2021; Matutini et al., 2021). These are the same datasets that are used for the development of fine-grained image recognition models to begin with, since citizen science-based occurrence records often contain one or more images as proof of observation. This motivates us to use the occurrence records we already have to model the ecological context of the query species, rather than modeling species distribution using external occurrence data. Considering all different methods to model species distributions and their implementation details is outside the scope of this paper. We will discuss some different methods of representing features containing information about location, date and habitat in such a way that they can be used in machine learning models.

1.2.3 Biased performance

As we aim to use contextual information to decrease the impact of imbalanced data, other methods to deal with imbalanced data will not be considered here. However, some of these methods, like undersampling, are applied in training the convolutional part of our model. We will address some different types of bias in performance and consider some existing as well as propose some novel metrics to measure them.

1.2.4 Transformer architecture

We propose a transformer architecture to model the ecological context of the query image. The application of a transformer to image representations from geographically nearby observations is, to our best knowledge, a novel one. We will discuss relevant transformer models used for similar purposes.

1.2.5 Convolutional neural networks

Our proposed method to model ecological context consists of a transformer component that is applied to an image representation. This is done after the representation has been computed by a convolutional neural network. So, the calculations made to project the image to an image representation are not changed in our approach. Most other works that integrate contextual information into an image recognition model for species identification use the same approach. There are exceptions, that either adjust the image input (Xu et al., 2023) or the feature vectors during convolution (Chu et al., 2019). However, most other methods do not 'touch' the pixels of the image before the image representations are computed by the convolutional neural network (Chu et al., 2019; Terry et al., 2020; Yang et al., 2022). In other words, no early fusion of image features and other information features is performed and the convolutional part of the model is not replaced. We choose to follow this approach, in order to retain the power and efficiency of convolutional neural networks. Furthermore, this has the added benefit that our new transformer component can be added to the model without training the entire model again. This is advantageous, as training the convolutional part of the model can take a long time when a lot of data is available. Since we do not make changes in the architecture of the convolutional part of the model, different techniques for processing images to compute image representations are also outside of the scope of this work.

1.3 The problem: biased performance

A challenge for fine-grained image recognition models with the purpose of identifying wild species within a photograph is that they are often biased. We consider two types of biases. Firstly, *geographical bias*: lower performance on data from geographical areas where little data is available. Secondly, *abundance bias*: lower recall on classes with little data available. Classes with few occurrence records usually represent rare and/or locally distributed species, while classes with many occurrence records usually represent abundant and/or widespread species. For simplicity, we will refer to 'rare' or 'common' species when we consider classes with few or many observation respectively. However, it should be noted that rarity and having few observation records are not strictly the same things. The underrepresentation of areas with little data and species with few occurrences leads to imbalanced performance. In previous methods, this issue was addressed by using other information as additional inputs to an existing model, like location coordinates, date and local habitat variables. These variables can contain important information on which species are present where and in what time of the year. These methods often resulted in an improvement of model performance (Aodha et al., 2019; Chu et al., 2019; Xu et al., 2023). However, reduction of the mentioned biases in performance was rarely reported. A possible reason for this is that in many models, the biases in available data across geographical areas (such as differences in data density) are not taken into account (Aodha et al., 2019). Another reason is that current image recognition models are not able to capture correlations between the distribution ranges of different species (co-occurrence patterns). Information on the occurrences of nearby species and local environmental variables are not taken into account, although they may provide important contextual information regarding the location and habitat of the query observation (Aodha et al., 2019; Pollock et al., 2014; Wittich et al., 2018).

1.3.1 Data imbalance

The origin of a large part of the bias in performance has to do with imbalance in data. Imbalance in species observation datasets can be divided into different types based on the nature of the imbalance. First of all, some areas will be more dense in regards to the number of observations. This may be caused by the geographical distribution of participants in the citizen science project that created the occurrence records dataset, that is biased towards well-financed and industrialized countries (Feldman et al., 2021). Another reason for this type of imbalance is that some areas are visited more often by the people that made the observations (Feldman et al., 2021). This can be because some areas are more popular or more accessible to the general public than other areas. Furthermore, birdwatchers travel further to places were rare species can be observed, creating both a spatial and a class-imbalance related bias (Johnston et al., 2020). This first source of imbalance could be solved by collecting data that 'fill the gaps'. The second source of imbalance has to do with the distribution of species themselves. Species can be present in high numbers in one area, but in low numbers in other areas. The third source of imbalance is the overall biodiversity and abundance in different areas. In Southern Europe, for example, there are hundreds of species of grasshoppers and crickets, while in Northern Europe only a few tens of species are found. Species richness varies per biome, generally increasing towards the equator (Fischer, 1960). These two sources of data imbalance cannot simply be reduced by collecting more data, as the imbalances are a naturally occurring phenomenon. If these imbalances would be neutralized, the data would not represent the natural distribution of species well.

1.3.2 Performance across areas

Usually, model performance is higher when new data is provided from areas where training data is available in larger amounts. For example, if species A is common in the area that is overrepresented in the data, species A may also be more likely to be predicted in other areas, even though species B may be more common in those areas. This effect has two main reasons. Firstly, larger amounts of training data results in higher performance on the local species distribution (community), because the model is familiar with it. Secondly, over-classification of the majority group can occur (Krawczyk, 2016). This imbalance lowers the performance of the model on the subdomain of data from underrepresented areas. When image recognition models are developed, they may be regionally implemented in different areas. Decreased performance on data from geographical subdomains can be problematic if one regional implementation of a global model is expected to achieve similar results on regional data as the implementation in other regions.

1.3.3 Performance on different species

Species recall (the rate of correct classifications out of all data containing a particular species) is often lower for classes with few occurrences, that represent rare species. Common species (with many observations) usually have a higher recall. One reason that a species may be overrepresented is because it is more widespread than others, even if it is as common as a set of local species in each location. For example, say species A occurs in the whole of Europe, whereas species B to Z only occur in small, separate areas within Europe, where they are as common as species A. Species A may be overclassified because it is globally more present in the data, resulting in higher recall for species A than for other species. Note that even though the underlying source of data imbalance is natural, it can still cause bias in performance.

Although the two issues mentioned above are slightly different, they occur together in practice. The example in **1.3.2** shows how bias in the geographical distribution of *data* affects performance across *areas*. The example in **1.3.3** shows how bias in geographical distribution of *species* affects performance across *classes*. However, in reality both types of bias in data can cause both types of bias in performance.

1.4 Proposed solution: modeling ecological context using a transformer

To decrease biased performance of fine-grained image recognition models, we propose to implicitly model the local ecological context of an observation by considering species observed nearby. The composition of species in a small area around the query observation location contains information that is relevant for identification of the species in the observation, for two reasons. Firstly, co-occurrence patterns (patterns of two species occurring in similar locations and/or at similar times) can help us make better occurrence predictions (Rodríguez de Rivera et al., 2018; Thompson et al., 2020). For example, the occurrence of a species of butterfly may be much more likely in areas where its host plant is also present. Secondly, the local composition of species may be a unique proxy for a certain area and/or habitat (Auster et al., 2001). Because there are so many different species, all with different distribution ranges, the composition of a substantial number of present species may very well be unique for a certain area, a certain habitat, or both, such as 'heathers in Scandinavia'. In other words, the model may be able to infer the location and/or ecology of the query species by using the composition of species present in the neighborhood, which can then be used to make a geo-aware species prediction.

By defining the size of the neighborhood of a query observation using a fixed number of neighbors, the neighborhood size will be adjusted to the amount of data available in an area. The advantage of this is that we avoid defining neighborhoods that are too small (i.e., areas with too sparse data to accurately model ecological context). Furthermore, we avoid defining areas that are too large (i.e., areas that have so much data that ecological context could easily have been modeled on a smaller, more precise scale).

1.4.1 Motivation for using a Transformer architecture

To model ecological context around a query observation, we propose to add a transformer component to an image recognition model. Our aim is to increase both the general performance and the balance in performance across classes and areas in our species recognition model. In the last years, transformer models have shown excellent performance on different types of tasks (Dosovitskiy et al., 2020; Vaswani et al., 2017; Zhang et al., 2023). The use of multi-head attention modules within the transformer enables it to enhance an input instance in such a way that relevant context from other input instances is taken into account (Cordonnier et al., 2020). For example, a word embedding can be enhanced based on other words in the sentence or an image patch embedding can be enhanced based on other patches in the image (Dosovitskiy et al., 2020; Vaswani et al., 2017). This gives rise to the idea that a transformer architecture may be able to enhance a query input vector that represents an image from an observation in such a way that contextual information from nearby observations is taken into account. In the context of geo-aware image recognition, we can use the geographically nearby occurrence records

to obtain these nearby observations. Thus, we model the ecological context of a query species by using the presence of neighboring species in order to increase performance in species classification. Particularly, we expect the model to use the presence of one species in the neighborhood to increase its prediction likelihood for another species that often co-occurs with the first species. We also expect species classification to work especially well in areas with a unique ecological context (i.e., a unique composition of local species), because the model is flexible enough to adjust its predictions to this context. Another advantage of the transformer architecture is that each element of the context can be weighted differently by the multi-head attention mechanisms. In our context, this means the different amounts of contextual information that different (groups of) neighbors provide can be taken into account. This is advantageous because in reality, not all neighbors are equally important for modeling the ecological context. For example, the presence of a local rare species may provide more contextual information than the presence of a common widespread species. Furthermore, a transformer architecture is able to capture information in the relationship between input elements. This means ecological interactions between neighboring species may be extracted to provide contextual information. For these reasons, we expect our model to have a better and less biased performance than models that do not take ecological context into account.

1.4.2 Interaction between species composition and location

In the new model, we will use direct metadata information from the query image (location, date), derived data (local habitat variables) as well as the information from neighboring species to model ecological context. However, using direct metadata information and derived information such as location, date and habitat variables may prove to be unnecessary. The composition of species in the neighborhood is likely to be unique in every area, especially when enough neighbors are taken into account. This means that the composition of neighboring species can serve as a proxy for information about location and habitat. If this is the case, using information about location and habitat as input to the model may be redundant. To test this, we perform an ablation where only the query image representation and information from neighboring observations are used as input to the model. Nevertheless, we expect using location, habitat and date variables as explicit input to the model may still provide some added value to the model, because they possibly contains different information than the representations of nearby neighbors. That is why these variables are used as additional inputs in our full model. The ablation study will evaluate if modeling ecological context using neighboring occurrences is complementary to explicitly taking location, date and habitat information from the query image into account.

1.5 Research questions: can bias be reduced by using ecological context?

Fine-grained image recognition models for species identification are biased towards geographical areas with more data and towards common species. If models were able to take into account the local and ecological context around a query species, their bias may be reduced. This raises our main research question: To which degree can we reduce geographical and abundance-based biased performance by modeling the local ecological context? From this question, the following lower-level research questions emerge:

1. To what degree does modeling ecological context increase overall model accuracy?
2. To what degree does modeling ecological context reduce abundance bias?
3. To what degree does modeling ecological context reduce geographical bias?
4. To what degree does modeling ecological context reduce abundance bias across different areas equally?
5. Can modeling ecological context and directly integrating non-visual information complement each other in regards to reducing biased performance?
6. What are the consequences of modeling ecological context in regards to computational requirements?

So far, most research into integrating non-visual information in fine-grained image recognition models has focused on increasing the accuracy of the models. In contrast, the focus of this work is on reducing bias in performance. We propose a method to model ecological context that we expect to, in addition to improving overall accuracy, help decrease biased performance over geographical subdomains of the data and increase recall over classes representing rare species. In Chapter **3**, we will cover our proposed method and relevant evaluation metrics in more detail.

1.5.1 Contributions of this work

The contributions of this work can be summarized as follows:

- We propose a transformer architecture to model local ecological context, using nearby observations and their relations to adjust image representations. The transformer architecture improves model performance.

- We evaluate the potential to reduce geographical and taxonomy-based bias using a number of different evaluation metrics, aimed at answering our research questions. Our model managed to reduce taxonomy-based bias and substantially reduce geographical bias.
- We compare our proposed ecology modeling with strategies for integrating location, habitat and date information by performing a number of ablations. We also evaluate the complementariness of our approach and traditional approaches. We find that modeling ecology is partly complementary to integrating location, habitat and date information in a model.

2 Related work

In this chapter, we discuss previous work on integrating non-visual information into image recognition models. We limit the information sources we consider to location, date, habitat and species distributions. We first consider how spatial and temporal information can be represented so it can be used by a model. Then, we consider strategies to integrate non-visual information in existing models. We focus on the use of location information, as using this information has led to the biggest performance gains in previous work. Finally, we consider some previous work on transformers within image recognition models.

We describe methods for integration of non-visual information in the context of image recognition models. That means we assume the new information has to be integrated into an existing model that processes image data into representations that can then be used to predict the image class. Most modern approaches for image processing apply machine learning, in particular convolutional neural networks. Most works considered in this chapter use this approach and they differ in how the non-visual information is integrated into their respective classification models.

2.1 Representing geographical information

Non-visual geographical information like latitude, longitude and date can be used as input to a neural network but have to be processed. This processing usually involves scaling the coordinate variables so they are easier to process for machine learning models. For example, Chu et al. (2019) normalized latitude and longitude values to fall within the range $[-1, 1)$, after which the location data is used directly as input to their network. Date is usually processed to a day-of-year variable, which can then also be normalized to fall in the range $[-1, 1]$. Aodha et al. (2019) and Yang et al. (2022) wrap their normalized coordinate and date values by performing the mapping from x to $[\sin(\pi \cdot x), \cos(\pi \cdot x)]$, resulting in a total of six values; two to represent each of the three variables (latitude, longitude, and day-of-year). In other words, the variables are mapped to a circular representation. Note that this mapping makes intuitive sense for at least the longitude and date variables, as it forces the representation of data from December 31st to be similar to that from January 1st and data with large longitude values (e.g. the Russian Far East) to be similar to data with small longitude values (e.g. from Alaska). Xu et al. (2023) use an approach that is slightly more complex, since they apply a fully connected neural layer to the normalized and wrapped location and date variables separately before concatenating them for further use. The advantage of this approach is that the fully connected layer may capture some spatial or temporal patterns that are present solely in the location variables or solely in the date variable. For a more comprehensive review of different methods to process or encode location information, see Mai et al. (2022).

2.2 Strategies for integrating location information

There are different methods to use spatial and temporal information in an image recognition model. They mainly differ in the part of the model in which the information is integrated and in the type of information that is used in the model. We distinguish four types of feature fusion to integrate information: very early fusion (before convolution), early fusion (during convolution), middle fusion (after convolution but before classification), and late fusion (after class scores have been calculated for the image features). For a review of different strategies of integration see Chu et al. (2019).

2.2.1 Geographical features as input

The first method to use spatial and temporal information in an image recognition model is feeding this information from the query observation into a model directly, typically by using middle fusion. This information is usually first processed in the way described in **2.1**. The advantage of using geographical features as input to the network is that you can compute vectors that jointly encode image features and geographical features. In this way, possible interactions between image features and geographical features can be captured and exploited for better performance.

Proposed by Chu et al. (2019), one strategy for integrating the location information is to do it in multiple points in the model. First, they apply a number of fully connected neural network layers to the processed location information. Then, they modulate the image features within the convolutional neural network by adding the location features to the image feature at multiple points within the network. In this way, they were able to capture possible deeper correlations between location and mid-level concepts extracted from the image early in the network. In contrast to other methods, they perform a form of early fusion, although they do not touch the lowest level image features, as these represent low-level concepts that do not yet benefit from modulation by location information.

Yang et al. (2022) expand on this idea using a dynamic feature fusion method instead of addition. In their model, the processed location and date information is transformed from a one-dimensional to a two-dimensional

vector. The new shape is then used as weights to project the current image feature vector to a new vector. This process is repeated iteratively. The underlying idea of the dynamic feature fusion is to separate the different classes in representational space in regards to species distribution, making them easier to classify.

Xu et al. (2023) use a different approach. They also modulate the image features later in the image recognition pipeline using the dynamic feature fusion method, but only after convolution (middle and late fusion). Additionally, they modify the input image directly (very early fusion) by multiplying processed geographical information that is projected to two dimensions to the pixel values in the corner of the input image. By using the corner, they hope to turn non-informative noise in the image into information that can be used for classification, without touching pixels that contain relevant information, which are expected in the center of the image.

2.2.2 External features as input

A second method to use spatial and temporal information in an image recognition model is using external data to map a location to a set of features which are then used as input to the model. The advantage of this approach is that the model does not need to encode geographical features in such a way that more complex information, like habitat variables, are represented in the encodings. Rather, more complex information is used directly by exploiting existing knowledge resources.

Tang et al. (2015) use this method to extract features from maps using the GPS coordinates linked to an image, which are then used for context-aware image recognition. Their method was not aimed at recognizing species, but more general classes of images. Because they extract features from a large number of sources, they reduce the dimensionality of their final feature vector and concatenate it with the visual features for further processing. In this way, they could take data from a great variety of sources into account.

In the context of species identification, maps and tables are available with habitat information for each location and/or date. The habitat information from the location of the query observation can be used as a feature in a neural network. Terry et al. (2020) used this method to extract temperature and habitat information using the location and date variables linked to an image. They combined this information with the processed location and date variables to create a joint metadata vector, which they used as input to several different image recognition models. Using both location and date variables as well as extracted habitat and temperature variables, they hoped to capture as much geographically valuable information in the model as possible.

2.2.3 Label whitelisting

A third method to use spatial and temporal information in an image recognition model is to filter out species that are not expected in a certain area by making a 'whitelist' of species known to be present for each area. No feature fusion is performed. The advantages of this approach is that it is relatively easy to implement and it can be added as a post-processing step to any type of model.

This approach was used by Chu et al. (2019) as one of their strategies. They create a circular area with a radius of 100 miles (roughly 161 kilometers) around the query location and use the occurrences of species within that radius to create a whitelist of species that could occur at the query location. Possible output labels were limited to the species on this whitelist. Radii of 50, 500 and 5000 miles were also tried but delivered worse results. Although simple in implementation, this method fails when a species is observed outside its regular range. Furthermore, the presence of widespread but rare species can not easily be established for each area if limited data is available or if the considered neighborhood is too small to have accurate data for all species.

2.2.4 Modeling species distributions

The fourth method to use spatial and temporal information in an image recognition model is to first model species distributions to calculate the likeliness of occurrence of each species in each area. Typically, these distribution models are created by using known species occurrences in the neighborhood of the query observation. Then, the species distribution model is used to calculate the prior likelihood distribution of a species occurring in the location of the observation. These prior likelihoods can be multiplied with the probability score outputs of the processed image features to obtain final (posterior) prediction probability scores. In other words, late fusion is performed. This strategy follows Bayes' theorem and uses the assumption that image features are conditionally independent of other information features given the class (species) (Berg et al., 2014). The advantage of this approach is that modeling a prior distribution and an image feature distribution separately makes intuitive sense and is explainable. Using only location information, one can already visualize where certain species are expected. An important drawback is that the assumption of independence between image features and other features may not hold in reality. For example, if image features contain some information about a certain location, like snow in the background, the features representing image and location are not conditionally independent of each other given a class. However, the presence of interactions between images and locations that would be relevant for model performance may be rare for images of plants and animals (Chu et al., 2019).

Wittich et al. (2018) used two sources of species distribution data to create a ranked list of probable species to occur in a certain location. The first species distribution data source was presence-absence data, containing a zero for each species that does not occur in a certain area and a one for each species that does. They used the counts of this binary presence-absence data of different species in a number of tiles around the query location. Using these counts and the distances of each tile center to the query location, they calculated the ranked list of probable species using four different formulas. The second species distribution data source was occurrence records from the Global Biodiversity Information Facility (GBIF). They used this data to create ranked lists of probable species using the same methods as for the first source, but added one formula. In this last formula, they included a temporal distance measure by weighting monthly species occurrence frequencies in a Gaussian manner around the month in which the query species was observed. The use of multiple sources of information on species occurrence should be able to generate better prior probabilities of species occurrences.

Berg et al. (2014) used a similar approach. They use observation occurrence records to construct a spatio-temporal prior for species occurrence, by dividing the estimated density of the occurrence of the query species by the estimated density of the occurrence of any species. The densities were calculated by taking the weighted counts of the nearest 500 neighboring occurrences, with weights dependent on distance to the query location. To decrease computational requirements, they discretized observations into spatiotemporal cubes with dimensions of a quarter of a degree latitude/longitude and six days. In this way, they could more easily calculate the spatio-temporal distance between the query and neighboring data points. Also, this method gave them the possibility to define how spatial distance relates to temporal distance. Once the spatio-temporal priors were calculated, they used them to reweigh the output probabilities of their support vector machine (SVM) image classification model. The advantage of their approach is that they can deal with differences in data densities across areas. They calculate densities based on an adaptive neighborhood size, because they define the neighborhood size using the distance to the 500th nearest neighbor to the observation location. In areas with sparse data, this neighborhood will be bigger and thus data that originates further away from the query location is taken into account.

Chu et al. (2019) used an approach similar to their whitelisting method to compute location-based Bayesian priors. They created a histogram (data vector) of species occurrences within a radius of 100 miles around the query location and used the histogram values directly as a prior class label probability. Again, different radii were tried but a radius of 100 miles gave the best results. Although more complicated, this strategy performed worse than their whitelisting approach. An advantage of this approach is that local species abundances are taken into account. However, this may also be a disadvantage as it may cause overclassification of common species.

Aodha et al. (2019) used a special method to model species distributions. Their model learns both a species embedding and a geography embedding. Then, by multiplying the geography embedding with all species embeddings, they get a vector of species occurrence likelihoods for that geography. Geography is encoded using multiple neural network layers applied to the processed location and time variables. An advantage of their approach is that geographical information does not need to be discretized, since their model can use any location coordinates and any date as input. Another advantage is that they can confirm that species with similar embeddings tend to occur in similar areas, because their species embeddings and geography embeddings share the same representational space. A disadvantage is that the embeddings will be of different quality in areas with different amounts of available data, as all geography embeddings share one representational space.

2.3 Transformers

While transformer networks using attention modules were originally used for language models, they are now popular in many other AI applications, including computer vision. The first transformer, proposed by Vaswani et al. (2017), used the sequence of words in a sentence to adjust the word embedding of each word in order to take into account the context of each word provided by other words in the sentence.

Dosovitskiy et al. (2020) showed transformers can be used in image recognition to replace CNN architectures. In their work, they split an input image into patches and feed the sequence of linear representations of these patches as input to the transformer, instead of a sequence of words in a sentence. Before the patches can be used as input, they are flattened and are mapped to a vector using a trainable linear projection. The result of this projection are patch embeddings. Besides these vectors the authors also prepend the vector sequence with one extra learnable embedding. The state of this so-called classification input embedding after processing by the transformer serves as the image representation that is used for classification. The authors add a learnable position embedding to each of the patch embeddings and to the classification input embedding. After this addition, the sequence of vector embeddings are ready to be used as input to the encoder. The structure of the transformer encoder as applied by Dosovitskiy et al. (2020) will be considered in more detail in the next paragraphs. After the encoder module, the processed classification input embedding now constitutes the image representation. It is put through a classification head, which is composed of a single learnable layer, to calculate the class prediction scores.

The transformer encoder consists of alternating multi-head-attention blocks and multilayer perceptron (MLP) blocks. Each block consists of three parts: 1. layer normalization, 2. the main element (attention or MLP), and 3. a residual connection connecting the input of the block with the output of the main element in the block. The MLP main elements consist of two learnable layers, each with a Gaussian Error Linear Unit (GELU) non-linear activation.

The multi-head attention main elements are standard multi-head attention mechanisms as described by Vaswani et al. (2017). In a multi-head attention mechanism, multiple single self-attention mechanisms ('heads') are applied to a sequence of vectors. Each single self-attention mechanism first calculates three linear projections (called key, query and value) of the input sequence vectors. The weights of these linear projections are learned separately for each attention head. The dot product of one key and another query projection is calculated and then scaled by dividing it by the square root of the input vector dimension. The outputs of this pairwise similarity operation between key and query are attention scores, one for each combination of two vector input elements. These scores are then normalized to obtain attention weights. Using matrix multiplication, these weights are combined with the value projections to produce the context-aware output vectors. This is done for all single self-attention mechanisms, after which their outputs are concatenated and projected to vectors of the same lengths as the input vectors.

The advantage of attention modules is that the input embeddings are adjusted in such a way that relevant information from other input embeddings can be taken into account. In this way, the 'context' of the query input, based on other inputs, is used to enhance the representation of the query input. For language transformers, this means the context of a sentence is taken into account in the representation of each word. In vision transformers, the context of different parts of the image is taken into account in the representation of each image patch. The advantage of multi-head attention modules in particular is that attention can be focused on multiple distinctive aspects of the input sequence, which can improve performance.

Xu et al. (2023) used a single self-attention mechanism to process their location and date information and exploit the correlation between these variables. In their method, the location and date information, already processed as described in **2.1**, are used as inputs to the self-attention mechanism. The output of the attention mechanism is processed further by applying two trainable linear layers. The output of this operation is then used to modify the image features, as described in **2.2.1**. The advantage of this approach is that they can capture patterns between location and date variables to enhance the encoding of these variables. However, their attention module is not applied to any image representation, so their approach is limited to capturing relations between location and date variables.

Mai et al. (2020) used two sequential layers of multi-head attention mechanisms for the purpose of spatial context decoding. The problem they wanted to tackle was to predict attributes of a point of interest (e.g. the capacity of a building in a certain location) using the location of the query point of interest, and the location and attributes of neighboring points of interest. Using location embeddings and feature embeddings of neighbors as input, they applied one multi-head attention mechanism to calculate initial predictions of the query attributes. Then, they used the same input plus the initial predictions and applied a second multi-head attention mechanism to calculate the final predictions of the query attributes. The purpose of this approach was to model spatial context of a data instance in a certain location so that features of that data instance could be predicted.

3 Method

In this chapter, we explain the details of our proposed method. We start by explaining and motivating the use of the transformer architecture and the changes we make to it, after which we cover the data we use. We proceed with covering the design of our current model and our additional transformer component, including technical details. Then, the implementation details during training and inference are precisely explained. Finally, we discuss our expectations regarding performance, the models we compare our new model with and the evaluation metrics that we use. Code for all described models will become available at https://gitlab.com/wouter.ubbink/msm-geo.

3.1 A transformer architecture for modeling ecological context

We propose to use a transformer component to model the ecological context of the query species. In our model, we make use of both neighboring observations and query location, habitat and date variables. The query image representation is combined with 'community representations' made from neighboring species for different taxonomic subgroups. The community representations are meant to embed the species distributions in the local neighborhood. Together, the image representation and the community representations form the input to our transformer component. The transformed output image representation is then fused with the location, habitat and date variables. Then, a final neural network layer is applied to the fused features to calculate the final class scores. Because we introduce new information after convolution but before class scores are calculated, we perform middle fusion. This type of fusion has the advantage that the convolutional part of the model is not changed. This makes development and deployment of the model much easier, as retraining the convolutional part of the model requires a lot of computational resources.

The application of transformers to model ecological context around a query image is, to our best knowledge, a novel one, although others have performed somewhat similar approaches. Xu et al. (2023) used an attention module to improve the representation of location and date information within an image recognition context. However, they used location and date information as input to the transformer module, which we do not. They did not use image representations or neighboring species as input to the attention module, which we do. In our approach, we expect to capture and exploit the relationship between the image and its ecological context, instead of the relationship between location and date variables. Perhaps most similar to our approach, Mai et al. (2020) used two multi-head attention mechanisms to predict attributes of a point of interest at a certain location using the attributes of points of interest in nearby locations. The similarity between their work and ours is that we both encode spatial context using embeddings from geographically nearby data points. Instead of using attributes of points of interest, we use both image representations and neighbor-based community representations as input to our transformer component, as the main goal of our model is image classification. Furthermore, we combine the information of multiple neighboring data points before using them as input to the transformer, so that data from more neighboring observations can be used in the model.

Since the application is novel, our architecture should be as simple as possible. However, the original transformer architecture described by Vaswani et al. (2017) is not exactly what we are looking for, since we do not need a decoder component, nor do we need masked multi-head attention mechanisms. Furthermore, it makes sense to follow up on an existing vision transformer, since we are using processed image features as input to the transformer. For these reasons, we decided to follow the vision transformer (ViT) architecture, described by Dosovitskiy et al. (2020), as closely as possible. The details of their approach are described in **2.3**.

3.2 Conceptual changes to the transformer architecture

Our transformer architecture is an adaptation of the ViT architecture (Dosovitskiy et al., 2020). To capture the relationship between the query species and its ecological context, we provide the model with information from neighboring data points. Although we want to keep the model architecture as simple as possible, we do make some changes in order to fit the model architecture to our needs. Here, we describe the changes on a conceptual level. The technical details of our implementation will be given in **3.5** and **3.6**.

Firstly, we change the input of the transformer encoder. As we want to model the ecological context around a query observation, we will use species information from nearby observations as input to the model. This information is projected to community representations, which have the same shape as our query image representation. In this way, the query image representation and community representation can be used as inputs to the transformer. This means that in our approach, the image is not divided into patches in the way done in the ViT model. Instead, the neighboring community representations take up the place of different image patches. Flattening the images is not strictly needed, as the convolutional part of our model has already projected the input image to a flat vector representation. However, flattening in the ViT model is done using only a dense layer that shares weights across inputs. This means we do not need to take the 'flattening' operation

out. Instead, we use the dense layer as an extra layer that has the potential to homogenize the inputs, since its weights are shared across inputs.

Secondly, we combine the information from multiple neighbors to create input representations. The number of inputs to the transformer encoder is computationally limited, since the pairwise similarity operation in the multi-head attention mechanisms scales exponentially with larger numbers of inputs. Therefore, we cannot simply use one representation for each neighbor as input to the transformer. This is because we want to use information from more than a few neighbors, since chances are that useful context can only be modeled using enough neighboring observations. That is why we choose to combine the species information from many neighbors into a few community representation inputs for the transformer. In this way, more neighboring observations can be used to model the ecological context without increasing the computational requirements of the multi-head attention mechanisms too much. An analogy can be made between combining large numbers of pixels into a few image patches and combining large numbers of neighbors into a few neighbor groups.

Thirdly, neighbors are grouped by taxonomic subgroup instead of position. In the ViT model, pixels are grouped by position under the assumption that neighboring pixels contain related information. In our model, we use neighbors inside a small area around the query location. Although in theory the position in relation to the query observation and in relation to each other could contain some relevant information, we expect this to be little, especially if the neighborhood has a small size. Rather, we use the fact that neighbors of the same taxonomic groups contain related information because of their taxonomic similarities and, consequentially, their similar roles in the ecological community. That is why we choose to group neighbors by taxonomic subgroup. The species information from the neighbors of each taxonomic subgroup is combined to form one community representation that can be used as input for the transformer encoder.

Fourthly, positional embeddings have a different meaning in our model. In the ViT model, positional embeddings are a learnable vector of the same length as the input vector, so that the input vector and the positional embedding can be summed. The embeddings represent the positional information that is specific to each input patch. In our approach, the embeddings represent information specific to each taxonomic group. Different taxonomic groups will provide different types of contextual information. Since the positional embeddings in the ViT are simply a learnable vector, we are able use the exact same architecture and learning approach to incorporate these different types of information. So, our model does not differ in a technical sense, but what the embeddings represent is different between the ViT model and our model. That is why we call the positional embeddings in the ViT model taxonomy embeddings in our model.

3.3 Data

3.3.1 Observation records

Observation records are used from Observation International (The Netherlands), Artsdatabanken (Norway), Artdatabanken (Sweden), United Kingdom Species Inventory (United Kingdom), Arter.dk (Denmark) and FINBIF (Finland). Observations include one to four images of the query species, location coordinates and date. These datasets are not publicly available. In total, the unfiltered data consists of 35 million images of more than 32 thousand different plant, animal and fungal species, with associated location and date information for each image. The images and labels have been uploaded by citizens and the validity of all labels has been validated by human experts. These datasets fit our research purpose as they contain both images and metadata. Large amounts of observations are expected to be needed to train a flexible, complex model like a transformer.

The data we select is limited to observations from locations with coordinates between 33 and 72 degrees latitude and between -25 and 43 degrees longitude, roughly covering Europe. We apply this focus because the data sources we use contain few occurrence records outside of Europe. However, even after this selection the data still contains many more observations in the countries hosting the data portals than in other European countries. Data density is especially high in The Netherlands. Only one image per observation is used, as our model can only process one query image at a time. After these selections are performed, our data consists of more than 22 million observations with one image each, containing over 31 thousand different species. Although 1179 species are filtered out of the dataset, we use the original number of species to determine the number of classes in our models, for technical reasons.

Eight subsets are created, based on the taxonomy of the species in the observations. The taxonomic subgroups are Plantae (plants, including algae and protozoa), Fungi (including lichen), Chordata, Lepidoptera (butterflies and moths), Diptera (flies and mosquitoes), Insecta (other insects), Arthropoda (other arthropods), and Mollusca (molluscs and other animals).

The labels of the observations most often correspond to the species in the observation. A limited part of the labels actually signifies the genus or subspecies (i.e. a taxonomic subcategory ranking below species) rather than the species. This is because for some observations, the subspecies was identifiable to the observer or validator, and for others only the genus could be inferred. For simplicity, we refer to classes as representing species. In **Table 1**, the number of classes and the number of observations are given for each taxonomic subgroup.

We divide the data into a training set containing 80 percent of the data, and a validation and a test set both containing 10 percent of the data.

Subgroup t	**Classes s_t**	**Observations**
Arthropoda	1,256	495,653
Chordata	2,941	4,591,317
Diptera	2,479	909,220
Fungi	4,817	1,784,489
Insecta	6,882	3,687,107
Lepidoptera	3,798	4,621,396
Mollusca	1,041	351,486
Plantae	9,294	5,894,081
Total	**32,508**	**22,334,749**

Table 1: Number of classes and observations for each taxonomic subgroup.

3.3.2 Location and date preprocessing

As a preprocessing step, we convert the date variable to a day-of-year variable, and divide it by the number of days in the year of the date, so that the values are limited to the range $(0, 1]$. Then, we wrap the date variable so that the representation of data from late December is similar to that from early January. Similar to the work of Aodha et al. (2019), we do this using the mapping from x to a circle using the transformation $[\sin(\pi \cdot x), \cos(\pi \cdot x)]$. This results in two values for each data instance. We call the vector containing these two values $x_{date} \in \mathbb{R}^2$.

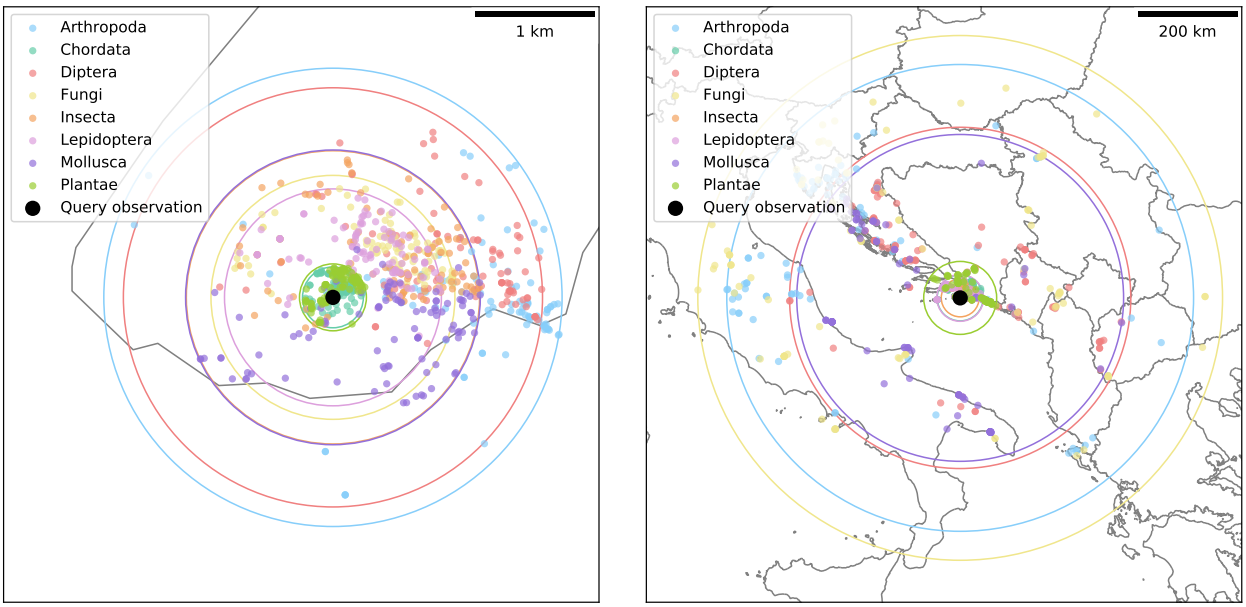
We do not wrap the location information, as the area we cover is not a spherical surface. We do scale the location coordinate values (latitude and longitude) to fit in the range $[0, 1]$, creating $x_{location} \in \mathbb{R}^2$. We also use 64 different variables containing information regarding land cover and climate (e.g. elevation, annual mean temperature, urban areas). This data is discretized to a grid with cell sizes of 10 minutes (one sixth of a degree latitude/longitude). The size of these cells is approximately equal to 19 km in the longitudinal direction and 6 to 15 km in the latitudinal direction, depending on the longitude. We scale each habitat variable to fit in the range $[0, 1]$, creating $x_{habitat} \in \mathbb{R}^{64}$. $x_{habitat}$ is determined by the habitat grid cell in which the location of the observation resides.

3.3.3 Neighbor preprocessing

For each observation, neighbor data is collected from the observation records. From each of $|t|$ taxonomic groups, data from the k geographically nearest neighbors is collected. We use $|t| = 8$ and $k = 100$. For each taxonomic group t, a vector is created with length $s_t + 4$, where s_t is equal to the total number of species within taxonomic group t. The first s_t dimensions of the vector are filled with the frequency values of the species occurrences (abundances) within the neighborhood. The sum of these values is thus equal to k. The other 4 dimensions contain variables relating to the distance from the query location to the neighbors. They include the distance to the closest neighbor, the distance to the furthest neighbor, the average distance from query to all k neighbors and the standard deviation of the distances from query to all k neighbors. These extra variables are included because they contain information about the shape and size of the neighborhood, which may be valuable for classification. For example, in areas with sparse data the values of these distance metrics are different than in areas with a lot of observations. Since we want to build a model that does especially well in areas with sparse data, including these extra variables may enable the model to learn patterns from them that are relevant to predicting the right species. The difference between neighbor information from areas with dense and sparse areas can be seen in **Figure 3**. Neighbor vectors are denoted $x_t \in \mathbb{W}^{s_t} \frown \mathbb{R}^4$, where $\frown$ indicates 'concatenation' and $t \in \{arthropoda, chordata, diptera, fungi, insecta, lepidoptera, mollusca, plantae\}$.

3.4 Current model architecture

The model that is currently deployed by Naturalis Biodiversity Center in Leiden, The Netherlands for use by regional partners consists of all elements in our complete model except information on neighboring species, the transformer component and the addition of location, habitat and date variables. The first part of this current model is the image processing, which encompasses the main convolutional model and t submodels, one for each taxonomic group. Since we have eight taxonomic groups in our data, we use $t = 8$.



(a) Terschelling, The Netherlands

(b) Mljet, Croatia

Figure 3: Maps of neighbor data from different locations. The query observation is marked with a large black dot. Neighboring observations, 100 from each taxonomic subgroup, are marked with dots of different colors for each subgroup. Circles are drawn to mark the outer border of the neighborhood for different taxonomic groups using corresponding colors. In some areas, like in The Netherlands **(a)**, a lot of data is available. This creates neighborhood sizes in the range of a few hundred meters to a few kilometers. All neighbors are drawn from the same small island. In other areas, like in Croatia **(b)**, much less data is available. This creates large neighborhoods in the range of tens to hundreds of kilometers, covering multiple countries. Note that some taxonomic subgroups have smaller neighborhoods than others. This has to do with a bias in the data, but can also be caused by taxonomy-specific geographical effects. For example, Mollusca observations are mostly present in coastal areas.

The 'main' convolutional model, consisting of a deep neural network with EfficientnetV2M architecture (Anavyanto et al., 2023; Tan & Le, 2021), is trained to determine which subgroup t an image belongs to. Once the subgroup of the query image is determined by the main convolutional model, the image is processed by the specific convolutional submodel the image belongs to. These submodels are each separately trained and also employ the EfficientnetV2M architecture. The output of this image processing component consists of an output vector of length 1280. The content of this vector is a representation of the species in the image, in the context of the submodel it is trained on. Since we use the image representations as input to our new models, we call this representation vector $x_{image} \in \mathbb{R}^{1280}$

The final part of the model consists of $|t|$ classification heads. In the current model, only the image representation is used as input to the classification heads. The vector x_{image} is projected to a vector of length s_t using a simple learnable layer that performs a linear projection, with s_t being the number of different classes (species) the taxonomy-specific submodel t is trained on. The output of this projection is activated by a SoftMax layer that produces likelihood scores for each class. So, the current classification heads can be described mathematically as:

$$\hat{y} = softmax(W_{final} x_{image} + b_{final}) \tag{1}$$

where $W_{final} \in \mathbb{R}^{s_t \times 1280}$ represents the weights in the final layer, $b_{final} \in \mathbb{R}^{s_t}$ represents the biases, and $\hat{y}$ indicate the class likelihoods. These output scores represent the likelihood that the query image contains a certain species. The class (species) with the highest likelihood score can be interpreted as the most likely label. Since the submodels are trained separately, the output vectors of different submodels exist in different representational spaces. Furthermore, the length of the likelihood score vector of each head is different, as different taxonomic groups contain different numbers of different species s_t. For these reasons, $|t|$ different classification heads are trained separately. An overview of the complete current model architecture can be found in **Figure 4**.

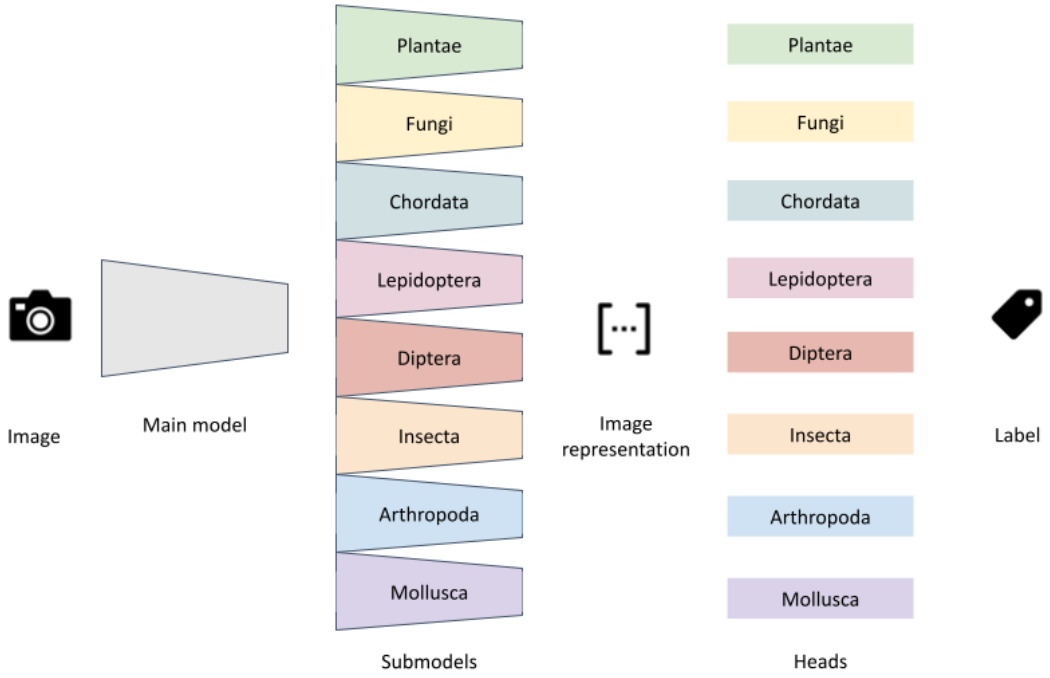


Figure 4: The main components of the current model, from left to right. The approach consists of processing an image input through the main convolutional model, one of eight taxonomy-specific submodels, and one of eight classification heads to compute a class label as output. The main convolutional model and one submodel together make up the initial image processing component. The output of these components are the image representations. From there, one of the classification heads projects the representation to a set of class likelihoods, from which the final label prediction can be determined.

3.5 New model architecture

Our new model architecture replaces the classification heads of the current model architecture. The components that compute an image representation from an image are left unchanged, so they are not considered further.

In our new model, neighboring species data is added as input. The neighbor data vectors x_t are each projected to a 'community representation' $a_t^{[0]}$. This is done using a standard dense (fully connected) neural network layer with a ReLU activation function:

$$\forall t \in Subgroups : a_t^{[0]} = ReLU(W_t x_t + b_t) \tag{2}$$

where $Subgroups = \{arthropoda, diptera, chordata, fungi, insecta, lepidoptera, mollusca, plantae\}$, $ReLU(x) = max(x, 0)$, and $W_t \in \mathbb{R}^{1284 \times (s_t+4)}$ and $b_t \in \mathbb{R}^{1284}$ represent the weights and biases in these layers. By adding data on neighboring species as input to the model, we are implicitly modeling the local species distribution. In this way, we expect the model to make better predictions, as species distributions provide information on label likelihood. By transforming the species frequencies using learnable weights, we avoid the need to model a local species distribution explicitly. Furthermore, by performing middle fusion rather than late fusion, we do not use the assumption that image features are conditionally independent of other information given the class. By combining modeling species distribution and middle fusion, we hope to profit from the advantages of both approaches. Also, by using this linear projection, we avoid the risks of using species abundances directly mentioned in **2.2.4**.

The length of the community representation is matched with the length of the query image representation (i.e. 1280), but has an additional 4 dimensions for the distance metrics from each taxonomic group. However, to ensure uniform elements as input to the transformer the image representation vector now also needs to be extended with 4 dimensions. The values of these 4 dimension are put to zero.:

$$a_{image}^{[0]} = x_{image} \frown [0, 0, 0, 0] \tag{3}$$

So, the final length of both the query image representation $a_{image}^{[0]}$ and the community representations $a_t^{[0]}$ is 1284. The query image representation and neighbor community representations are stacked and used as input for the transformer. The inner workings of the transformer will be considered in the next subsection.

The preprocessed location coordinates are projected to a location representation vector $a_{location}$ of length 8. The scaled habitat variables are projected to a habitat representation $a_{habitat}$ of length 64. The preprocessed date variables are projected to a date representation a_{date} of length 8. All projections are performed using a dense layer with ReLU activation. In this way, possible patterns in the location, habitat and date data can be learned by the model. The approach of these projections is similar to that of Xu et al. (2023). Mathematically,

$$a_{location} = ReLU(W_{location}x_{location} + b_{location}) \tag{4}$$

$$a_{habitat} = ReLU(W_{habitat}x_{habitat} + b_{habitat}) \tag{5}$$

$$a_{date} = ReLU(W_{date}x_{date} + b_{date}) \tag{6}$$

where $W_{location} \in \mathbb{R}^{8\times2}$, $b_{location} \in \mathbb{R}^8$, $W_{habitat} \in \mathbb{R}^{64\times64}$, $b_{habitat} \in \mathbb{R}^{64}$, $W_{date} \in \mathbb{R}^{8\times2}$, and $b_{date} \in \mathbb{R}^8$ represent weights and biases.

The output of the transformer is the transformed query image representation $z_{image} \in \mathbb{R}^{1284}$. This vector is concatenated with the vectors representing location, habitat and date, creating a vector $z \in \mathbb{R}^{1364}$:

$$z = z_{image} \frown a_{location} \frown a_{habitat} \frown a_{date} \tag{7}$$

Finally, this vector is projected to an output vector that has length s_t, corresponding to the number of species in taxonomic group t. This projection is performed using SoftMax activation, to compute likelihood scores $\hat{y}$ for each class (species):

$$\hat{y} = softmax(W_{final}z + b_{final}) \tag{8}$$

with $W_{final} \in \mathbb{R}^{s_t\times1364}$ and $b_{final} \in \mathbb{R}^{s_t}$.

Separate models are made for each of the $|t|$ different taxonomic groups. A visualisation of the new model for the group Chordata is shown in **Figure 5**.

3.6 Transformer component structure

The query image representation x_{image} and community representations $a_t^{[0]}$ form the input of the transformer component, analogous to the image patches in the ViT network. First, these representations are projected to internal representations $a_{t'}^{[1]}$, with $t' \in \{image, arthropoda, chordata, diptera, fungi, insecta, lepidoptera, mollusca, plantae\}$:

$$a_{t'}^{[1]} = W_0 a_{t'}^{[0]} + b_0 \tag{9}$$

with $W_0 \in \mathbb{R}^{1284\times1284}$ and $b_0 \in \mathbb{R}^{1284}$. This layer of shared weights replaces the flatten layer in the ViT network. It has the potential to homogenize image and community representations, since weights are shared across the $|t| + 1$ representations.

Each of the internal representations $a_{t'}^{[1]}$ is then added to a taxonomy embedding $E_{t'}$, similar to position embeddings in the original ViT network:

$$a_{t'}^{[2]} = a_{t'}^{[1]} + E_{t'} \tag{10}$$

with $\forall t' : E_{t'} \in \mathbb{R}^{1284}$. These taxonomy embeddings have the same length as the image representations, namely 1284. That means the taxonomy embeddings form a layer of $1284 \cdot (|t| + 1)$ learnable parameters. The query image representation is added to its own 'taxonomy embedding'. In contrast to the ViT model, we do not perform dropout after this addition. The taxonomy embeddings are meant to enable the network to learn patterns that are particular for a taxonomic group.

The resulting sequence of vectors is then fed into a standard transformer encoder (Vaswani et al., 2017). The transformer consists of L layers. We use $L = 1$, to keep our model simple. We expect less layers to be needed than in the ViT network, since our transformer does not replace the convolutional process. For the architecture of the transformer, we mostly stick to the architecture proposed in the ViT model. This means each layer consists of layer normalization, followed by a multi-head attention (MHA) mechanism. This attention mechanism includes dropout, in contrast to the ViT model. Dropout is left out of all mathematical equations in this text. Then, the outputs of this mechanism are added to the input of the transformer layer from before layer normalization, creating a residual connection:

$$[a_{image}^{[3]}; ...; a_{plantae}^{[3]}] = LN([a_{image}^{[2]}; ...; a_{plantae}^{[2]}]) \tag{11}$$

$$[a_{image}^{[4]}; ...; a_{plantae}^{[4]}] = MHA([a_{image}^{[3]}; ...; a_{plantae}^{[3]}]) \tag{12}$$

$$a_{t'}^{[5]} = a_{t'}^{[4]} + a_{t'}^{[2]} \tag{13}$$

with $LN(x)$ representing layer normalization and $MHA(x)$ representing the multihead attention mechanism. Our multihead attention mechanism includes 4 heads and the dimensionality of key and query projections is 2.

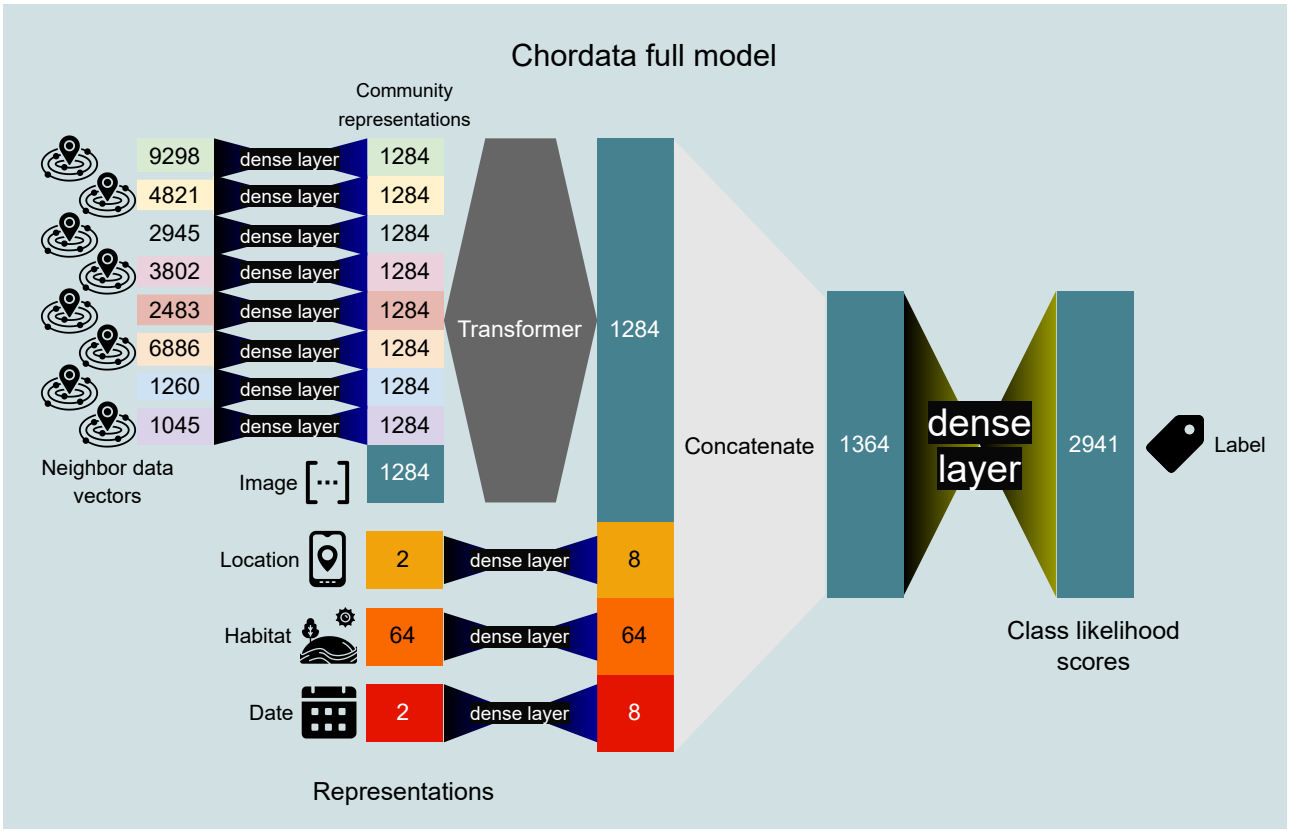


Figure 5: The architecture of one of our new models, that replaces the Chordata classification head of the current model. The inputs that are put through the model flow from left to right. Colored rectangles with numbers represent vectors and their lengths. The pastel colored vectors represent neighbor data and community representations. The teal colored vectors represent the image representation vector in different steps of the process. The yellow, orange and red vectors represent location, habitat and date vectors respectively. The black connections with colored sides represent dense layers with different activation functions: blue for ReLU and yellow for SoftMax. The gray hexagonal shape represents the transformer component. The internal structure of the transformer component is visualized in **Figure 6**. The light gray trapezoid represents concatenation. Icons are used for the neighbor data, image representation, location, habitat, date and label. Icons by Aranagraphics ('nearby'), Lizel Arina ('array'), Vectors Tank ('gps phone'), Nurfajri Aldi ('ecosystem'), Prosymbols Premium ('calendar'), and edt.im ('label') on flaticon.com.

These values were determined experimentally. The different heads and key/query dimensions enable different types of attention patterns to be learned. We expect the relationship between neighboring species and query species, as well as between species to be relatively simple (do these species co-occur?). So, we also expect these small hyperparameter values for the number of heads and key/query dimensions to be large enough.

This block of normalization, attention and a residual connection is then repeated, but with a multi-layer perceptron (MLP) block instead of a MHA mechanism. The MLP block contains two dense neural network layers with Gaussian Error Linear Unit (GELU) activation and a dropout layer after each dense layer:

$$[a^{[6]}_{image}; ...; a^{[6]}_{plantae}] = LN([a^{[5]}_{image}; ...; a^{[5]}_{plantae}]) \tag{14}$$

$$a^{[7]}_{t'} = GELU(W_6 a^{[6]}_{t'} + b_6) \tag{15}$$

$$a^{[8]}_{t'} = GELU(W_7 a^{[7]}_{t'} + b_7) \tag{16}$$

$$a^{[9]}_{t'} = a^{[8]}_{t'} + a^{[5]}_{t'} \tag{17}$$

with $GELU(x)$ representing GELU activation, $W_6 \in \mathbb{R}^{2568 \times 1284}$, $b_6 \in \mathbb{R}^{2568}$, $W_7 \in \mathbb{R}^{1284 \times 2568}$, and $b_7 \in \mathbb{R}^{1284}$. Again, these dense layers share weights across representations.

After the transformer encoder layer(s), a final layer normalization is applied, followed by a final dropout layer:

$$[a^{[10]}_{image}; ...; a^{[10]}_{plantae}] = LN([a^{[9]}_{image}; ...; a^{[9]}_{plantae}]) \tag{18}$$

The transformed image representation is then used in the final part of the model, described in the previous section:

$$z_{image} = a_{image}^{[10]} \tag{19}$$

The transformed community representations are not further used in the model.

To summarize, the adjustments we made to the ViT transformer module are:

- No dropout is performed after adding the taxonomy embeddings.
- Dropout is performed within the MHA block.
- A dropout layer is added at the very end of the transformer component.

The dropout rate for the final layer is 0.5. The dropout rate for internal dropout layers within the MHA and MLP block is 0.1.

The internal structure of the transformer component is visualized in **Figure 6**.

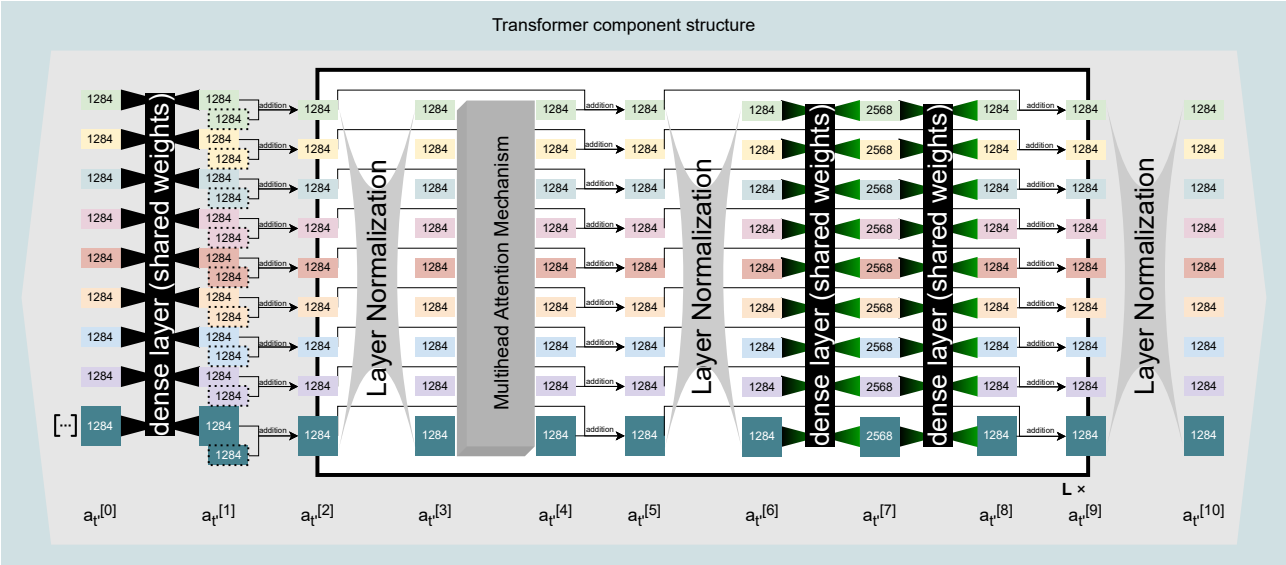


Figure 6: The internal architecture of the transformer component. Compare with the visualization of the transformer encoder component in the ViT model (Dosovitskiy et al., 2020). Only one of L layers is shown. The inputs that are put through the model flow from left to right. Colored rectangles with numbers represent vectors and their lengths. The pastel colored vectors represent processed community representations in different steps of the process. The teal colored vectors represent the image representation vector in different steps of the process. The mathematical notation for the representation vectors is provided in the bottom of the figure. The black connections with colored sides represent dense layers with different activation functions: black for no activation and green for GELU. The rectangles with dotted borders represent learnable taxonomy embeddings. The arrows represent addition, including skip connections. The grey web-like shape represents layer normalization. The grey block represents the multihead attention mechanism. The white box represents the part of the model that is iterated when multiple layers L are used. Dropout is not visualized.

3.7 Training

First, the main convolutional model and the convolutional submodels are trained separately. This is done without the transformer component: each submodel is connected with its specific classification head. For the main model, the input for each training instance is a single image and the output is a label corresponding to one of the submodels t. The model is trained using the actual taxonomic groups for each image as ground truth. For the convolutional submodels, the input for each training instance is a single image of that subgroup and the output is a vector of length s_t, where s_t is the number of species in the particular taxonomic group t. That vector represents the likelihood that the image contains each of s_t different species. The submodels are trained using the true class labels for each image as ground truth. The parameters learned in the classification heads are not used in the final model, as the classification heads are replaced by the new models. After training the main convolutional model and the submodels, the parameters of these models are frozen.

After training the main convolutional model and the submodels, the output vectors (image representations) of all images before the classification heads, which each have length 1280, are calculated and saved.

Then, new models are trained. The input for each training instance consists of:

- a query image representations from a single observation.
- $|t|$ data vectors of neighbor abundances around the location of the query observation, that each have s_t values summing to k.
- a location vector
- a habitat vector
- a date vector

Using the taxonomic subgroup of the query observation, the correct one of $|t|$ new models is selected and trained. The output of the model during training is a vector of length s_t, that represents the likelihood that the observation contains each of s_t different species. The new model is trained using the true class labels as ground truth. The loss is backpropagated through the final dense layer to the location, habitat and date projections and through the transformer to the taxonomy embeddings and the taxonomy-specific projections.

During training, we use batches of 1024 observations. This number is chosen for technical reasons to do with efficiently loading the data from memory. Because this batch size is quite large, we experimented with different learning rates and chose a learning rate somewhat higher than used in the ViT model. The initial learning rate lr_1 is set at $lr_1 = 0.002$. Each epoch, the learning rate is decreased using the formula $lr_{n+1} = lr_n \times e^{-0.1}$. We use Adam optimization. Training continues for 20 epochs, after which the model version with the highest validation accuracy is selected.

3.8 Inference

Inference is similar to training. During inference, the input to the complete model consists of:

- a query image representations from a single observation.
- $|t|$ data vectors of neighbor abundances around the location of the query observation, that each have s_t values summing to k. These are retrieved using a k-nearest neighbor algorithm.
- a location vector, created by processing the coordinate values
- a habitat vector, selected based on the coordinate values
- a date vector, created by processing the date

The output of the main convolutional model using the query image as input is used to determine to which taxonomic subgroup the query species belongs. Based on this determination, a convolutional submodel and corresponding transformer model is selected. Using the query image as input, the submodel computes a query image representation. From all inputs, a vector of length s_t is calculated by the model. This vector represents the likelihood that the observation contains each of s_t different species within taxonomic subgroup t.

3.9 Performance hypothesis

We expect the model to make more accurate predictions because it can use the ecological information inherent in the distribution of neighboring species that are contained in the community representations. This information can be used to adjust the query image representation according to the local and ecological context of the observation. That is why we expect the model to improve accuracy in areas with a particular ecological context, eventually improving the geographical and taxonomical balance in performance. Since the number of neighbors used in the neighbor data vector is fixed, the algorithm automatically adjusts the size of the neighborhood it takes into account, which will depend on the local density of occurrence records for each taxonomic group. This approach to defining neighborhoods is similar to that of Berg et al. (2014). Because of this feature, we expect the model can perform with comparable accuracy in areas with both densely and sparsely distributed observations. Since rare species, which have low numbers of occurrence records, are often limited to a local distribution range, the performance on rare species will profit more from local context information than the performance of widespread species. That is why we expect that our model improves the performance on rare species as well, leading to a more balanced performance across species.

Although we expect the transformer component to increase model performance, we evaluate the additional computational requirements so a cost-benefit analysis can be made. Transformer encoder modules generally demand a lot of computational power and/or time. For this reason, we compare the model's performance and the model's computational requirements with baseline models that are less complex but also less powerful. Which model is ultimately most appropriate will depend on the available computational resources of the user.

3.10 Ablation studies

We will compare the performance of our transformer model with a number of ablated models. These ablations are described here.

3.10.1 Model I

The first ablation is our baseline model, called model I (for image). It requires only an image as input and is identical to the current model described in **3.4**. It does not model location or ecological context whatsoever and has no transformer component. It is composed of a single dense layer with SoftMax activation, described in **Equation 1**. The purpose of this baseline is to compare models that do not use any additional information besides an image with models that do use additional information as input. In this way, we can measure the effect of adding contextual information (either location, habitat, date or neighboring observations). Significance tests are performed between this model and the full model.

3.10.2 Model ILHD

The second ablation is an extension of our current model, inspired by the authors using geographical and external features as input to the model (Chu et al., 2019; Tang et al., 2015; Terry et al., 2020; Xu et al., 2023; Yang et al., 2022). It is called model ILHD to reference the image, location, habitat and date variables used as input. It is similar to the full model, but does not include neighbor data, nor a transformer component. Mathematically, **Equation 7** is replaced with:

$$z = x_{image} \frown a_{location} \frown a_{habitat} \frown a_{date} \tag{20}$$

The purpose of this model is to measure the added benefit of using location, habitat and date information as input to the current (baseline) model. Significance tests are performed between this model and model I.

3.10.3 Model T(IN)

The third ablation is called model T(IN) to reference the transformer component and the image and neighbor inputs. It is similar to the full model, but the location, habitat and date variables are removed as input to the final layer. Mathematically, **Equation 7** is replaced with:

$$z = z_{image} \tag{21}$$

In this way, we can measure the added benefit of using location, habitat and date information as input to the full model. Significance tests are performed between this model and the full model.

3.10.4 Model IN

The fourth ablation is called model IN. It is similar to model I, but uses neighbor data as input. However, it does not include a transformer component. Community representations are concatenated with the image representation and a final dense layer with SoftMax activation is applied. Mathematically, **Equation 7** is replaced with:

$$z = x_{image} \frown a^{[0]}_{arthropoda} \frown a^{[0]}_{chordata} \frown a^{[0]}_{diptera} \frown a^{[0]}_{fungi} \frown a^{[0]}_{insecta} \frown a^{[0]}_{lepidoptera} \frown a^{[0]}_{mollusca} \frown a^{[0]}_{arthropoda} \tag{22}$$

The purpose of this model is to measure the added benefit of adding neighbor data to the baseline model. Furthermore, the model may be used to measure the added benefit of the transformer component. This model does use neighbors as input, but does not transform any representations. That is why it may be compared fairly with the model T(IN) to measure the effect of using a transformer component. Significance tests are performed between this model and both models I and model T(IN)

3.10.5 Model INLHD

The fifth ablation is called model INLHD. It is similar to both model IN and model ILHD. Representations for the image, the ecology, the location, the habitat and the date are concatenated and a dense layer with SoftMax activation is applied. Mathematically:

$$z = x_{image} \frown a^{[0]}_{arthropoda} \frown a^{[0]}_{chordata} \frown a^{[0]}_{diptera} \frown a^{[0]}_{fungi} \frown a^{[0]}_{insecta} \frown a^{[0]}_{lepidoptera} \frown a^{[0]}_{mollusca} \frown a^{[0]}_{arthropoda} \frown a_{location} \frown a_{habitat} \frown a_{date} \tag{23}$$

This model can be compared with the full model to measure the effect of using a transformer component. It can also be compared with the ILHD model to measure the effect of adding neighbor data. It can also be compared with the IN model to measure the effect of adding location, habitat and date information. Significance tests are performed for all of these comparisons.

Using these model names, the full model could be called 'T(IN)LHD'. This model includes all elements mentioned in the ablation models. A visualization of the ablation models, similiar to the visualization of the full model in **Figure 5** is shown in **Figure 7**. The number of parameters for each model is given in **Table 2**. An overview of the different effects we measure is given in **Table 3**.

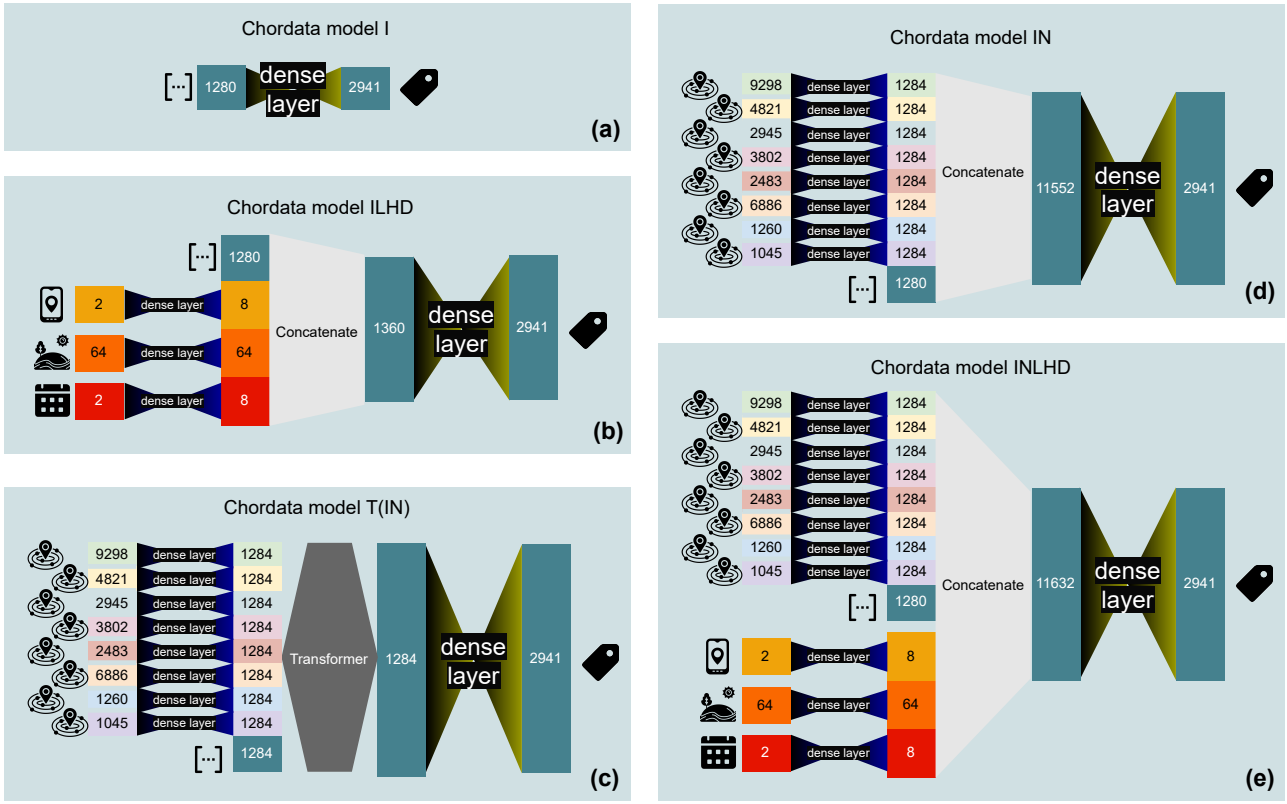


Figure 7: The architecture of our ablation models for the taxonomic group Chordata. The inputs that are put through the model flow from left to right. Colored rectangles with numbers represent vectors and their lengths. The pastel colored vectors represent neighbor data and community representations. The teal colored vectors represent the image representation vector in different steps of the process. The yellow, orange and red vectors represent location, habitat and date vectors respectively. The black connections with colored sides represent dense layers with different activation functions: blue for ReLU and yellow for SoftMax. The gray hexagonal shape represents the transformer component. The internal structure of the transformer component is visualized in **Figure 6**. The light gray trapezoid represents concatenation. This figure can be compared with **Figure 5**, in which the full model is visualized. Icons are used for the neighbor data, image representation, location, habitat, date and label. **a)** model I. **b)** model ILHD. **c)** model T(IN). **d)** model IN. **e)** model INLHD. Icons by Aranagraphics ('nearby'), Lizel Arina ('array'), Vectors Tank ('gps phone'), Nurfajri Aldi ('ecosystem'), Prosymbols Premium ('calendar'), and edt.im ('label') on flaticon.com.

3.10.6 Performance hypotheses for ablations

We expect models with more different inputs to outperform models with less inputs on most of our evaluation metrics. We also expect models with a transformer module to outperform similar models without a transformer module. If our full model outperforms all other models on most or all evaluation metrics, we are able to say that our new proposed architecture is a useful improvement upon the baseline models.

However, we simultaneously expect the performance improvement of the full model upon and model T(IN) to be limited, since both integrating location/habitat/date variables and using the neighborhood species distribution are a way to model the ecological context of the query species. This means combining both methods may create redundancies, reducing potential performance improvement.

Subgroup\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Arthropoda	1.6	1.7	56.3	56.4	51.7	51.8
Chordata	3.8	4.0	75.8	76.0	53.9	54.1
Diptera	3.2	3.4	70.4	70.6	53.3	53.5
Fungi	6.2	6.6	97.4	97.8	56.3	56.7
Insecta	8.8	9.4	121.3	121.9	58.9	59.5
Lepidoptera	4.9	5.2	85.7	86.0	55.0	55.3
Mollusca	1.3	1.4	53.8	53.9	51.4	51.5
Plantae	11.9	12.7	149.2	149.9	62.0	62.8

Table 2: Number of parameters for each model, in millions. The number depends on the model structure and the number of taxonomy-specific classes.

Effect\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
adding LHD input	✓	✓				
adding LHD input			✓	✓		
adding LHD input					✓	✓
adding N input	✓		✓			
adding N input		✓		✓		
adding transformer			✓		✓	
adding transformer				✓		✓
replace model	✓					✓

Table 3: Models to compare for different effect measurements. Statistical tests will be performed for each row in the table between the two selected models.

3.11 Evaluation metrics

To compare models, we use four evaluation metrics that capture information about performance in relation to different kinds of imbalance in the data:

- **Accuracy**: The percentage of correctly identified observations out of all observations. This is a measure of general global model performance.
- **Average recall**: The (macro-)average recall across all classes (species). Recall for species X is the percentage of observations correctly identified as X out of all observations that are truly species X. This metric is a measure for balanced performance across species. Note that species with few occurrences (rare species) have the same weight in this metric as species with many occurrences (common species). In this way, the overrepresentation of common species in the data is neutralized in this metric.
- **Average local accuracy** (ALA): The (macro-)average accuracy across all areas. An area is defined as a rectangular space of 5 degrees latitude by 5 degrees longitude. This metric is a measure for balanced performance in different areas. Areas with little data have the same weights in this metric as areas with a lot of data. In this way, the overrepresentation of data-dense areas is neutralized in this metric. Areas with very little data (fewer than 100 observations in the test set) are left out of this metric, to avoid too much noise in the data.
- **Average local average recall** (ALAR): The average of the average recall across all areas. As a combination of the two metrics above, this is a measure of balanced performance across species across different areas. Again, areas with fewer than 100 observations are left out.

In some cases, we will also report a metric dedicated to reducing geographical bias:

- **Variance in local accuracies** (VLA): The variance in accuracy across all areas. Areas are defined in the same way as for ALA, and again areas with fewer than 100 observations are left out. This metric measures directly how similar local accuracies are. Low values are considered better, as we aim to reduce geographical bias and thus homogenize local accuracies.

Furthermore, we use two metrics that capture information about computational requirements:

- **Training time**: The time it takes to train all taxonomic submodels for 20 epochs on a Intel Xeon Gold 6342 processor using 4 or 20 cores, in hours.
- **Inference time**: The average time it takes to predict a single instance using the trained model, in milliseconds.

4 Results

In this section, we report the results of training and evaluating our models. We start by reporting the performance of the trained models and the effect sizes of different changes in the model architecture. We finish by reporting training and inference times of the models.

4.1 Model performance

We trained the full model and all ablations on all taxonomic data subsets. In total, 48 models were trained and evaluated. The performance of all these models on the four performance metrics mentioned in the previous chapter can be found in **Appendix A**.

We combined the test results of the different taxonomic subgroups, to get a picture of the performance of each architecture over all data. The performance metrics were calculated over the combined test results and can be found in **Table 4**. The macro-average precision of the different model is also reported. Note that these combined results are not the macro-average of the results from eight taxonomic models. Rather, taxonomic subgroups with more data are weighted more heavily in the calculation of the combined results. The full model scored highest on all four metrics.

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	84.47	85.26	84.30	84.49	85.16	**85.95**
Average Recall	46.89	48.23	48.58	48.88	48.68	**50.02**
ALA	78.50	80.48	79.95	80.06	79.53	**81.42**
ALAR	71.22	72.73	71.88	72.08	72.41	**73.83**
Average Precision	69.77	70.41	70.72	**70.80**	68.26	69.03
VLA	3.29e-3	2.28e-3	2.41e-3	2.29e-3	2.87e-3	**2.01e-3**

Table 4: Combined results from all taxonomic models. The numbers indicate model metrics on all test data, using the trained model from the taxonomic group corresponding to the data instance. The different columns indicate different model architectures (ablations) and rows indicate different performance metrics. The highest score for the performance metrics and the lowest score for the bias metric are made bold.

All models were trained for 20 epochs. After each epoch, validation accuracy was calculated. The version of the model with the highest validation accuracy was selected for testing the model. In **Table 5**, the number of epochs from each selected model version can be found.

Subgroup\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Arthropoda	4	6	5	7	20	20
Chordata	20	20	2	3	20	16
Diptera	18	20	12	5	11	15
Fungi	18	18	1	1	17	13
Insecta	20	19	2	2	16	20
Lepidoptera	19	19	1	2	18	13
Mollusca	15	17	6	3	16	16
Plantae	20	20	3	3	20	20

Table 5: Number of trained epochs before peak performance was reached on the validation set for each model. After training for 20 epochs, the model version with the highest validation accuracy was selected for testing.

4.2 Effect measurements

For each taxonomic subgroup, we performed eight pairwise comparisons between two models (ablations). For each comparison, we first calculated the McNemar test statistic on correctly/incorrectly predicted test samples, and the corresponding p-value. The results of all comparisons can be found in **Appendix B**. Then, we calculated statistics for the paired recalls on all classes, the paired local accuracies and the paired local average recalls. For these tests, we used the Wilcoxon signed rank test, since the data is paired and may not be normally distributed. Since we made multiple comparisons, we determined significance by applying the Holm-Bonferroni method. We

sorted all comparisons by their p-values in increasing order. We used $\alpha = 0.05$ and $m = 256$, because we make 256 comparisons. The significance threshold is calculated as $\frac{\alpha}{m-c}$, where c is the rank of the test in the sorted list, starting at 0. All statistical tests were two-sided. An effect is considered significant if the p-value is lower than the calculated threshold. The complete procedure for determining thresholds and significance presences can be found in **Appendix C**. In total, 125 out of our 256 tests resulted in a significant outcome, after the correction made via de Holm-Bonferroni procedure. For all these 125 effects, $p < 3.8e-4$. In **Table 6**, an overview is provided of the number of positive and negative effects that were found to be significant. These counts vary from zero (significant for no taxonomic groups) to eight (significant for all eight taxonomic groups).

Model 1	Model 2	Change	Metric	Significant positive effects	Significant negative effects
I	ILHD	adding LHD input	Accuracy Avg. Recall ALA ALAR	5 6 4 4	
IN	INLHD	adding LHD input	Accuracy Avg. Recall ALA ALAR	1 1	
T(IN)	Full Model	adding LHD input	Accuracy Avg. Recall ALA ALAR	5 4 4 3	
I	IN	adding N input	Accuracy Avg. Recall ALA ALAR	4 7 2 2	2 1 1
ILHD	INLHD	adding N input	Accuracy Avg. Recall ALA ALAR	3 5 1	3 1 1 1
IN	T(IN)	adding transformer	Accuracy Avg. Recall ALA ALAR	4 2 1 1	3 2 1 1
INLHD	Full Model	adding transformer	Accuracy Avg. Recall ALA ALAR	4 3 3 3	2 2
I	Full Model	replace model	Accuracy Avg. Recall ALA ALAR	6 8 4 4	

Table 6: Counts of significant positive and negative effects across eight taxonomic groups. This table is a summary of the tables in **Appendix B**. If no number is shown, the number of significant effects found was zero. For accuracy, significance was determined using the McNemar test statistic calculated over the numbers of observations predicted (in)correctly by the two compared models. For other metrics, the Wilcoxon signed rank statistic was used, calculated over all class recalls, or over local accuracies or local average recalls with more than 100 observations. Significance levels were determined by performing a Holm-Bonferroni correction, with the least strict threshold being $p < 3.76e-4$. In total, 125 significant effects were found: 104 positive effects and 21 negative effects. Some changes induced only positive or only negative significant effects. Other changes showed different behavior for different taxonomic groups.

4.3 Training time

We computed training times for different models using two settings with different amounts of computational resources. In the first setting, training times were computed on a CPU using 4 cores (8 hyperthreads), in the

second setting we used 20 cores (40 hyperthreads). In both cases, the CPU we used was the Intel Xeon Gold 6342 Processor working at 2.80GHz. In each case, training time is defined as the time to train the model for all taxonomic groups for 20 epochs. Training times can be found in **Table 7**.

On limited computational resources (8 CPUs), the addition of neighbor information and the transformer architecture both substantially increased training time. When more CPU cores were used, all training times decreased. However, the more complex models benefited much more from these extra CPU cores. This resulted in training times for complex models that are only somewhat larger than those for the baseline model.

It should be noted that all these training times pale in comparison with training times for the convolutional part of the complete model. Those training times were not included in our calculations but would be much greater than training times for our models, that used image representations as input. Training times for the convolutional part of the model are in the range of many weeks, even when training on GPUs.

CPUs\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
8	187	184	352	337	971	958
40	177	175	189	177	222	219

Table 7: Training times of training all taxonomic groups for 20 epochs, in hours.

4.4 Inference time

Additional inference time for larger models was limited. The inference time of a single forward pass using the baseline model was 98 ms. This time was calculated for the Chordata submodel as an average of a 100 singular measurements on a machine with 8 CPU hyperthreads. Model ILHD, that included representations of location, habitat and date had an inference time that was roughly 1 ms greater. Models IN and INLHD, that included representations of neighboring species, had an inference time that was roughly 32 ms greater than the baseline. Models T(IN) and the full model had inference times that were roughly 18 ms greater than the baseline. For other taxonomic groups, inference times may differ, since the number of model parameters is different. Note that the convolutional part of the complete model is not taken into account in these calculations, but will not differ between the different model architectures we used.

Besides the forward pass, some models retrieved more data and transformed it, which took additional time. Retrieving the 800 neighbors using a KD-tree for models IN, INLHD, T(IN) and the full model took around 15 ms. A small addition to inference time should also be expected for retrieving habitat information, and transforming location and date information. Overall, we conclude that additional inference time was limited and should not hinder large-scale application of our models.

5 Analysis

In this section we analyse the results of one taxonomic submodel further, so we can more thoroughly evaluate its performance without needing to combine results from different models. We chose to analyse the results for the subgroup Chordata, as it contains both birds and mammals, the two taxonomic groups that have drawn the most scientific attention in citizen-science related studies (Feldman et al., 2021). From the model performances in **Table 11**, we can see that model INLHD scores highest on all four performance metrics. Therefore, we analyse the difference between this model and the baseline model (model I) further. Note that model INLHD outperforming the full model is a deviation from most other models, as the full model performs best for most taxonomic groups.

The overall improvement of model INLHD on the baseline model was 1.13 percent point accuracy, 4.89 percent point average recall, 3.77 percent point ALA and 2.35 percent point ALAR. In **Table 19**, the effects of different changes to the model are decomposed. Although not all effect measurements are statistically significant, the effects paint a clear picture. Both adding location, habitat and date information and adding neighborhood species information have a positive effect on performance, while employing the transformer architecture has a negative effect. This is the case for all metrics in all comparisons in the Chordata subgroup.

5.1 Recall improvement and reducing abundance bias

Model INLHD improved the average recall rate of the baseline model substantially. The recall rate was quite different for different classes (species). Since classes with a lot of observations (representing common/abundant species) provided more training data, the recall for these species tended to be higher. The recall for classes with fewer observations (rare species) was lower, because fewer training data could be used to recognize these species. The relation between class recall and number of observations for a class is visualized in **Figure 8**. In this figure, we can also see how the average recall differs between the baseline model and model INLHD. To visualise how the accuracy and average recall depend on class size (number of observations in a class), we plotted the running accuracy and running average recall. To calculate these metrics, classes are ranked (sorted) by size. Class 1 has the most observations, class 2 has the second most observations, etc. Model performance is first evaluated only on test data of class 1. Then, performance of the models is evaluated iteratively using test data of one additional class each time. The running accuracy can be plotted as the overall accuracy of the model at each iteration. In the final iteration, performance on the complete test set is evaluated. Since classes with many observations are evaluated first and have higher recalls, the accuracy and average recall 'drop' during iteration progression.

We see that the running average recall is very similar between models when only classes with many observations are taken into account: the blue and yellow solid lines overlap. We also observe that average recall does not improve very substantially between models for the classes with very few observations: the blue and yellow solid lines run in parallel. The improvement of the average recall mainly seems to originate from the classes with a medium number of observations.

5.1.1 Correlation class size and class recall

Since recall is higher for classes with many observations, there is more room for improvement (error) for classes with few observations. Indeed, classes with more observations show lower recall improvements, which can be seen in **Figure 9**. The linear correlation between recall improvement and number of observations in a class is hard to observe in **Subfigure 9a**. It is also statistically weak (Pearson r statistic: -0.0855, p-value 4.63e-05). However, this has largely to do with the large variance of recall improvements among classes with few observations. To neutralize this effect, we grouped the 2263 classes in the test set into 31 bins, each containing 73 classes, in order of their number of observations. These numbers were chosen because they are the prime factors of the number of classes in the Chordata test set, so that each bin has the same size. Now, we observe a clear pattern in **Subfigure 9b**: classes with fewer observations show larger improvements. This correlation is also much stronger statistically (Pearson r statistic: -0.477, p-value 6.67e-03).

5.1.2 Reducing abundance bias

The larger performance improvements for classes with few observations already point to a reduction of abundance bias. Comparing accuracy and average recall tells us more about reduction of abundance bias. This is because average recall is mostly determined by the recall of rare species, which there are more of than common species. In contrast, accuracy is mostly determined by the recall of common species, as these provide many more observations. So, one could argue that a higher average recall implies reduced abundance bias if the improvement is larger than the improvement in overall accuracy. If the improvement in accuracy and the improvement in average recall is the same, the improvement may have been independent of class size (species abundance). In this sense, model INLHD reduces abundance bias, since the improvement in average recall is larger than the

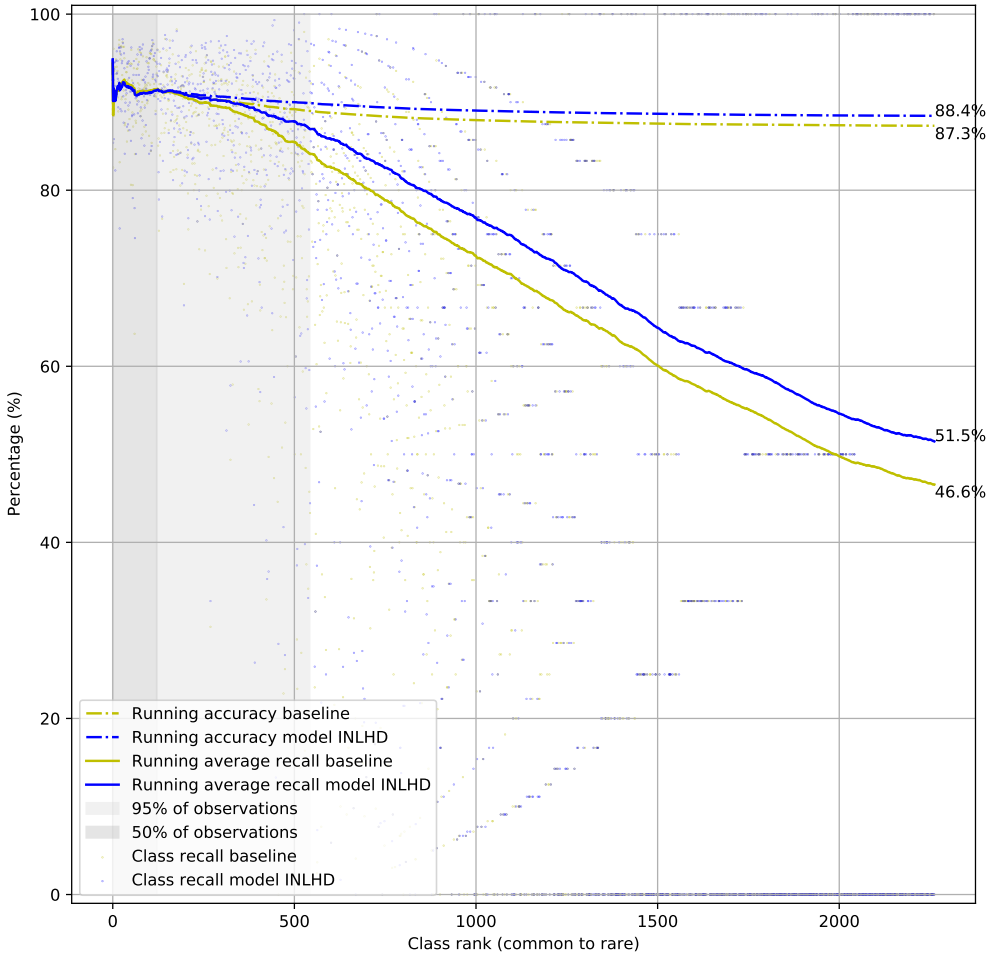


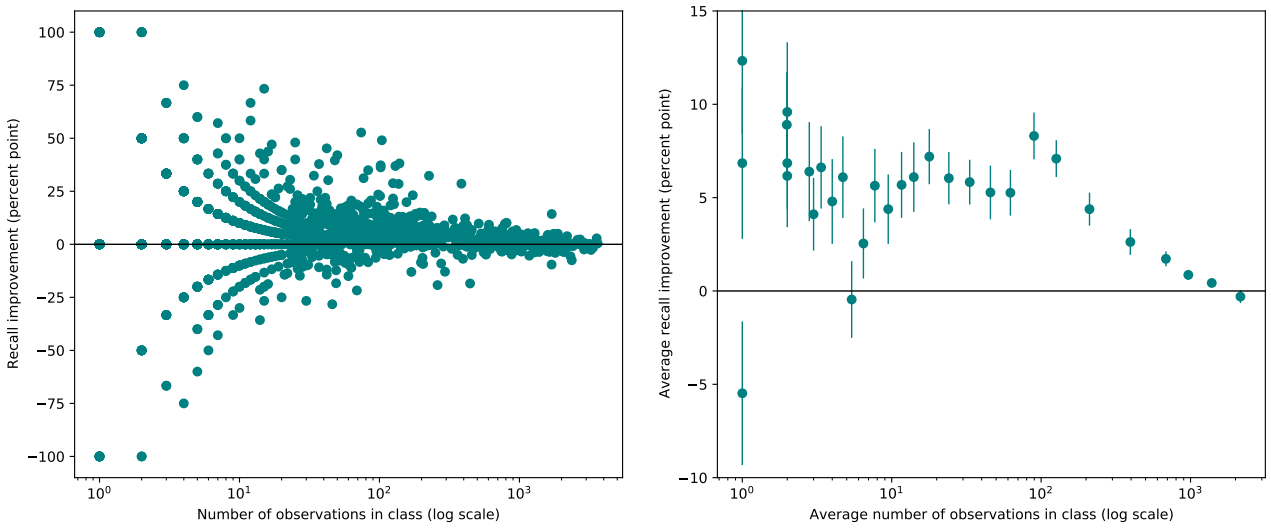
Figure 8: Relationship between class recall and class rank (lower ranks indicate fewer observations). Class recall is shown for both the baseline model (yellow semi-transparent dots) and model INLHD (blue semi-transparent dots). The running accuracy is shown with a dash-and-dot line in yellow for the baseline model and in blue for model INLHD. The same was done for the model average recall, which are shown with solid lines. The concept of running accuracy and running average recall is explained in the main text. The final (overall) accuracy and average recall for both models are given at the end of the plotted lines. The dark grey rectangle signifies the classes that make up 50% of the test data. Both dark grey and light grey rectangles together signify the classes that make up 95% of the test data.

improvement in accuracy. The conclusion that the improvement was not independent of class size is supported by the correlation between recall improvement and number of observations in a class described in the previous subsection.

However, the improvement in average recall should also be compared with the improvement in accuracy in a relative sense. The accuracy is higher to begin with, since it is mainly determined by the majority classes that have high recall rates. Since the error for these classes is smaller, there is less room for improvement of the accuracy rate. To analyse the relative improvement, we calculated the relative reduction of the error. For accuracy, the error on the baseline model is 12.7 percent point, while the improvement by model INLHD is 1.1 percent point. This means 8.7% of the error is reduced. For average recall, the error on the baseline model is 53.4 percent point, while the improvement using model INLHD is 4.9 percent point. This means 9.2% of the error is reduced. The error reduction in average recall is larger than the error reduction in accuracy. So even in this more conservative relative sense, model INLHD slightly reduces abundance bias for the Chordata subset. Note that this is not necessarily the case for all other taxonomic subgroups.

5.2 Species recall improvements

To analyse for which classes the recall rate improved, we created **Figure 10**. In this bubble plot, the recalls for each class are compared between the baseline model and model INLHD. Each 'bubble' data point corresponds



(a) *Individual classes*

(b) *31 bins of 73 classes each*

Figure 9: Relationship between model improvement and number of observations in the test set of a class. All classes in the test set of the Chordata submodel were used. The improvement is defined as the increase in class recall from using the baseline model to using model INLHD. **a)** Individual classes. Each class is represented by a teal dot. **b)** Bins of classes. Classes were divided into 31 bins that contained 73 classes each. Bins were determined by the number of observation in each class. Each teal dot represents a bin, showing the average number of observations in the test set and average recall of classes in each bin. Bars are shown representing the standard error of the mean for each bin. Since a lot of classes only contain 1 or 2 observations, multiple bins have an average of 1 or 2 observations.

to a class (species) and the size of a data point indicates class size. A number of observations can be made for this figure:

- The models show quite different behavior. Many classes in model INLHD have a recall rate that is substantially higher or lower than in the baseline model. This different behavior between models is supported by other results. Out of all observations in the test set, the baseline model and model INLHD made the same prediction in only 78.7 percent of cases. This is surprisingly low, considering both models reached accuracies that were very similar and above 87 percent. Of all mistakes made by one of the models, only 12-13 percent was also made by the other model.
- The recall of many abundant classes seems (nearly) equal between models. This is no surprise, as the overall increase in average recall is primarily caused by the large number of less abundant classes (rare species) that had a low baseline recall and have a higher recall in model INLHD (see **Figure 8**).

If we consider some individual species, we might be able to explain why its recall rate improved. For example, take the class *Limosa limosa islandica*. This class represents a subspecies of *Limosa limosa*. This class counts 403 observations in the test set and its recall increased from 49.4% (baseline) to 57.8% (model INLHD). There are two biological insights that can help us explain this substantial increase:

- Birds labeled as the subspecies *Limosa limosa islandica* are visually very similar to birds labeled as the species *Limosa limosa*. Individuals of the subspecies may be identified by its slightly shorter beak and feet or its slightly darker red-brown underside that continues further down, differences hard to distinguish for both humans and image models. The reader may try comparing the two classes using **Figure 11**. This can result in similar image representations that are hard to classify for our baseline model, that only uses image representations for classification.
- There are areas where *L l islandica* is much more abundant than *L limosa*. In fact, the name *islandica* refers to its presence in Iceland. When an observation is from Iceland, this can be a good indicator the observation may be an individual of *L l islandica*. Information on geographical ranges is available to model INLHD, but not to the baseline model. Model INLHD performs much better than the baseline on observations from Iceland, because this location is strongly linked to a single class within the genus *Limosa*. This can be seen in **Figure 12**.

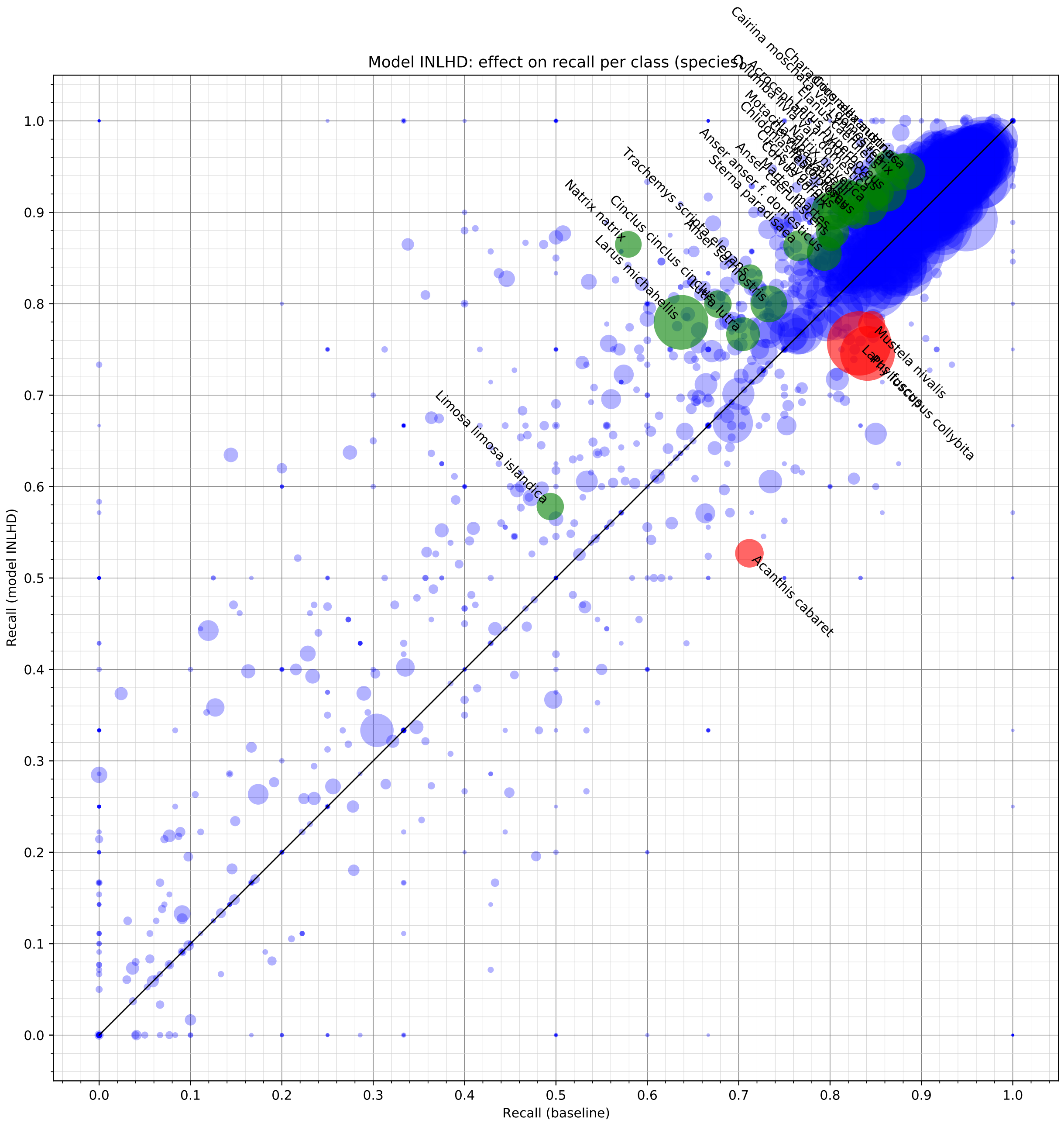


Figure 10: Relationship between class recalls in model INLHD and the baseline model I. Each bubble represents one class (species) and its size indicates the number of class observations in the test set. Species with at least 300 observations and an average recall at least 6 percent point higher in model INLHD compared to the baseline are made green and are labeled. Species with at least 300 observations and an average recall at least 6 percent lower in model INLHD compared to the baseline are made red and are labeled. Other species are made blue. The black diagonal line represents the points were average recall is equal in both models.

- The distribution of observations from *L l islandica* over the months of the year is different from that of other species in the genus *Limosa*. In **Figure 13**, these distributions are visualised. Again, the model INLHD can use this information to make predictions with higher accuracy. For example, if an observation occurs in February, March or April, the chances of the observation being *L l islandica* are much larger than in other months.

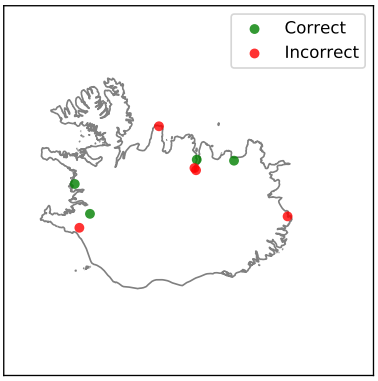
5.3 Genus recall improvements

For some classes, it is unclear if an improvement in recall is really caused by a change in the model. For example, The class *Larus michahellis*, with 1702 observations in the test set, increased in recall from 63.7% (baseline) to 78.0%. However, similar classes (often species from the same genus) decreased in recall. For example, the

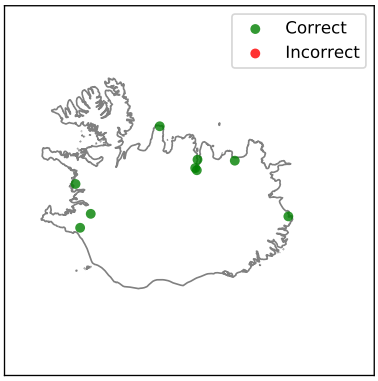
(a) *Limosa limosa*

(b) *Limosa limosa islandica*

Figure 11: Pictures of birds from (sub)species within the genus *Limosa*. Pictures by christoph_moning (a) and theworldofpaolo (b) on iNaturalist.



(a) Baseline model



(b) Model INLHD

Figure 12: A map of predictions of observation in the genus *Limosa* in Iceland. The baseline model **(a)** makes more errors than model INLHD, which can learn patterns between location and class likelihood.

class *Larus fuscus*, with 2366 observations in the test set, decreased from 83.2% to 75.7% recall. Since these species are visually similar, the recall improvement for one class could be realised at the cost of the recall rate of another class.

The model INLHD behaved differently than the baseline and may have coincidentally shifted its bias from one class to another. To deal with this stochastic behavior change, we grouped the classes into genus groups. This is shown in **Figure 14**. Usually, most of the errors an image recognition model makes are errors between visually similar objects. In our case, visually similar species are most probably members of the same genus. Since similar classes are grouped, the erratic behavior changes between models was neutralized. In the figure, we can now see that most genera either had an increased recall in model INLHD or had the same recall compared to the baseline. No more red labeled genera are present. We calculated genus recalls by combining and weighting the recalls of the classes in a genus. The average genus recall increased from 61.39% (baseline) to 64.34% (model INLHD). For the genus *Larus*, the genus recall increased from 78.4% (baseline) to 80.6% (model INLHD), which is a substantial but below-average increase.

A genus that does stand out is *Natrix*. Its genus recall increased from 72.5% (baseline) to 86.3% (model INLHD). None of the eight *Natrix* classes decreased in recall, and several increased substantially. For Natrix, the main reason for its higher performance is probably the combination of its visual similarity among classes and its different spatial ranges between classes, described in **1.1.3**.

5.4 Local accuracy improvement and reducing geographical bias

Model INLHD improved the average local accuracy rate of the baseline model substantially. Since some areas contain much more data than others, the accuracy rate for data from these areas tended to be higher. The accuracy in areas with sparse data was lower, because fewer training data was used to classify the species that occur there. Two maps of accuracies in five-degree latitude and longitude rectangular areas are shown in figure **Figure 15**. The figure shows that the accuracy in many areas improved substantially between the baseline

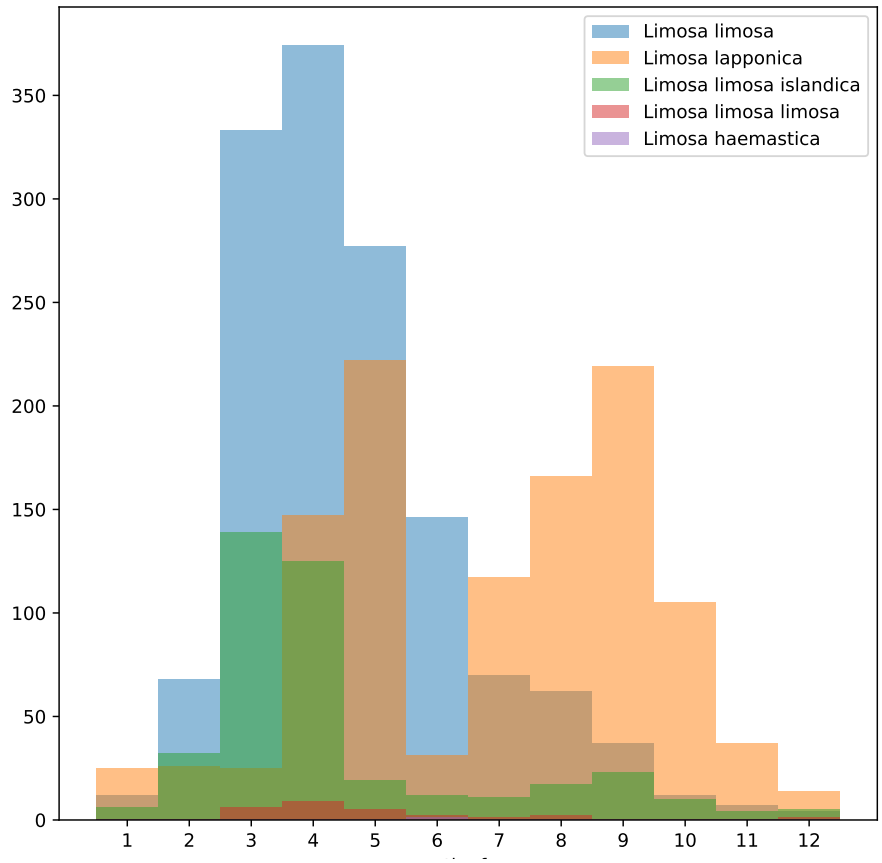


Figure 13: Distribution of observations in the genus *Limosa* over the months of the year (1 = January, 2 = February, etc.). The y-axis denotes number of observations in the test set. Different (sub)species are indicated with different colors.

model and model INLHD. Only one grid cell's local accuracy decreased, all others increased. Areas with few data points, like many areas in southern Europe, show the highest improvements between models.

Comparing global accuracy and average local accuracy tells us something about geographical bias. This is because average local accuracy is mostly determined by the accuracy of areas with sparse data, which are more numerous than areas with dense data. In contrast, average local accuracy is mostly determined by the accuracy of areas with dense data, as these provide many more observations. The fact that the average local accuracy improved more (3.77 percent point) than global accuracy (1.13 percent point), suggests that geographical bias in the model is reduced. Note the similarity between comparing rare and common classes and comparing dense and sparse areas. We can also calculate error reduction in local accuracies. The error in ALA for the baseline model was 19.5 percent point. The actual improvement was 3.77 percent point, which means 19.3% of the error was reduced. For global accuracy, error reduction was only 8.7%.

If geographical bias was reduced, one would expect to find a negative correlation between the density of data (number of observations in an area) and model improvement (difference in local accuracy between baseline and model INLHD). This correlation was present, although not significant (Pearson r -0.226, p-value 1.10e-01). However, this may have been because the distribution of data density is highly skewed, since the number of observations in different areas differs in orders of magnitude. To compensate for this, we evaluated the correlation between the log (base 10) of the area data density and model improvement. Indeed, the correlation between these variables was stronger (Pearson r -0.416, p-value 2.39e-03).

Finally, we observed a bias reduction by looking at the variance of the local accuracies (VLA). The VLA of the baseline model was 4.31e-03, while the VLA of model INLHD was only 2.00e-03. This substantial reduction also indicates that geographical bias was reduced, as local accuracies are more similar to each other when variance is smaller. We can see that the local accuracies became more equally distributed for model INLHD in **Figure 15**. The grid cells in the map for model INLHD show similar shades of color, while the grid cells in the map for the baseline model show different shades. The range of local accuracies decreased from 23.9 percent point (66.4% to 90.3%) to 20.5 percent point (70.0% to 90.5%).

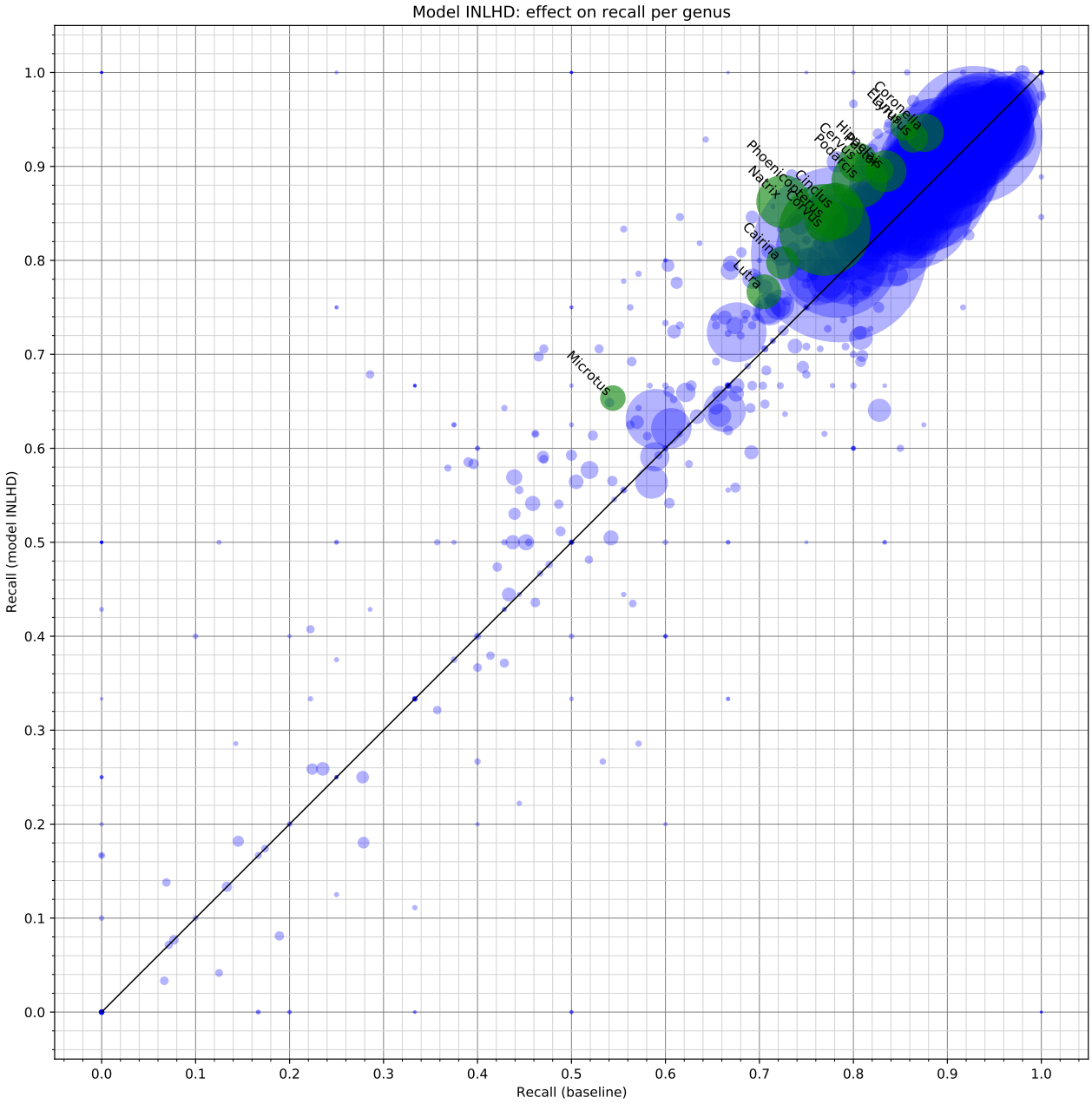
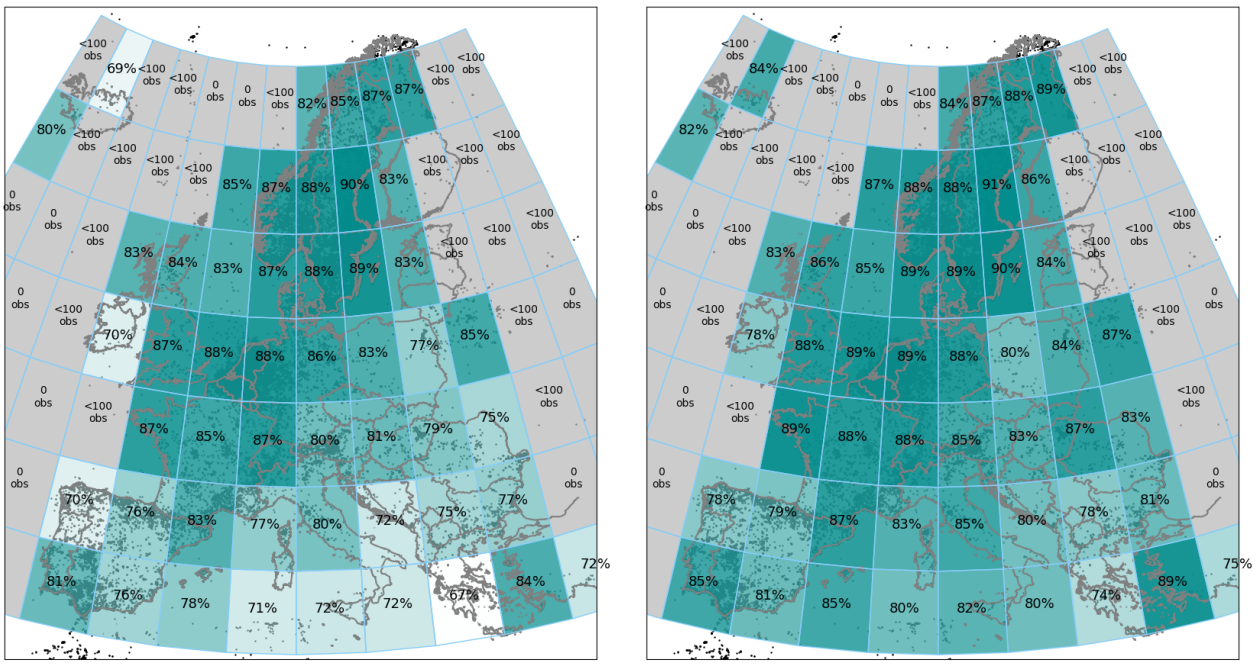


Figure 14: Relationship between genus (group of classes) recalls in model INLHD and the baseline model I. Each bubble represents one genus and its size indicates the number of observations in the test set for all species which fall under that genus. Genera with at least 300 observations and an average recall at least 6 percent point higher in model INLHD compared to the baseline are made green and are labeled. Other genera are made blue. There are no genera with at least 300 observations and an average recall at least 6 percent point lower in model INLHD compared to the baseline. The black diagonal line represents the points were average recall is equal in both models.

5.5 Area ALA improvements

Some areas improved their local accuracy more than others. The area that showed the biggest accuracy improvement is the grid cell containing northern Iceland, which improved its accuracy from 68.8% to 84.4%, an improvement of 15.6 percent point. The most common class in this area is *Bucephala islandica*, with 15 observations in the test set. The recall of this class improved from 6/15 (baseline) to 14/15 (model INLHD). The species this class corresponds to is visually quite similar to the species *Bucephala clangula*, which is not or only rarely present in Iceland. Model INLHD seems better equiped to distinguish these species, since it has access to information on location, habitat and neighboring species. This effect is similar to the effect described in **Section 5.2**. Separation of species' geographical ranges, like for species in the genus *Bucephala* is the most likely explanation for the great local improvement in accuracy in Iceland, especially since Iceland is spatially



(a) Baseline model

(b) Model INLHD

Figure 15: Maps of local accuracies in five-degree latitude by five-degree longitude grid cells. Maps are shown for two different models. Observations in the test set are shown as grey (inside grid) or black (outside grid) dots. Grid cells with less than 100 observations are made grey. Other grid cells are colored on the basis of the local accuracy, ranging from white (67%) to teal (91%). Local accuracies are annotated in each grid cell with more than 100 observations. A grid cell covering Eastern Türkiye falls just outside the map: its local accuracies for the baseline model and model INLHD were 66% and 70% respectively.

well separated from mainland Europe. This is also the reason Iceland again pops up in our analysis. However, we expect the effect of geographical pattern recognition by the model to extend within less separated areas on the continent of Europe, although possibly to a smaller degree.

5.6 Reducing abundance bias in different areas

Our metric for reducing abundance bias in different areas, average local average recall, improved by 2.35 percent point between the baseline and model INLHD. Since this metric is mainly determined by the recall of species that are locally rare, this increase in ALAR suggests that the model improved performance on rare species in areas with sparse data.

Further analysis of the ALAR metric is somewhat difficult. The local average recall in areas with sparse data is not necessarily expected to be lower. In these areas, few classes actually occur, which means relatively many common species will be present. This increases local average recall, since common species show high recall rates. However, areas with a lot of data also show high local average recall, as a lot of data is available for training. These contradictory patterns make a more thorough analysis complicated, so this is omitted.

5.7 Synergy between visual similarity and geographical uniqueness

In **Subsections 1.1.3, 5.2 and 5.5**, we pointed out example species that showed high recall improvements in model INLHD. We noted that the model might perform especially well when two species are visually similar, but have a distinct geographical range. In this section, we evaluated whether the combination of these two properties of a species (visual similarity to some other species and geographical uniqueness) indeed affects model performance.

Before we can distinguish the effects of visual similarity and geographical uniqueness on recall improvement, we need to consider confounding variables. One confounding variable is already known: class size (number of observations in a class). We have seen that this variable is linked to recall improvement (see **5.1.1**) and we know that species with many recorded observations tend to have larger ranges (see **1.3.3**). These larger ranges

can distort the geographical uniqueness variable, since more local ranges tend to be more 'unique' (separated from other ranges). So, instead of working with absolute recall improvement, we work with error reduction R_{err}, which is defined as the recall improvement divided by the error rate of the baseline model. In this way, a large part of the confounding variable 'class size' is neutralized. This definition creates a small number of classes were the error reduction is non-defined, namely if the baseline error is zero. These classes are taken out of the analysis.

To evaluate the effects of visual similarity and geographical uniqueness, we need numerical proxies for these variables. We define visual similarity S_{vis} between a class and its most similar class as:

$$S_{vis} = \max_{j \neq i} R_{i,j} \tag{24}$$

where cluster similarity between classes i and j, $R_{i,j}$, is defined as:

$$R_{i,j} = \frac{D_i + D_j}{M_{i,j}} \tag{25}$$

In this last equation, D_x denotes the average intra-cluster visual distance of all observations in the test set of class x. The intra-cluster visual distance is defined as the Euclidian distance between an input image representation and the class' image representation cluster centroid. Cluster centroids are defined as the average of all image representations in a class. These distances exist in image representation space, consisting of the 1280 dimensions of the image representation vector. $M_{i,j}$ denotes the inter-cluster visual distance, which is the Euclidian distance between two cluster centroids. This definition of similarity is derived from Davies and Bouldin (1979). The class values of S_{vis} range from 1.96 (visually unique, distinctive) to 13.16 (visually very similar to another class).

We define geographical uniqueness as:

$$U_{geo} = \frac{1}{S_{geo}} = \frac{1}{\max_{j \neq i} R'_{i,j}} \tag{26}$$

Again, cluster similarity $R'_{i,j}$ is defined as:

$$R'_{i,j} = \frac{D'_i + D'_j}{M'_{i,j}} \tag{27}$$

So, the geographical uniqueness is the reciprocal of geographic similarity S_{geo}. Cluster similarity $R'_{i,j}$, intra-cluster distances D'_x, and inter-cluster distances $M'_{i,j}$ have the same definitions as for visual similarity, but exist in geographical (two-dimensional) space. The normalized latitude and longitude values are used to calculate these variables. The class values of U_{geo} range from 0.000115 (not unique) to 0.941 (very unique, local).

After we calculated the error reduction R_{err}, visual similarity S_{vis} and geographical uniqueness U_{geo} for all classes, two multiple regression analyses were performed using ordinary least squares. First, we performed a regression analysis without interaction, modeling $R_{err} \sim S_{vis} + U_{geo}$. The results can be found in **Table 8**. Then, we performed an analysis with interaction, modeling $R_{err} \sim S_{vis} + U_{geo} + S_{vis} * U_{geo}$. These results can be found in **Table 9**.

In both analyses, the intercept was positive, with high significance. This indicates general improvement of the model independent of visual similarity and geographical uniqueness. This is not unexpected, since information on date, habitat and neighboring species could have improved model INLHD. In the first analysis, we also see a strong and significant effect of geographical uniqueness on error reduction. This is consistent with the view that locally present species benefit from the new model architecture. In the second analysis, the positive effect of geographical uniqueness disappeared. Instead, we see a strong and significant effect of the interaction between visual similarity and geographical uniqueness. This synergistic effect is consistent with our hypothesis that species that that are both visually hard to distinguish and geographically isolated benefit especially well from the new model architecture. Finally, we see that the singular effect of visual similarity has a small but significant negative effect on error reduction. This may be because the image representation is considered a 'less important' feature in model INLHD. If the model focused on information from neighbors too much, it could have ignored small variations in the image representation, leading to worse results on species that are visually hard to distinguish.

Effect	Coefficient	Standard error	t	p-value
Intercept	0.122	0.021	5.73	<0.001
S_{vis}	-0.009	0.006	-1.53	0.125
U_{geo}	0.256	0.088	2.90	0.004

Table 8: Multiple regression analysis on the effects of visual similarity and geographical uniqueness on error reduction, without interaction.

Effect	Coefficient	Standard error	t	p-value
Intercept	0.139	0.022	6.26	<0.001
S_{vis}	-0.017	0.006	-2.61	0.009
U_{geo}	-0.672	0.363	-1.85	0.064
$S_{vis} * U_{geo}$	0.411	0.156	2.64	0.008

Table 9: Multiple regression analysis on the effects of visual similarity and geographical uniqueness on error reduction, with interaction.

6 Discussion

In this section, we discuss the results of our full model and its ablations. We start by interpreting our findings, after which we reflect critically on our methods. We then consider possible directions for future research into model improvements and we finish with notes on implementation and implications of this work.

6.1 Interpretation of the results

6.1.1 Main findings

The full model clearly improved upon the baseline and performs better than all ablations. When results from models for different taxonomic groups were combined, the full model had the highest scores on all four main performance metrics (see **Table 4**). The improvement of the new model over the image-only baseline was 1.48 percent point, which corresponds to an error reduction of 9.53 percent. Average recall improved by 3.13 percent point, corresponding to an error reduction of 5.89 percent. In general, classes with few observations, representing rare species, profited more from the new model than classes with many observations. Average local accuracy (ALA) improved by 2.92 percent point, an error reduction of 13.58 percent. Local accuracies became more equally distributed. Average local average recall (ALAR) improved by 2.61 percent point, an error reduction of 13.90 percent. Also, the full model had the best (lowest) score on the metric for geographical bias VLA. The full model did not show the highest performance on average precision. Performances on this metric are be further evaluated in **6.1.9**.

The addition of location, habitat and date information systematically improved all model metrics. However, this improvement was smaller when neighboring species information was already present in the model. Training time and inference time increased for the new model architectures, but this increase was limited when enough parallelization was applied.

6.1.2 Individual taxonomic subgroups

Table 6 shows the number of significant effects we found for each metric for each measured effect. Counts of positive and negative effects are separated. The addition of location, habitat variables and date as input to the model only induced positive significant effects across different taxonomic groups. The addition of neighborhood information induced different types of effects regarding accuracy, but almost always improved average recall: this is further evaluated in **6.1.3**. The use of the transformer architecture induced both positive and negative significant effects for different taxonomic groups: this is further evaluated in **6.1.4**.

When looking at effect measurements for different taxonomic groups separately in **Appendix B**, a pattern can be observed. The taxonomic groups can be divided into two rough groups. The first group constitutes the subgroups Diptera, Fungi, Insecta, Lepidoptera and Plantae. For this group, all measured model changes induced effects that were largely positive and only sometimes slightly negative. The only clear exception to this pattern is the Plantae subgroup, which shows a clear negative effect for adding neighboring species information. For these subgroups, the full model performed best on all metrics. The second group consists of subgroups Arthropoda, Chordata and Mollusca. For this group, the addition of neighboring observation information induced a strong positive effect. However, the addition of the transformer architecture induced a negative effect. For this group, either model IN or model INLHD performed best.

6.1.3 Effect of neighboring species information

The addition of neighboring species information as input to the model showed clear benefits in almost all cases. Not only did this reduce model biases, it generally increased model performance. The geographical distribution of species, spatially, temporally and possibly spatio-temporally, seems to be a clear reason behind model improvement. Species that are only present in certain locations, or at certain times, can be better predicted when the model has access to this information. The use of contextual information for improved species identification is done by human experts in a similar way (Terry et al., 2020). The distribution of neighboring species contains relevant information for species identification. We hypothesise this is at least partly because the local distribution of species can serve as a unique proxy for a certain habitat or ecological niche (set of environmental conditions optimal to one species). Furthermore, the model is able to learn patterns between co-occurrences of species. Although earlier works showed that local species abundance can aid identification (Berg et al., 2014; Chu et al., 2019; Wittich et al., 2018), the capacity for a primarily image-based recognition model to learn species co-occurrence patterns is novel.

6.1.4 Effect of transformer architecture

The use of a transformer architecture showed mixed results. For some taxonomic groups, there was a clear improvement compared to models where neighboring species information was concatenated. For other groups, it reduced model performance. It is not completely clear why the transformer architecture showed different performance for different taxonomic groups. Although both positive and negative effects were deemed significant, all measured effects came from one trained version of the model. Possibly, models with the same architecture and hyperparameters that are trained again with differently initialized parameters could show different performance. In that case, the effect of using the transformer architecture may turn out to be independent of taxonomic groups. Another explanation is that the taxonomic groups and the effect of the transformer architecture are truly linked. In that case, it is not obvious which taxonomic groups profit from the architecture and why. Possibly, only taxonomic groups with more observations may profit from the more flexible transformer architecture. An argument for this explanation is that transformers are known for their low inductive bias, making them only effective if enough data is available (Dosovitskiy et al., 2020; Mialon, 2023). This would explain why the transformer architecture has a negative effect on taxonomic groups with few observations, namely Arthropoda and Mollusca. It would also explain why the transformer architecture has a positive effect on some groups with many observations, such as Insecta, Lepidoptera and especially Plantae. However, it does not explain why the transformer shows a negative effect on the taxonomic group Chordata. The relationship between model performance and available data is further explored in **6.1.8**.

6.1.5 Interaction metadata variables and neighboring species information

For this research, we were also interested in the question if modeling ecological context using neighbor information and directly integrating non-visual information (location, date, habitat variables) can complement each other. If we look at the comparisons of models in **Appendix B**, we get a nuanced view.

On the one hand, we can see that adding location, date and habitat variables often complements the addition of neighboring species information. When one of these additions was already present, the other often still induced a positive effect on performance.

On the other hand, we can also see that both methods of providing contextual information are not independent. When one was already implemented, the effect of adding the second method was often reduced and sometimes even completely vanished. This was expected, as we expect an overlap in information between neighborhood species presence and the location and date. Some species are simply more present in some locations. If the model already knows about locally present species, information about the location or habitat is less informative. Vice versa, if the model already has access to information about location, date and habitat, information on neighboring species is less informative.

6.1.6 Overfitting

The mentioned improvement through the addition of neighboring species information is not too surprising, as the model is able to learn by itself which information is valuable and which information is not. However, the performance increase is not trivial. The addition of neighboring species information comes with a substantial increase in the number of model parameters (see **Table 2**). This increases the chances of model overfitting, which can lead to worse performance during testing and on new data (Hawkins, 2004). We observed this type of overfitting when experimenting with model architectures using small subsets of the training data. When we finally trained the model with the complete dataset, overfitting may have been reduced because the model had much more data to train on. Using more data during training reduces overfitting (Ying, 2019).

To further evaluate possible overfitting, we visualized how the validation accuracy of models with different architectures developed during training. This visualization can be found in **Figure 16**. The figure shows that models IN and INLHD started performing worse after only a few epochs of training. For other taxonomic subgroups, the number of trained epochs until peak validation performance was reached for these architectures was also lower than for the other architectures (see **Table 5**). This pattern is most likely due to overfitting. Although the overfitting will surely be facilitated by the large number of parameters in these architectures, this is not the only reason these models overfit. After all, the model T(IN) and the full model also have large numbers of parameters, but seem to show much less overfitting. We think this has three reasons. First, the transformer architecture introduces some dropout layers, which are known to help reduce overfitting (Srivastava et al., 2014). Second, the transformer architecture reduces the number of parameters in the final layer in comparison with models IN and INLHD, since this last dense layer does not connect all community representations with the output layer. Using less parameters also reduces overfitting (Ying, 2019). Third, the topology of the transformer architecture may facilitate reducing overfitting. This is explained in the next subsection.

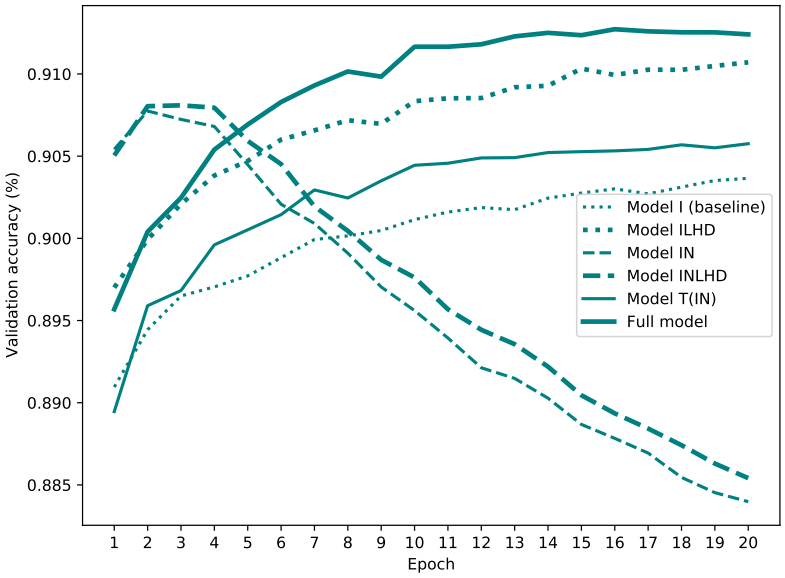


Figure 16: Validation accuracy during training of different model architectures for the Chordata subset.

6.1.7 Preventing overfitting using the transformer architecture

In models IN and INLHD, many connections exist between the individual community representations and the output layer. Many of these connections probably do not directly contribute to the accuracy of the network, as most relevant information the model uses comes from the image representation. These unused connections are susceptible to overfitting, as the change of their weights can facilitate learning new patterns without damaging performance. In contrast, in the models with a transformer architecture, no direct connections between community representations and the output layer exist. This essentially forces all information from neighboring species through a 'bottleneck layer', which is the transformed image representation. Potential overfitting needs to be facilitated through the transformed image representation. However, the vector size of this representation is limited. Changing any of the weights connected to this representation may harm performance, as the image representation already contains a lot of relevant information. This status makes the image representation more robust, and changes to its weights will only occur if there is strong reason to (large backpropagated weight updates). In this way, the transformer architecture may create a 'barrier' for overfitting. To our best knowledge, the capacity of a transformer bottleneck architecture to reduce overfitting has not been mentioned in other work. However, the broader concept of information compression via dimensionality reduction is a key element of deep learning and is known to reduce overfitting (Schittenkopf et al., 1997). In this regard, preventing overfitting by using an attention-based model architecture may not be unexpected. After all, attention is about concentrating on the most relevant information, and ignoring other information.

It needs to be noted that transformer models in general increase model complexity, increasing the number of model parameters and increasing the risk of overfitting. It is specifically in this case, where the transformer models are compared to models with even more parameters (IN and INLHD), and where many features in the

input may contain no relevant information, that the transformer bottleneck topology may also reduce overfitting.

6.1.8 Model performance in relation to amount of data

To evaluate the relationship between model performance and amount of available data further, we train additional models. For this study, we use the Plantae subgroup, as it has the most data. Apart from the models we trained using 100 percent of our data, we train seven additional versions of models. For each, we use 50 percent of the data used in the previous version. So, the models are trained on respectively 100%, 50%, 25%, 12.5%, 6.3%, 3.1%, 1.6% and 0.8% of our training data. Models are trained for three architectures: model ILHD, model INLHD and the full model. The performance of each model can be found in **Figure 17**.

With less data to train on, the model performance dropped. The performance of model INLHD dropped much more rapidly than that of the other models. This supports the presumption that model INLHD was at risk of overfitting, since overfitting is more likely when less data is available (see **6.1.6**).

However, we also observe that the performance of the full model decreased the least when trained with less data. This does not support the statement that the transformer architecture is only beneficial when enough data is used for training, which was described in **6.1.4**. On the contrary, it suggests that the transformer architecture is actually beneficial when fewer data is available. This is unlikely, since transformer models are known to work better with more data (Dosovitskiy et al., 2020; Mialon, 2023). Rather, we expect that the high performance of the full model was due to the additional input information. The transformer made use of species information from neighboring observations. The mapping from this information to a class may have been more straightforward to learn than the mapping from an image to a class. In that case, the neighboring species information was more valuable if fewer images were available to train on. When enough images were available, the model will have been better able to distinguish different species based on the image, and the neighboring species information became less important. Model INLHD also had access to neighboring species information when less images were available for training. However, the added value of this information may have been negligible if overfitting diminished performance.

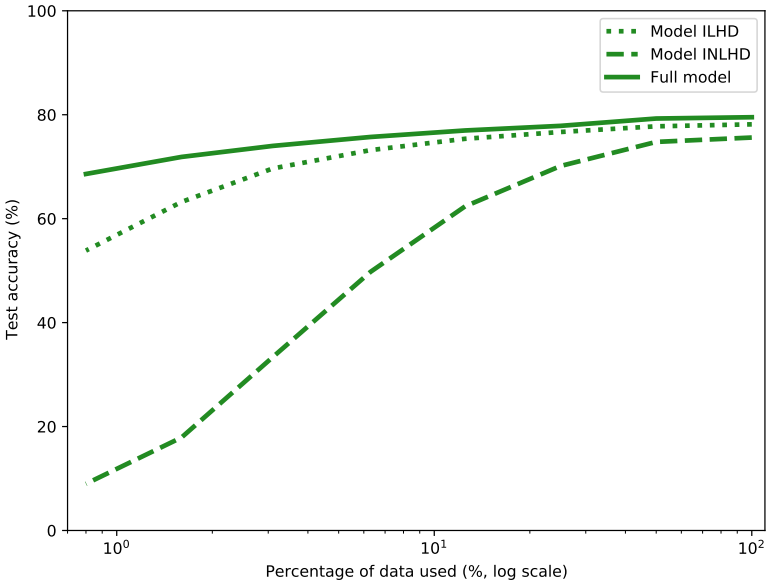


Figure 17: Test accuracy of different model architectures on differently sized parts of the training data for the Plantae subset.

6.1.9 Average Precision

In contrast to other evaluation metrics, the full model did not have the highest macro-average precision (the average of the precisions of all classes). Instead, model INLHD scored highest on this metric, followed by model IN (see **Table 4**). This high average precision for model INLHD originated in relatively high precision scores for many minority classes representing rare species, which can be seen in **Figure 18**. Although the full model showed higher recall on minority classes, model INLHD outperformed the full model regarding precision. This suggests that model INLHD may have been more conservative, only predicting minority classes (rare species)

when it was very confident. This would explain both the high precision and low recall rates for model INLHD. Counterintuitively, this is not the case. Actually, model INLHD more often predicted minority classes than the baseline and the full model, which can be seen in **Figure 19**. Thus, it is not directly clear why model INLHD showed higher precision and lower recall rates.

Possibly, model INLHD asserted an (overly) high importance to some features from the neighbor information inputs. In some cases, these 'strong indicators' may have enabled INLHD to make correct predictions for minority classes, increasing model precision rates. An example of a strong indicator from neighbor information may be a host plant of a certain butterfly, a lichen that only occurs on certain trees, or the species itself. However, these strong indicators only lead to successful predictions in some cases. If they override the high-quality input from the image representation in other cases, they can lead to lower recall rates and lower overall model performance. This view of strong indicators from neighboring species information that are rarely correct is in line with our earlier hypothesis on overfitting by model INLHD and the robustness of the full model based on the bottleneck architecture. The idea that using fewer features can increase model performance already exists and is the basis of feature selection techniques. These techniques can increase recall rates and reduce overfitting, even though precision rates may in some cases decrease (Kamalov et al., 2023).

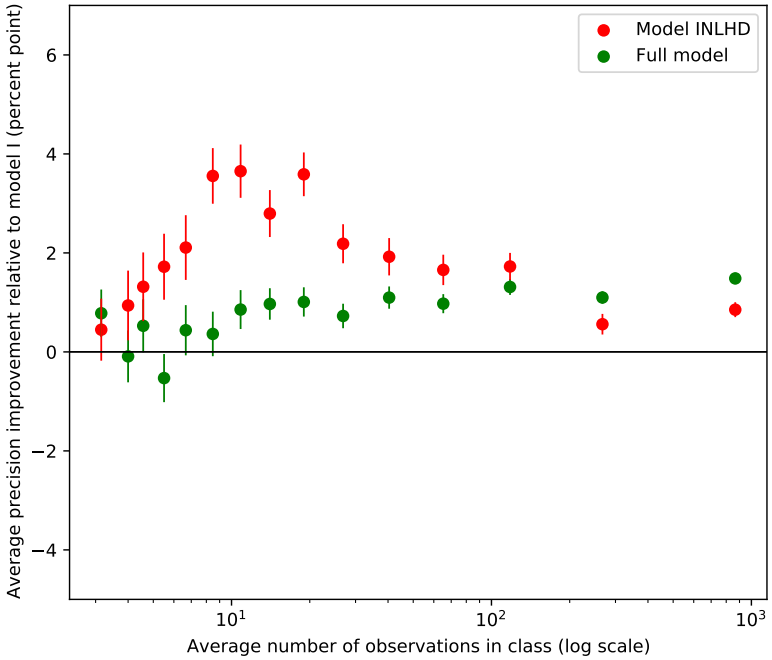


Figure 18: Relationship between precision improvements and class size (number of observations in the test set of a class), for two models. The improvement is defined as the increase in class precision from using the baseline model to using model INLHD (red dots) or the full model (green dots). Only classes were used that were predicted at least once by the baseline model, model INLHD and the full model. Classes were divided into 15 same-sized bins that contained 1531 classes each. Bins were filled with classes in order of number of observations. Each dot represents a bin, showing the average number of observations in the test set and average precision improvement of classes in each bin. Bars are shown representing the standard error of the mean for each bin.

6.2 Critical reflection

6.2.1 Limited number of training epochs

In **Figure 16**, we can see some models may not have been completely optimized. Training the model for more epochs may have further increased model performance. We can infer that this is the case for many taxonomic subgroups from **Table 5**. The peak performance of many models was only reached after twenty epochs, suggesting potential for further improvement. The goal of this work is not model optimization but rather comparing different model architectures. In that light, models may still have been fairly compared as long as they were trained for the same number of epochs. However, we also see that the performance difference

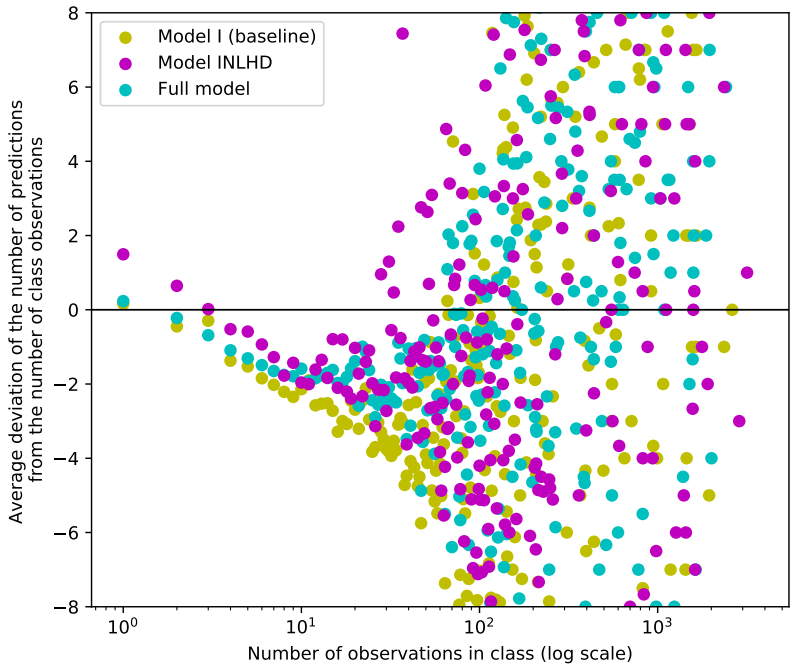


Figure 19: Relationship between the number of predictions of a class (in the test set) and the number of observations in a class (in the test set), for three different models. Classes with the same number of observations are combined into groups. For each group, the average number of predictions for these classes is calculated and represented as a dot. The number of predictions is displayed relatively to class size. If a group of classes was predicted as often as it actually occurred, a dot will end up on the horizontal line denoting a deviation of zero. Groups of classes that were predicted more often than they actually occurred, have positive deviation numbers and groups of classes that were predicted less often than they occurred have negative deviation numbers. Different colors represent different models. It can be seen that classes with very few observations in the test set (1 or 2) were predicted more often on average. Classes with few observations (2 to 30) were predicted less often than they actually occur. Model INLHD was more likely to predict classes with few occurrences than other models.

between different models varies over the number of epochs trained. Models IN and INLHD initially showed maximum performance, but were surpassed by model ILHD and the full model after six to ten epochs. More importantly, the performance difference between the transformer models (T(IN) and the full model) and models I and ILHD seems to have decreased in epochs ten to twenty. If this trend would have persisted when training was continued for more epochs, the advantages of the transformer architecture may have turned out to be smaller than reported in earlier sections of this work.

6.2.2 Impure validation data

In this work, we kept training data, validation data and test data strictly separate. Only training data was used for training, only validation data was used for selecting the best model after training, and only test data was used for calculating the performance metrics for each model. However, we did use image representations that were computed by a model that used the same data during development. The convolutional part of this model was trained using the data we used for training and validation, and was validated using the data we used for testing. No independent testing was performed, as the original model was employed in a practical context and no scientific testing metrics were needed. This means the embedding space in which the image representations exist was 'trained' using the data we used for validation and 'optimal space' was selected using the data we used for training. In theory, this created some unfortunate indirect leakage between datasets. Although we never directly trained our models on data from the validation or test sets, the image representations were computed by a model that has 'seen' the validation data (as training data) and the test data (as validation data) before.

Since this leakage was quite indirect and the data we used as test data was only used for validation of the convolutionary part of the model, we expect our test data to have been 'pure enough' to be used for testing our models. However, our validation data was somewhat 'impure', since image representations were computed by a

model trained on this data. Since our validation data was not used for testing, this impurity did not influence the performance metrics reported in this work. However, it may have influenced the model version we selected for testing, for the full model and all model ablations.

6.2.3 Separation of taxonomic groups

The separation of the data and the model architecture into the eight taxonomic groups was somewhat arbitrary. The choice for the eight groups for this work was made based on the same separation that was made during development of the current image-only model for separating the eight convolutionary submodels. Since we work with image representations computed by these submodels, it was a logical choice to retain the same taxonomic groups. Note that the separation into eight groups is a convenience too, as it is used both in separating convolutional submodels and in separating neighbors into taxonomic groups.

Working with multiple taxonomic groups has a number of advantages. First of all, the classification model has a lower number of classes. This decreases the number of weights in the last layer, reducing model complexity. A reduced number of classes also means that the embedding space of the image representation is optimized for a smaller number of classes. If an image representation of the same size should contain information on more possible species, fewer dimensions can be used to represent each species. If we assume that the main convolutionary model has selected the correct submodel, the model also has a lower error rate, since the model can optimize its weights for data of fewer classes, giving it more room to specialize on one taxonomic group. Furthermore, it has fewer classes it can predict and so a lower probability to make a mistake. However, this last benefit is limited, since observations will most likely not be confused with observations from a completely different subgroup. An observation of a particular butterfly may be confused with another butterfly, but will probably not be confused with any mammal. However, employing multiple submodels also has a disadvantage. The main convolutional model could select the incorrect taxonomic submodel, in which case the submodel will be forced to make an incorrect prediction.

The current separation into eight submodels is a compromise: more or less submodels could be used. Which separation is best depends on available computational resources, the size of the dataset, the total number of species to classify and the capacity of the deep neural network.

6.2.4 Alternative optimization metrics

In our work, we optimized models on validation accuracy. Model versions were evaluated after every epoch and the version with the highest validation accuracy was selected as final model. However, model accuracy was not the only relevant metric. For example, we could have optimized our model in regards to average recall or average local accuracy. The advantage of selecting on accuracy is that this reduces the chance that overall model accuracy improvement is low or negative. If implementation of a new model is considered by users employing the model, an increase in overall model accuracy may be a prerequisite for implementation. However, our focus was on reducing model bias. Since metrics like average recall and average local accuracy are more related to reducing bias, choosing these metrics as optimization goals may have resulted in larger improvements on these metrics.

6.2.5 Duplicate observations

There is a possibility that pseudo-duplicate observations are present in our data. Some individual animals may attract a lot of observation from observers. For example, people watching birds are known to visit vagrant birds, which have flown outside of their usual range and make for a locally rare occurrence. If one such individual bird was recorded multiple times by enthusiastic bird watchers, this may have distorted the data. After all, if one of these observations was used in the training data and one in the test data, the test data contained observations of individuals on which the model had trained. This is especially worrying for models that used neighboring species as input, but models that used a combination of date and location would also have been at risk. This is because these multiple bird observation records typically occur in one or a few locations over the range of a few days. The model could have overfitted on these observations as neighboring species, or on their location-date combination proxy. Multiple observations of the same vagrant bird in the same location at the same time cannot be considered true new data, as they are not expected to occur again. However, it should be noted that these types of multiple observations are mostly limited to birds, as individuals of other taxonomic groups rarely attract multiple enthusiastic observers at the same time. Furthermore, the respective birds are usually wandering birds, meaning they could occasionally turn up in a large range of areas. This further limits potential problematic relationships the model may have learned between a location and a wandering bird. The possible problematic effect of multiple identical observations could be limited in future work by dividing the data based on year. For example, records from 2010-2020 could be used as training data and records from 2021 and 2022 could be used as validation and test data respectively. However, this approach could introduce data biases due to the use of cameras with other characteristics (e.g. regarding noise suppression or color saturation).

An alternative way to limit the effect of this issue would be to omit a species from its own neighborhood data input. However, this may negatively impact performance because the historic presence of a species is a strong indicator for its current presence.

6.3 Future work

6.3.1 End-to-end training

End-to-end training of the complete model, including the convolutionary part, may further improve the model. In our work, we retrained a part of the model, using ready-to-use image representations as input. The first part of the model, consisting of the convolution operations that transform an image to an image representation, was unchanged. Instead, the weights of the convolutionary part of the model were frozen during training of our models. The advantage of this choice is that it saves a lot of computational resources, as training the convolutionary part of the image model takes more time and more computational power. However, this choice also has a disadvantage. Usually, models with a neural network architecture perform best when they are trained end-to-end, meaning the entire network is trained together (Duan & Principe, 2021). The image representations are computed by a convolutionary model that was optimized to compute embeddings that could be classified using a single dense layer (the classification heads in the current network). This means that the representations may not be perfectly suited for our models, which have additional inputs and a different architecture. Training the entire model end-to-end may further improve the model, as the generation of image representation can then be optimized for the particular architecture that is used in later parts of the model.

6.3.2 Shared weights

Including shared weights across taxonomic groups for some layers of the model may further improve its performance. In our work, we trained several different models for different taxonomic subgroups. Although this was practical, it may not have been an optimal design choice. Many weights in our different submodels may have been shared between taxonomic groups. For example, the dense layers connecting the neighboring species abundance vectors to the community representations have a similar structure across taxonomic groups, and the taxonomy embeddings may also be similar across the different models we built. Possibly, the weights in these layers may be shared across taxonomic subgroups. In that way, the weights have a lot more data to train on, since they can be trained using all data instead of just a subset of the data. However, it is also possible that these weights have a different function in different taxonomic submodels. Building networks with shared weights for some layers and taxonomy-specific weights for other layers may be a way to optimize model performance.

6.3.3 Multiple query images

Including multiple images from a single observation as input to the network may further improve its performance. In our work, we used one image for each query observation. In reality, multiple images are regularly taken for a single observation. For example, a user may take an image of an entire plant, a second picture of a leaf and a third picture of a flower from the same plant. One data instance (observation) then consist of multiple images and one set of metadata variables. The architecture of the model should be adjusted to combine multiple images into a single image feature for this purpose. This could further improve model performance, since additional pictures contain additional information that the model can exploit to predict the correct class. In future work, the added value of additional images may be compared with the added value of additional information used in our work (location, habitat variables, date, and neighboring species).

6.3.4 Temporal distance as part of the neighbor definition

Including a temporal component in the definition of distances to neighbors could enable the model to learn new patterns, but at the cost of inference speed. In our work, we determined neighboring species as species that were spatially close to the query observation. However, one could use a different distance definition that includes a temporal component. If this is done, species that occur in the same part of the year will be included into the neighboring species set while species present in other parts of the year may be excluded. If the time of day is taken into account, this may also be used to distinguish diurnal and nocturnal species. Including a temporal component into the neighbor definition has been done by Berg et al. (2014) and Wittich et al. (2018). Note that including such a temporal component would increase the dimensionality of the neighborhood space. In our context, this would substantially decrease the computation speed for calculating which observations are neighbors to the query observation, which increases inference time. Another disadvantage is that including a temporal component would mean excluding a number of neighbors that currently fall within the neighborhood. This implies a larger spatial neighborhood size would be needed to collect data from the same number of

neighboring observations. This larger spatial range may decrease the quality of the added value the neighboring species provide.

6.4 Relevance and implementation

This work adds to the growing body of different applications for transformer networks (Dosovitskiy et al., 2020; Vaswani et al., 2017; Zhang et al., 2023). The attention-based architecture has shown to be useful for a broad range of tasks, where the context around a query object is taken into account to enrich the object representation (Azad et al., 2022; Cordonnier et al., 2020; Ferrando et al., 2022; Li et al., 2020). In our work, we showed that the local species community provides relevant contextual information to identify a species in an image. The transformer architecture allows us to implicitly model species' distribution ranges and relevant co-occurrence patterns. Previous methods had to model distribution ranges explicitly and connect them to an image recognition model using hard-coded formulas (Berg et al., 2014; Wittich et al., 2018).

By avoiding early fusion of the image feature with other information, we left the convolutional part of the model intact, similarly to models made by Chu et al. (2019), Terry et al. (2020), and Yang et al. (2022). This enables the model to be added as a module to an existing CNN. It can also be used next to a baseline model to create two predictions from one image representation: one based on only the visual feature and one geo-aware prediction. The model generating geo-aware predictions could be updated by retraining it, without the need to retrain the convolutional part of the model. Although our model was trained on data from Europe, the model architecture can be used to classify data from any geographical range. This is facilitated by the definition of the neighborhood size we used, which is based on data density. It would be recommended to use different types of taxonomic groups, as these provide different ecological contextual information. Possibly, the adjustment of embeddings using geographical data with a transformer could be extended to applications other than species identification, such as land-use classification from satellite imagery. It may also be applied to non-visual tasks, such as bio-acoustic species identification.

7 Conclusion

In regards to our research questions, we can draw the following conclusions:

- Modeling ecological context using neighboring species information generally increases model performance. The use of a transformer architecture is beneficial for most taxonomic groups. Model performance can be further improved by adding information on the location, date and local habitat of the observation. In the full model, accuracy across all taxonomic groups increased with 1.5 percent point, which is an error reduction of 9.5 percent.
- Modeling ecological context reduces abundance bias. In absolute terms, the performance on rare species (classes with little data) substantially increases. It should be noted that these classes have a lower baseline performance, so a more substantial improvement can be expected. In the full model, average recall across all taxonomic groups increased with 3.1 percent point. Abundance bias was also reduced specifically in areas with little data, since the average local average recall (ALAR) increased.
- Modeling ecological context reduces geographical bias. Performance in areas with little data substantially improved and, as a result, performance became more equal across areas. The error in average local accuracy (ALA) was reduced by 13.58 percent.
- Modeling ecological context can partly complement the integration of location, habitat and date information. However, the two methods overlap in the information they contribute and their positive effects on performance are not completely additive.
- Modeling ecological context increases training time. However, this increase is modest in comparison to training the convolutional part of a model, and can be further reduced using multiple computing cores. Inference time is also increased slightly, but not in a way that jeopardizes quick inference in the field by users of applications employing the model.

8 Acknowledgements

Parts of this work were funded by the EU Horizon project MAMBO under grant agreement No.101060639.

Research and development of image recognition models for species identification would not be possible without the invaluable work of volunteers. We would like to thank all citizens that uploaded their observations and all validators that safeguarded data quality by confirming species in observations.

I would like to thank my supervisors Laurens Hogeweg, Django Brunink, Ronald Poppe, Vincent Kalkman and Albert Salah for their contributions and guidance throughout the project.

Code for the models and analyses described in this work will become available at https://gitlab.com/wouter.ubbink/msm-geo.

References

Anavyanto, A. F., Maimunah, M., Yudianto, M. R. A., & Sukmasetya, P. (2023). Efficientnetv2m for image classification of tomato leaf deseases. *PIKSEL: Penelitian Ilmu Komputer Sistem Embedded and Logic*, *11*(1), 55–76.

Aodha, O. M., Cole, E., & Perona, P. (2019). Presence-Only Geographical Priors for Fine-Grained Image Classification. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 9595–9605. https://doi.org/10.1109/ICCV.2019.00969

Auster, P. J., Joy, K., & Valentine, P. C. (2001). Fish species and community distributions as proxies for seafloor habitat distributions: The stellwagen bank national marine sanctuary example (northwest atlantic, gulf of maine). *Environmental Biology of Fishes*, *60*, 331–346.

Azad, R., Heidari, M., Wu, Y., & Merhof, D. (2022). Contextual attention network: Transformer meets u-net. *International Workshop on Machine Learning in Medical Imaging*, 377–386.

Berg, T., Liu, J., Lee, S. W., Alexander, M. L., Jacobs, D. W., & Belhumeur, P. N. (2014). Birdsnap: Large-Scale Fine-Grained Visual Categorization of Birds. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2019–2026. https://doi.org/10.1109/CVPR.2014.259

Chu, G., Potetz, B., Wang, W., Howard, A., Song, Y., Brucher, F., Leung, T., & Adam, H. (2019, September). Geo-Aware Networks for Fine-Grained Recognition. https://doi.org/10.48550/arXiv.1906.01737

Cordonnier, J.-B., Loukas, A., & Jaggi, M. (2020). Multi-head attention: Collaborate instead of concatenate. *arXiv preprint arXiv:2006.16362*.

Cornell Lab of Ornithology. (2023, December 27). *Merlin bird id*. https://merlin.allaboutbirds.org/

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2), 224–227.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Duan, S., & Principe, J. C. (2021). Training deep architectures without end-to-end backpropagation: A brief survey. *arXiv preprint arXiv:2101.03419*.

Feldman, M. J., Imbeau, L., Marchand, P., Mazerolle, M. J., Darveau, M., & Fenton, N. J. (2021). Trends and gaps in the use of citizen science derived data as input for species distribution models: A quantitative review. *PloS one*, *16*(3), e0234587.

Ferrando, J., Gállego, G. I., & Costa-Jussà, M. R. (2022). Measuring the mixing of contextual information in the transformer. *arXiv preprint arXiv:2203.04212*.

Fischer, A. G. (1960). Latitudinal variations in organic diversity. *Evolution*, *14*(1), 64–81.

Hawkins, D. M. (2004). The problem of overfitting. *Journal of chemical information and computer sciences*, *44*(1), 1–12.

iNaturalist. (2024, March 20). *Inaturalist*. https://www.inaturalist.org/

Johnston, A., Moran, N., Musgrove, A., Fink, D., & Baillie, S. R. (2020). Estimating species distributions from spatially biased citizen science data. *Ecological Modelling*, *422*, 108927.

Kamalov, F., Thabtah, F., & Leung, H. H. (2023). Feature selection in imbalanced data. *Annals of Data Science*, *10*(6), 1527–1541.

Krawczyk, B. (2016). Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence*, *5*(4), 221–232. https://doi.org/10.1007/s13748-016-0094-0

Li, J., Ji, D., Li, F., Zhang, M., & Liu, Y. (2020). Hitrans: A transformer-based context-and speaker-sensitive model for emotion detection in conversations. *Proceedings of the 28th international conference on computational linguistics*, 4190–4200.

Mai, G., Janowicz, K., Hu, Y., Gao, S., Yan, B., Zhu, R., Cai, L., & Lao, N. (2022). A review of location encoding for GeoAI: Methods and applications. *International Journal of Geographical Information Science*, *36*(4), 639–673. https://doi.org/10.1080/13658816.2021.2004602

Mai, G., Janowicz, K., Yan, B., Zhu, R., Cai, L., & Lao, N. (2020, February). Multi-Scale Representation Learning for Spatial Feature Distributions using Grid Cells. https://doi.org/10.48550/arXiv.2003.00824

Matutini, F., Baudry, J., Pain, G., Sineau, M., & Pithon, J. (2021). How citizen science could improve species distribution models and their independent assessment. *Ecology and Evolution*, *11*(7), 3028–3039. https://doi.org/10.1002/ece3.7210

Mialon, G. (2023). On inductive biases for machine learning in data constrained settings. *arXiv preprint arXiv:2302.10692*.

Observation International. (2024, March 12). *Obsidentify*. https://waarneming.nl/apps/obsidentify/

PlantNet. (2024, March 18). *Pl@ntnet plant identification*. https://plantnet.org/

Pollock, L. J., Tingley, R., Morris, W. K., Golding, N., O'Hara, R. B., Parris, K. M., Vesk, P. A., & McCarthy, M. A. (2014). Understanding co-occurrence by modelling species simultaneously with a joint species distribution model (jsdm). *Methods in Ecology and Evolution*, *5*(5), 397–406.

Rodríguez de Rivera, Ó., López-Quílez, A., & Blangiardo, M. (2018). Assessing the spatial and spatio-temporal distribution of forest species via bayesian hierarchical modeling. *Forests*, *9*(9), 573.

Schittenkopf, C., Deco, G., & Brauer, W. (1997). Two strategies to avoid overfitting in feedforward networks. *Neural networks*, *10*(3), 505–516.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929–1958.

Tan, M., & Le, Q. V. (2021, June). EfficientNetV2: Smaller Models and Faster Training. Retrieved November 22, 2023, from http://arxiv.org/abs/2104.00298

Tang, K., Paluri, M., Fei-Fei, L., Fergus, R., & Bourdev, L. (2015, May). Improving Image Classification with Location Context. https://doi.org/10.48550/arXiv.1505.03873

Terry, J. C. D., Roy, H. E., & August, T. A. (2020). Thinking like a naturalist: Enhancing computer vision of citizen science images by harnessing contextual data. *Methods in Ecology and Evolution*, *11*(2), 303–315. https://doi.org/10.1111/2041-210X.13335

Thompson, P. R., Fagan, W. F., & Staniczenko, P. P. (2020). Predictor species: Improving assessments of rare species occurrence by modeling environmental co-responses. *Ecology and Evolution*, *10*(7), 3293–3304.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Wittich, H. C., Seeland, M., Wäldchen, J., Rzanny, M., & Mäder, P. (2018). Recommending plant taxa for supporting on-site species identification. *BMC Bioinformatics*, *19*(1), 190. https://doi.org/10.1186/s12859-018-2201-7

Xu, J., Zhang, X., Zhao, C., Geng, Z., Feng, Y., Miao, K., & Li, Y. (2023). Improving Fine-grained Image Classification with Multimodal Information. *IEEE Transactions on Multimedia*, 1–14. https://doi.org/10.1109/TMM.2023.3291819

Yang, L., Li, X., Song, R., Zhao, B., Tao, J., Zhou, S., Liang, J., & Yang, J. (2022). Dynamic mlp for fine-grained image classification by leveraging geographical and temporal information. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10945–10954.

Yao, B., Khosla, A., & Fei-Fei, L. (2011). Combining randomization and discrimination for fine-grained image categorization. *CVPR 2011*, 1577–1584. https://doi.org/10.1109/CVPR.2011.5995368

Ying, X. (2019). An overview of overfitting and its solutions. *Journal of physics: Conference series*, *1168*, 022022.

Zhang, S., Gao, Y., Cai, J., Yang, H., Zhao, Q., & Pan, F. (2023). A novel bird sound recognition method based on multifeature fusion and a transformer encoder. *Sensors*, *23*(19), 8099.

Appendices

A Taxonomic subgroup results

In **Tables 10 to 17**, model performances are indicated for different taxonomic subgroups. The highest performance among models is made bold for each metric. A rough number of observations and classes for each subgroup is given, where thousand is denoted as 'K' and million is denoted as 'M'.

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	85.18	85.27	86.71	**86.93**	85.33	85.38
Average Recall	48.82	49.57	52.17	**52.95**	51.20	50.52
ALA	72.23	72.60	75.24	**75.68**	73.07	72.81
ALAR	55.62	56.03	58.32	**59.11**	57.75	57.20

Table 10: Arthropoda: 0.5M observations of 1K classes

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	87.32	88.15	88.29	**88.45**	87.56	88.40
Average Recall	46.58	48.23	50.19	**51.47**	47.20	49.28
ALA	80.50	83.03	84.15	**84.27**	81.12	83.33
ALAR	72.30	74.10	74.44	**74.65**	72.86	74.35

Table 11: Chordata: 4.6M observations of 3K classes

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	85.03	85.29	85.08	85.29	85.70	**85.84**
Average Recall	50.98	51.52	52.76	52.91	53.96	**54.58**
ALA	75.54	76.02	76.90	76.61	76.56	**77.16**
ALAR	64.24	64.57	65.29	64.98	65.65	**66.28**

Table 12: Diptera: 0.9M observations of 2K classes

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	76.84	77.65	77.47	77.52	77.64	**78.56**
Average Recall	40.59	41.46	42.54	42.56	42.07	**43.26**
ALA	67.79	69.05	69.48	69.65	68.82	**70.04**
ALAR	55.24	56.16	55.90	56.33	56.60	**57.10**

Table 13: Fungi: 1.8M observations of 5K classes

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	85.81	86.67	85.46	85.78	86.43	**87.33**
Average Recall	53.27	54.13	53.98	53.92	54.38	**55.21**
ALA	77.13	78.69	78.21	78.43	77.79	**79.74**
ALAR	66.84	67.99	67.72	67.69	67.76	**69.26**

Table 14: Insecta: 3.7M observations of 7K classes

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	93.02	93.53	93.06	93.22	93.35	**93.84**
Average Recall	70.40	71.39	70.53	71.29	71.58	**72.27**
ALA	89.45	90.43	89.74	89.92	89.79	**90.97**
ALAR	84.20	84.96	84.03	84.30	84.76	**85.54**

Table 15: Lepidoptera: 4.6M observations of 4K classes

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	82.46	82.59	**84.39**	84.05	82.58	82.97
Average Recall	52.16	52.00	**56.29**	55.54	54.14	53.97
ALA	72.90	73.70	**77.79**	76.94	73.33	74.56
ALAR	58.69	59.66	**62.94**	61.87	59.85	60.79

Table 16: Mollusca: 0.4M observations of 1K classes

Metric\Model	I (baseline)	ILHD	IN	INLHD	T(IN)	Full Model
Accuracy	77.05	78.15	75.39	75.61	78.49	**79.52**
Average Recall	33.85	36.36	35.76	36.17	36.46	**38.99**
ALA	71.13	73.35	69.58	69.44	73.06	**75.15**
ALAR	61.26	63.10	58.86	58.88	63.40	**65.30**

Table 17: Plantae: 5.9M observations of 9K classes

B Taxonomic subgroup comparisons

In **Tables 18 to 25**, different effects sizes and their significance levels are indicated for comparisons of model versions (ablations) for different taxonomic subgroups. The columns contain the models that were compared, the metric calculated, the effect between model performances when the effect was applied (in percent point increase), model statistic, the p-value, and an indicator if the threshold was passed. For accuracy, the McNemar test statistic was used, which was calculated over the numbers of observations predicted (in)correctly by the compared models. For other metrics, the Wilcoxon signed rank statistic was used, calculated over all class recalls, or over local accuracies or local average recalls with more than 100 observations. The thresholds for significance differ per row because the Holm-Bonferroni method for multiple testing was applied. The details of this correction procedure can be found in **Appendix C**.

Model 1	Model 2	Change	Metric	Effect	Statistic	p-value	Significant?
I	ILHD	adding LHD input	Accuracy	0.09	1.4e-1	7.1e-01	
			Avg. Recall	0.75	4.6e4	1.3e-01	
			ALA	0.37	9.0e0	1.1e-01	
			ALAR	0.41	1.4e1	1.9e-01	
IN	INLHD	adding LHD input	Accuracy	0.22	1.0e0	3.1e-01	
			Avg. Recall	0.78	5.3e4	2.4e-02	
			ALA	0.44	9.0e0	6.4e-02	
			ALAR	0.79	1.6e1	2.8e-01	
T(IN)	Full Model	adding LHD input	Accuracy	0.05	4.3e-2	8.4e-01	
			Avg. Recall	-0.68	4.9e4	2.0e-02	
			ALA	-0.26	1.7e1	8.9e-01	
			ALAR	-0.55	1.3e1	1.6e-01	
I	IN	adding N input	Accuracy	1.53	4.9e1	2.6e-12	✓
			Avg. Recall	3.35	5.1e4	2.3e-17	✓
			ALA	3.01	0.0e0	2.0e-03	
			ALAR	2.70	0.0e0	2.0e-03	
ILHD	INLHD	adding N input	Accuracy	1.66	5.9e1	1.3e-14	✓
			Avg. Recall	3.38	5.4e4	3.0e-17	✓
			ALA	3.08	0.0e0	2.0e-03	
			ALAR	3.08	0.0e0	2.0e-03	
IN	T(IN)	adding transformer	Accuracy	-1.38	4.0e1	2.1e-10	✓
			Avg. Recall	-0.97	7.6e4	1.0e-03	
			ALA	-2.17	5.0e0	2.0e-02	
			ALAR	-0.57	2.5e1	8.5e-01	
INLHD	Full Model	adding transformer	Accuracy	-1.55	5.1e1	9.3e-13	✓
			Avg. Recall	-2.43	6.8e4	2.5e-10	✓
			ALA	-2.87	0.0e0	2.0e-03	
			ALAR	-1.91	0.0e0	2.0e-03	
I	Full Model	replace model	Accuracy	0.20	7.7e-1	3.8e-01	
			Avg. Recall	1.70	5.6e4	1.3e-04	✓
			ALA	0.58	7.0e0	3.7e-02	
			ALAR	1.58	2.0e0	5.9e-03	

Table 18: Arthropoda

Model 1	Model 2	Change	Metric	Effect	Statistic	p-value	Significant?
I	ILHD	adding LHD input	Accuracy	0.83	1.5e2	1.1e-33	✓
			Avg. Recall	1.65	1.1e5	1.2e-39	✓
			ALA	2.53	5.1e1	1.5e-08	✓
			ALAR	1.80	1.4e2	7.5e-07	✓
IN	INLHD	adding LHD input	Accuracy	0.16	5.3e0	2.1e-02	
			Avg. Recall	1.28	3.5e5	1.9e-03	
			ALA	0.12	5.3e2	5.8e-01	
			ALAR	0.21	5.6e2	3.5e-01	
T(IN)	Full Model	adding LHD input	Accuracy	0.84	1.6e2	4.0e-36	✓
			Avg. Recall	2.08	1.2e5	5.4e-50	✓
			ALA	2.21	4.8e1	2.0e-08	✓
			ALAR	1.49	1.5e2	1.3e-06	✓
I	IN	adding N input	Accuracy	0.97	2.0e2	5.5e-46	✓
			Avg. Recall	3.61	2.3e5	7.6e-45	✓
			ALA	3.65	1.0e0	8.0e-10	✓
			ALAR	2.14	9.5e1	1.0e-07	✓
ILHD	INLHD	adding N input	Accuracy	0.30	2.0e1	7.2e-06	✓
			Avg. Recall	3.24	2.5e5	1.8e-32	✓
			ALA	1.24	2.0e2	6.3e-05	✓
			ALAR	0.55	4.9e2	1.1e-01	
IN	T(IN)	adding transformer	Accuracy	-0.73	1.2e2	1.9e-28	✓
			Avg. Recall	-2.99	2.6e5	3.6e-34	✓
			ALA	-3.03	2.3e1	4.4e-09	✓
			ALAR	-1.58	2.2e2	3.0e-05	✓
INLHD	Full Model	adding transformer	Accuracy	-0.05	6.7e-1	4.1e-01	
			Avg. Recall	-2.19	3.0e5	3.9e-16	✓
			ALA	-0.94	2.5e2	5.3e-04	
			ALAR	-0.30	5.2e2	2.6e-01	
I	Full Model	replace model	Accuracy	1.08	2.5e2	6.0e-56	✓
			Avg. Recall	2.70	1.2e5	4.5e-59	✓
			ALA	2.83	6.0e0	1.1e-09	✓
			ALAR	2.05	7.9e1	4.4e-08	✓

Table 19: Chordata

C Holm-Bonferroni procedure

All 256 p-values from model comparisons were sorted from smallest to largest and numbered $P_1 ... P_{256}$. The lowest threshold, applicable to P_1, was $\alpha/m = 0.05/256 = 1.95e-4$. In total, 123 p-values fell below this threshold, making the corresponding effects significant. The next relevant threshold was $\alpha/m = 0.05/(m - 123) = 3.76e-4$. Two more p-values fell below this threshold, and the corresponding effects were deemed significant. The next relevant threshold was $\alpha/m = 0.05/(m - 125) = 3.82e-4$. No other p-values fell below this threshold, so no other effects were deemed significant.

Model 1	Model 2	Change	Metric	Effect	Statistic	p-value	Significant?
I	ILHD	adding LHD input	Accuracy	0.26	2.6e0	1.1e-01	
			Avg. Recall	0.54	1.1e5	1.5e-04	✓
			ALA	0.48	1.9e1	1.1e-02	
			ALAR	0.33	5.2e1	1.5e-01	
IN	INLHD	adding LHD input	Accuracy	0.21	1.6e0	2.1e-01	
			Avg. Recall	0.15	2.5e5	7.6e-01	
			ALA	-0.29	6.0e1	6.8e-01	
			ALAR	-0.31	6.3e1	3.5e-01	
T(IN)	Full Model	adding LHD input	Accuracy	0.14	7.7e-1	3.8e-01	
			Avg. Recall	0.62	2.5e5	7.6e-01	
			ALA	0.60	3.1e1	3.1e-02	
			ALAR	0.63	3.8e1	3.8e-02	
I	IN	adding N input	Accuracy	0.05	9.3e-2	7.6e-01	
			Avg. Recall	1.78	2.6e5	3.0e-09	✓
			ALA	1.36	8.0e0	1.2e-03	
			ALAR	1.05	3.1e1	1.6e-02	
ILHD	INLHD	adding N input	Accuracy	0.0	1.6e-3	9.7e-01	
			Avg. Recall	1.39	2.7e5	4.4e-06	✓
			ALA	0.59	2.8e1	2.2e-02	
			ALAR	0.41	5.1e1	1.4e-01	
IN	T(IN)	adding transformer	Accuracy	0.62	1.5e1	1.2e-04	✓
			Avg. Recall	1.20	3.1e5	6.3e-04	
			ALA	-0.34	6.4e1	5.5e-01	
			ALAR	0.36	6.2e1	3.2e-01	
INLHD	Full Model	adding transformer	Accuracy	0.55	7.7e-1	3.8e-01	
			Avg. Recall	1.67	2.3e5	4.1e-02	
			ALA	0.55	3.1e1	3.1e-02	
			ALAR	1.30	3.8e1	3.8e-02	
I	Full Model	replace model	Accuracy	0.81	2.5e1	5.3e-07	✓
			Avg. Recall	3.60	1.6e5	2.2e-36	✓
			ALA	1.62	1.1e1	4.2e-04	
			ALAR	2.04	1.4e1	8.4e-04	

Table 20: Diptera

Model 1	Model 2	Change	Metric	Effect	Statistic	p-value	Significant?
I	ILHD	adding LHD input	Accuracy	0.81	3.4e1	4.5e-09	✓
			Avg. Recall	0.87	5.0e5	1.0e-13	✓
			ALA	1.26	1.0e0	5.3e-04	
			ALAR	0.92	3.0e0	7.8e-04	
IN	INLHD	adding LHD input	Accuracy	0.05	1.2e-1	7.2e-01	
			Avg. Recall	0.02	2.0e6	8.7e-01	
			ALA	0.17	7.5e1	9.6e-01	
			ALAR	0.43	6.4e1	5.8e-01	
T(IN)	Full Model	adding LHD input	Accuracy	0.92	4.5e1	1.5e-11	✓
			Avg. Recall	1.19	7.1e5	4.4e-16	
			ALA	1.22	2.6e1	1.5e-02	
			ALAR	0.50	3.1e1	3.1e-02	
I	IN	adding N input	Accuracy	0.63	2.1e1	4.4e-06	✓
			Avg. Recall	1.95	1.5e6	7.3e-13	✓
			ALA	1.69	2.7e1	1.7e-02	
			ALAR	0.66	4.4e1	1.3e-01	
ILHD	INLHD	adding N input	Accuracy	-0.13	8.5e-1	3.6e-01	
			Avg. Recall	1.10	1.6e6	6.9e-06	✓
			ALA	0.60	4.0e1	8.9e-02	
			ALAR	0.17	5.6e1	3.5e-01	
IN	T(IN)	adding transformer	Accuracy	0.20	1.4e0	2.4e-01	
			Avg. Recall	-0.47	1.6e6	1.8e-01	
			ALA	-0.66	4.0e1	1.5e-01	
			ALAR	0.70	5.0e1	2.2e-01	
INLHD	Full Model	adding transformer	Accuracy	1.04	5.7e1	3.5e-14	✓
			Avg. Recall	0.70	1.7e6	3.0e-03	
			ALA	0.39	4.3e1	1.2e-01	
			ALAR	0.77	3.2e1	3.5e-02	
I	Full Model	replace model	Accuracy	1.72	1.6e2	6.0e-36	✓
			Avg. Recall	2.67	5.6e5	1.0e-66	✓
			ALA	2.25	1.3e1	1.3e-03	
			ALAR	1.86	1.8e1	3.8e-03	

Table 21: Fungi

Model 1	Model 2	Change	Metric	Effect	Statistic	p-value	Significant?
I	ILHD	adding LHD input	Accuracy	0.86	1.2e1	2.9e-27	✓
			Avg. Recall	0.86	8.5e5	2.0e-23	✓
			ALA	1.56	5.5e1	5.6e-05	✓
			ALAR	1.15	7.6e1	1.5e-04	✓
IN	INLHD	adding LHD input	Accuracy	0.32	1.6e1	7.6e-05	✓
			Avg. Recall	-0.06	3.3e6	9.0e-01	
			ALA	0.22	2.0e2	1.7e-01	
			ALAR	-0.03	2.8e2	7.9e-01	
T(IN)	Full Model	adding LHD input	Accuracy	0.90	1.3e2	5.2e-31	✓
			Avg. Recall	0.83	9.7e5	9.1e-17	✓
			ALA	1.95	5.8e1	4.2e-05	✓
			ALAR	1.50	8.3e1	2.5e-04	✓
I	IN	adding N input	Accuracy	-0.35	1.8e1	1.9e-05	✓
			Avg. Recall	0.71	2.7e6	1.8e-5	✓
			ALA	1.08	9.9e1	1.2e-03	
			ALAR	0.88	1.5e2	9.6e-03	
ILHD	INLHD	adding N input	Accuracy	-0.89	1.2e2	1.2e-28	✓
			Avg. Recall	-0.21	3.0e6	5.4e-01	
			ALA	-0.26	1.9e2	1.2e-01	
			ALAR	-0.30	2.4e2	4.3e-01	
IN	T(IN)	adding transformer	Accuracy	0.97	1.4e2	2.7e-33	✓
			Avg. Recall	0.40	2.8e6	2.7e-02	
			ALA	-0.42	1.7e2	1.1e-01	
			ALAR	0.04	2.9e2	8.8e-01	
INLHD	Full Model	adding transformer	Accuracy	1.55	3.8e2	1.1e-85	✓
			Avg. Recall	1.29	2.5e6	2.0e-10	✓
			ALA	1.31	6.0e1	8.1e-05	✓
			ALAR	1.57	5.9e1	4.6e-05	✓
I	Full Model	replace model	Accuracy	1.52	3.7e2	6.6e-83	✓
			Avg. Recall	1.94	9.8e5	4.2e-75	✓
			ALA	2.61	4.5e1	1.6e-05	✓
			ALAR	2.42	5.7e1	3.9e-05	✓

Table 22: Insecta

Model 1	Model 2	Change	Metric	Effect	Statistic	p-value	Significant?
I	ILHD	adding LHD input	Accuracy	0.51	9.6e1	1.1e-22	✓
			Avg. Recall	0.99	3.5e5	2.8e-32	✓
			ALA	0.98	4.1e1	7.2e-06	✓
			ALAR	0.76	9.6e1	6.9e-05	✓
IN	INLHD	adding LHD input	Accuracy	0.16	9.2e0	2.4e-03	
			Avg. Recall	0.76	1.3e6	4/6e-05	✓
			ALA	0.18	2.6e2	3.9e-01	
			ALAR	0.27	3.1e2	3.4e-01	
T(IN)	Full Model	adding LHD input	Accuracy	0.49	9.5e1	2.3e-22	✓
			Avg. Recall	0.69	5.1e5	1.7e-13	✓
			ALA	1.18	6.5e1	5.8e-06	✓
			ALAR	0.78	1.4e2	4.1e-04	
I	IN	adding N input	Accuracy	0.04	5.3e-1	4.7e-01	
			Avg. Recall	0.13	1.2e6	2.2e-01	
			ALA	0.29	1.8e2	1.6e-02	
			ALAR	-0.17	2.8e2	1.8e-01	
ILHD	INLHD	adding N input	Accuracy	-0.30	3.6e1	1.8e-09	✓
			Avg. Recall	-0.10	1.2e6	9.2e-01	
			ALA	-0.51	1.5e2	1.9e-03	
			ALAR	-0.66	1.7e2	2.2e-03	
IN	T(IN)	adding transformer	Accuracy	0.29	2.9e1	5.8e-08	✓
			Avg. Recall	1.05	1.1e6	1.6e-08	✓
			ALA	0.05	3.1e2	6.6e-01	
			ALAR	0.73	1.5e2	7.0e-04	
INLHD	Full Model	adding transformer	Accuracy	0.62	1.5e2	9.7e-34	✓
			Avg. Recall	0.98	1.0e6	5.2e-09	✓
			ALA	1.05	7.5e1	1.8e-05	✓
			ALAR	1.24	1.3e2	2.3e-04	✓
I	Full Model	replace model	Accuracy	0.82	2.5e2	7.9e-57	✓
			Avg. Recall	1.87	4.2e5	2.0e-70	✓
			ALA	1.52	2.9e1	1.8e-06	✓
			ALAR	1.34	6.0e1	6.7e-06	✓

Table 23: Lepidoptera

Model 1	Model 2	Change	Metric	Effect	Statistic	p-value	Significant?
I	ILHD	adding LHD input	Accuracy	0.13	2.0e-1	6.6e-01	
			Avg. Recall	-0.16	2.7e4	6.4e-01	
			ALA	0.80	2.4e1	1.3e-01	
			ALAR	0.97	3.2e1	2.0e-01	
IN	INLHD	adding LHD input	Accuracy	-0.34	1.5e0	2.3e-01	
			Avg. Recall	-0.75	4.6e4	7.5e-02	
			ALA	-0.85	2.1e1	1.6e-01	
			ALAR	-1.07	2.0e1	2.2e-02	
T(IN)	Full Model	adding LHD input	Accuracy	0.39	1.9e0	1.7e-01	
			Avg. Recall	-0.17	3.9e4	9.5e-01	
			ALA	1.23	1.9e1	1.8e-02	
			ALAR	0.94	2.7e1	6.4e-02	
I	IN	adding N input	Accuracy	1.93	5.0e1	1.9e-12	✓
			Avg. Recall	4.13	4.0e4	2.6e-13	✓
			ALA	4.89	0.0e0	6.1e-05	✓
			ALAR	4.25	0.0e0	6.1e-05	✓
ILHD	INLHD	adding N input	Accuracy	1.46	2.9e1	9.2e-08	✓
			Avg. Recall	3.54	4.2e4	1.4e-09	✓
			ALA	3.24	1.0e1	2.6e-03	
			ALAR	2.21	1.7e1	1.2e-02	
IN	T(IN)	adding transformer	Accuracy	-1.81	4.3e1	6.3e-11	✓
			Avg. Recall	-2.15	5.1e4	1.8e-05	✓
			ALA	-4.46	0.0e0	1.5e-03	
			ALAR	-3.09	1.4e1	6.7e-03	
INLHD	Full Model	adding transformer	Accuracy	-1.08	1.5e1	8.7e-05	✓
			Avg. Recall	-1.57	6.0e4	5.1e-03	
			ALA	-2.38	1.7e1	1.2e-02	
			ALAR	-1.08	3.6e1	1.9e-01	
I	Full Model	replace model	Accuracy	0.51	3.1e0	7.6e-02	
			Avg. Recall	1.81	3.3e4	4.3e-05	✓
			ALA	1.66	9.0e0	6.3e-03	
			ALAR	2.10	1.0e1	7.6e-03	

Table 24: Mollusca

Model 1	Model 2	Change	Metric	Effect	Statistic	p-value	Significant?
I	ILHD	adding LHD input	Accuracy	1.10	2.1e2	6.2e-47	✓
			Avg. Recall	2.51	1.8e6	1.1e-125	✓
			ALA	2.22	1.0e2	3.1e-06	✓
			ALAR	1.84	1.5e2	1.9e-05	✓
IN	INLHD	adding LHD input	Accuracy	0.22	7.5e0	6.2e-03	
			Avg. Recall	0.41	5.5e6	4.4e-02	
			ALA	-0.14	4.8e2	7.1e-01	
			ALAR	0.02	4.7e2	4.3e-01	
T(IN)	Full Model	adding LHD input	Accuracy	1.03	1.9e2	1.9e-42	✓
			Avg. Recall	2.53	1.9e6	8.8e-126	✓
			ALA	2.09	5.6e1	1.9e-07	✓
			ALAR	1.90	9.5e1	1.1e-06	✓
I	IN	adding N input	Accuracy	-1.66	4.5e2	3.1e-99	✓
			Avg. Recall	1.91	5.3e6	8.7e-09	✓
			ALA	-1.55	1.3e2	3.8e-05	✓
			ALAR	-2.40	8.4e1	9.9e-07	✓
ILHD	INLHD	adding N input	Accuracy	-2.54	1.1e3	1.4e-236	✓
			Avg. Recall	-0.19	5.3e6	2.6e-09	✓
			ALA	-3.91	5.9e1	2.3e-07	✓
			ALAR	-2.38	7.1e1	2.9e-07	✓
IN	T(IN)	adding transformer	Accuracy	3.10	1.e3	<1.0e-300	✓
			Avg. Recall	0.70	5.2e6	6.1e-21	✓
			ALA	3.48	2.6e1	4.4e-08	✓
			ALAR	4.54	9.0e0	6.4e-09	✓
INLHD	Full Model	adding transformer	Accuracy	3.91	2.6e3	<1.0e-300	✓
			Avg. Recall	2.82	4.3e6	1.4e-84	✓
			ALA	5.71	3.0e0	4.3e-09	✓
			ALAR	6.42	9.0e0	6.4e-09	✓
I	Full Model	replace model	Accuracy	2.47	1.1e3	1.9e-233	✓
			Avg. Recall	5.14	1.3e6	<1.0e-300	✓
			ALA	4.02	6.0e0	5.2e-09	✓
			ALAR	4.04	1.0e1	6.8e-09	✓

Table 25: Plantae