Self-Supervised Pretraining and Controlled Augmentation Improve Rare Wildlife Recognition in UAV Images

Xiaochen Zheng[1,2] Benjamin Kellenberger[1] Rui Gong[3] Irena Hajnsek[2,4] Devis Tuia[1]
[1]ECEO, EPFL [2]IfU, ETH Zürich [3]CVL, ETH Zürich [4]DLR
xzheng@student.ethz.ch gongr@vision.ee.ethz.ch irena.hajnsek@dlr.de
{benjamin.kellenberger, devis.tuia}@epfl.ch

Abstract

Automated animal censuses with aerial imagery are a vital ingredient towards wildlife conservation. Recent models are generally based on deep learning and thus require vast amounts of training data. Due to their scarcity and minuscule size, annotating animals in aerial imagery is a highly tedious process. In this project, we present a methodology to reduce the amount of required training data by resorting to self-supervised pretraining. In detail, we examine a combination of recent contrastive learning methodologies like Momentum Contrast (MoCo) and Cross-Level Instance-Group Discrimination (CLD) to condition our model on the aerial images without the requirement for labels. We show that a combination of MoCo, CLD, and geometric augmentations outperforms conventional models pretrained on ImageNet by a large margin. Crucially, our method still yields favorable results even if we reduce the number of training animals to just 10%, at which point our best model scores double the recall of the baseline at similar precision. This effectively allows reducing the number of required annotations to a fraction while still being able to train high-accuracy models in such highly challenging settings.

1. Introduction

Wildlife censuses help determine the exact number and the spatial-temporal distribution of wild animals, which is vital to assess living conditions and potential survival risks of wildlife species [19, 2, 29]. Since recently, these hitherto manual surveys of wildlife reserves are increasingly replaced with counts derived automatically from images acquired by Unmanned Aerial Vehicles (UAVs), paired with deep learning models to automate the wildlife recognition task [19, 18, 30, 20, 10]. These models are typically supervised, pretrained on large-scale curated datasets such as ImageNet [7], MS-COCO [22] and then fine-tuned on the target imagery. Irrespective of the type of fine-tuning, this last step requires thousands of animals to be annotated, and hence expert knowledge. Meanwhile, large UAV campaigns can generate images in high numbers [19], with many containing either large numbers of animals, or none [20], both of which cases are cumbersome for manual annotators. Furthermore, the vastness of wildlife reserves mean that wildlife is a rare sight and background dominates the majority of images. This causes two problems: on the one hand, the datasets are strongly imbalanced toward the absence class; on the other hand, the objects of interest are

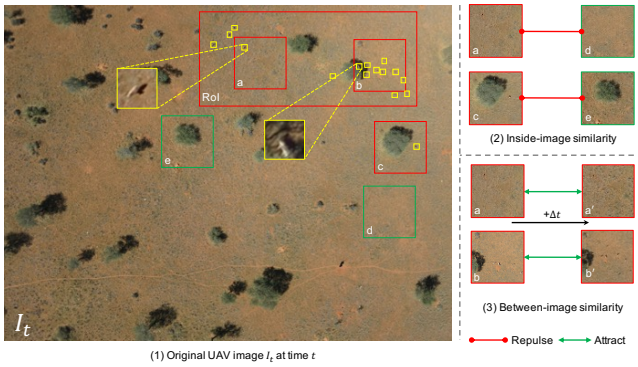

Figure 1: Overview of Kuzikus dataset. In (1), the region of interest (RoI) (red bounding box) only occupies a small part of original UAV image. Wildlife (yellow bounding boxes) is tiny which is hard to be recognized and labeled. That makes supervised learning difficult. Foreground (wildlife)/background crops are marked with red/green bounding boxes. Patches a, c, d, e are cropped from different locations of the same image. Patches a, a', b, b' are cropped from same locations of different images (sampling time interval is Δt). In (2) similar patches containing wildlife should be distinguished (repulsed) from the ones without wildlife. Whereas in (3) similar patches from the same category should be attracted. (2) and (3) are difficult for supervised learning and contrastive learning.

extremely small compared with the original aerial image size, as illustrated in Figure 1. These two issues significantly downgrade the capacity of a supervised model to solve the recognition task [19]. Hence, methods to reduce the labelling requirement are urgently needed.

A promising direction to this end is to use self-supervised learning (SSL), where models are first trained in a *pretext* task on the target images without the need for manually provided labels, and then fine-tuned on the actual objective (*downstream task*) with manual labels [23]. Earlier *pretext* tasks required models to reconstruct transformations between different *views* of the same image. Recently, focus has shifted to contrastive learning. Here, augmentation as a method of image transformation is still employed, but with a different objective: while traditional SSL methodologies forced the model to learn representations within one data point to *e.g.*, in-paint cut-out regions [27, 40], contrastive learning employs transformations in a comparison scheme and encourages the model to learn representations by maximizing the similarity between two randomly augmented *views* of the same data point, resp. image (positive pairs) and dissimilarity between different data points (negative pairs) [38, 13, 5, 34, 4, 12, 24]. The different *views* of same instance are randomly generated from a stochastic data augmentation module. Recent works have identified a stronger augmentation strategy to be vital for improved learning [5]. However, augmentation functions need to be carefully selected with respect to the problem and data at hand [39]. Choosing inadequate functions may result in removal of important information (*e.g.*, random resized cropping may remove animals at the border of UAV images). In contrast, some functions benefit certain scenarios more than others (*e.g.*, random vertical flips and rotations may be of limited use with natural images, but may provide strong learning signals for view-independent aerial images).

However, applying self-supervised learning techniques, such as contrastive learning, to UAV images on wildlife is challenging. One such problem is the requirement of contrastive learning methods to receive dissimilar imagery. In UAV acquisitions, as illustrated in Figure 1, high image sampling frequencies will generate strong autocorrelations between acquisitions in short time intervals (similar to adjacent frames in a video). Also, the vastness of many wildlife areas result in consistent characteristics, and hence in repeated or similar patches in the dataset. Many instance discrimination based methods, such as NPID [38], MoCo [13, 5], and SimCLR [4], are based on the assumption that each instance is significantly different from others and that each instance can be treated as a separate category. The large similarity between training images mean that the negative pairs used in the contrastive learning process is likely to be composed of highly similar instances, which in turn compromises feature representations due to incorrect repulsion between similar images. Instead of exploring the effect of hard negative sampling [16, 31], in this paper we propose to solve these problems with Cross-Level Instance-Group Discrimination (CLD) [37], which aims to deal with highly correlated datasets. Furthermore, we hypothesize that top view UAV imagery should be *invariant* to geometric transformations, *e.g.*, rotation. From this perspective, we propose to apply extra geometric transformation to contrastive model, which captures invariant information of UAV images introduced by different augmentations.

We propose an SSL model to pretrain wildlife recognition models based on contrastive learning. Our work build on the work of MoCo [13, 5] and Cross-Level Instance-Group Discrimination (CLD) [37]:

- We propose a methodology for image-level wildlife recognition with a reduced number of annotations by self-supervised pretraining.
- We show that using self-supervised pretraining outperforms supervised ImageNet pretraining on downstream recognition task.

We further find that applying controlled augmentation to self-supervised pretraining and fine-tuning the pretrained model with few labels will outperform ImageNet pretraining fine-tuned with all available training labels. Our self-supervised pretraining learns representations of natural wildlife scenes more efficiently than supervised pretraining.

2. Related Work

Self-Supervised Learning in UAV imagery. Unlike in the field of classic computer vision, self-supervised learning of aerial images has not yet been fully studied. Stojnic *et al*. [32] apply Contrastive Multiview Coding [35] to learn aerial image representations on both RGB and multispectral remote sensing images. [17] proposes a method based on contrastive learning with different image augmentations. [33] analyze different pretext tasks, *e.g.*, inpainting [27], context prediction [9], and contrastive learning with different image augmentations on remote sensing dataset. Besides, [1] use geo-location classification as the pretext task. The encoder is trained by predicting the global geo-location of input image. Tong *et al*. [36] generate pseudo-labels of unlabeled UAV imagry data to improve the downstream classification accuracy. Most of the downstream tasks in aerial imaery domain are scene classification, *e.g.*, land-cover or land-use [15] classification. [1] apply self-supervised learning to transferred downstream tasks, *e.g.*, object detection and image segmentation. Different from those tasks, our task is more domain specific, which is tiny and rare wildlife recognition in the wild.

Pretext tasks. Self-supervised representation learning is designed to solve certain pretext tasks. [11] uses the rotation angle as pseudo label and learn underlined structure

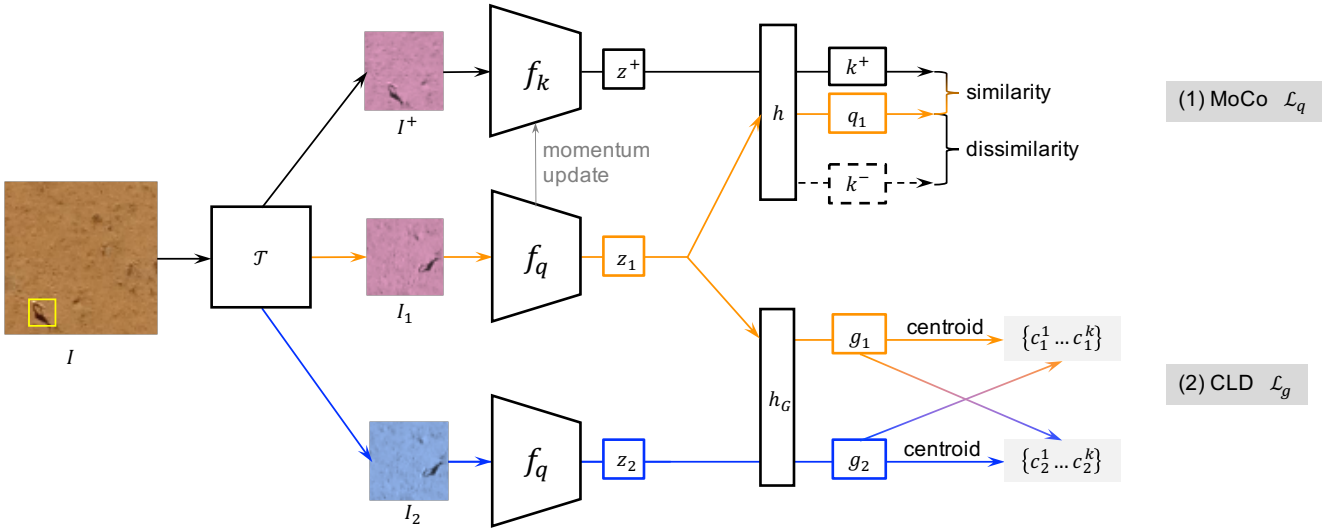

Figure 2: Overview of the employed SSL framework, consisting of MoCo [13] (upper part) and CLD [37] (lower part). Firstly, the two different *views* I_1 and I^+ of the same input I are encoded by f_q and f_k respectively. Then, the two representations z_1 and z^+ are projected into an embedding space. q and k are representations of query and key in the hypersphere. Query q_1 and its positive key k^+ are from the different augmented *views* of the same input and negative keys k^- are encoded from different inputs (dashed bounding box). CLD first encodes two different *views* I_1, I_2 of the same instance, then applies a different projection head and projects the representations from the same query encoders to a different embedding space from (1). Finally, a local K-Means clustering is used to find the k centroids of a batch of inputs. The centroid of assigned cluster g_1 can be served as positive key of view g_2, and vice versa.

of the objects by predicting rotation angle. [9, 23, 8] perform region-level relative location prediction. Other missions like color in-painting [27, 40], and solving jigsaw puzzles [25] are also applied as pretext tasks. In contrastive learning, learning between different augmented views are used as pretext tasks. In this work, we apply a instance discrimination task [38] in addition to geometric invariant mapping to contrastive self-supervised models.

Contrastive Representation Learning. Recently, the most competitive representation learning method without labels is self-supervised contrastive learning. Contrastive methods [4, 13, 5, 12, 3, 26, 23, 28, 24] train a visual representation encoder by attracting positive pairs from the same instance in latent space while repulsing negative pairs from different instances. One of the most important parts in contrastive learning is the selection of positive and negative pairs [35] for instance discrimination. To create positive pairs without prior label information, one common way is to generate multiple views of input images. [5, 13, 4] apply a stochastic augmentation module to randomly augment an input image twice. [39] proposes a new model with multi-augmentation and multi-head. It constructs multiple embeddings and captures varying and invariant information introduced by different augmentations. Methods combined contrastive learning with online clustering [3, 37, 21] are proposed to boost the performance of self-supervised learning which explore the data manifold to learn image representations by capturing invariant information.

3. Method

3.1. Contrastive Learning Framework

We apply MoCo [13] as our unsupervised learning method. MoCo is a mechanism for building dynamic dictionaries for contrastive learning. MoCo defines "query" and "key", which are representations encoded by encoder networks. Given a batch of inputs, query and positive key are from two different augmentation *views* of the same input whereas query and negative keys are from *views* of different inputs. The dynamic dictionary stores both positive and negative keys. With MoCo, self-supervised learning can be regarded as a training process to perform dictionary look-up. MoCo learns image representations by matching an encoded query q to a dictionary of encoded keys k using a contrastive loss. Query q should be similar to its matching key, positive key k^+, and dissimilar to negative keys k^-. As illustrated in Figure 2, the MoCo model consists of five parts:

A *stochastic data augmentation* module $\mathcal{T}$ [4] transforms one given input image I into different augmented

views with randomly applied augmentations, denoted I_i and I^+ for query q_i and positive key k^+ [13]. We sequentially apply five augmentations (Base and Color Aug. in Table 1) similar as MoCo v2 [5].

A *base encoder* f_q for query q maps the augmented *views* into feature space: $f_q : I_i, \rightarrow \boldsymbol{z}_i$, where $I_i \in \mathbb{R}^{C\times W\times H}, \boldsymbol{z}_i \in \mathbb{R}^s$, where z denotes the s-D encoded *representation*. The parameters are updated by backpropagation. This takes the form of a CNN in our case.

A *momentum-updated encoder* f_k for keys k shares the same structure with *base encoder* f_q and is initialized with the same parameters. However, they are not learned through backpropagation during training; instead, the parameters of $f_k(\cdot)$ are updated with a momentum mechanism [13].

A *projection head* projects the *representations* $\boldsymbol{z}_i$ into a unit hypersphere: $h : \boldsymbol{z}_i \rightarrow \boldsymbol{q}_i$, where $\boldsymbol{q}_i \in \mathbb{R}^m$ and $||\boldsymbol{q}_i|| = 1$, the same for $\boldsymbol{z}^+$. The similarity is measured by a dot product.

A *dynamic dictionary* holds the prototypical feature for all instances [38, 4, 37]. It is implemented as a queue of fixed size, fed with the stream of mini-batches that are used for training: in current mini-batch, the encoded representations are enqueued, and the oldest are dequeued [13].

A popular choice of contrastive loss for positive pairs $(\boldsymbol{q}, \boldsymbol{k}^+)$ and negative pairs $(\boldsymbol{q}, \boldsymbol{k}^-)$ is InfoNCE [26], denoted as $\mathcal{L}_q(\boldsymbol{q}, \boldsymbol{k}^+)$

$$\mathcal{L}_q(\boldsymbol{q}, \boldsymbol{k}^+) = -\log \frac{\exp\left(\boldsymbol{q}\cdot\boldsymbol{k}^+/\tau\right)}{\exp\left(\boldsymbol{q}\cdot\boldsymbol{k}^+/\tau\right) + \sum_{k^-}\exp\left(\boldsymbol{q}\cdot\boldsymbol{k}^-/\tau\right)} \tag{1}$$

where the dictionary contains K negative samples and τ denotes the temperature parameter, which is the hyperparameter scaling the distribution of distances [26, 13].

3.2. Clustering Based Contrastive Learning

The key idea of CLD [37] is to cluster instances locally, and perform contrastive loss to centroids and image representaions. Therefore, similar instances are clustered into the same group and the false rejection of instances with high similarity is alleviated, as illustrated in Figure 4. CLD uses two different *views* of the same instance as input. As such, the CLD branch shares the same query encoder f_q with MoCo but uses a different projection head h_G, as illustrated in the lower part of Figure 2.

To perform CLD, the unit-length features $\boldsymbol{g}_i$ of all instances in a mini-batch are first extracted from f_q and h_G. Then, CLD implements local k-means clustering to $\boldsymbol{g}_i$ for a mini-batch of instances and finds k local cluster centroids $\{\boldsymbol{c}_i^1, \ldots, \boldsymbol{c}_i^k\}$ with $\boldsymbol{g}_i$ assigned to $\boldsymbol{C}(\boldsymbol{g}_i)$. The same operation is performed to the other branch I_j from all instances in a mini-batch, denoted as $\boldsymbol{g}_j$, $\{\boldsymbol{c}_j^1, \ldots, \boldsymbol{c}_j^k\}$, and $\boldsymbol{C}(\boldsymbol{g}_j)$. CLD applies the contrastive loss between $\boldsymbol{g}_i$ and clustering of the other branch $\{\boldsymbol{c}_j^1, \ldots, \boldsymbol{c}_j^k\}$. Each cluster contains highly similar instances, and the assigned centroids together with representations from the other branch can be regarded as positive pairs. Namely, the centroids of the *other* clusters act as negative samples [37]. Thus, feature vector $\boldsymbol{g}_i$ and its counterparts $\boldsymbol{g}_j$ assigned centroid $\boldsymbol{C}(\boldsymbol{g}_j)$ comprise positive pairs and all *other* centroids comprise negative pairs. The local contrastive loss for CLD is

$$\mathcal{L}_g(\boldsymbol{g}_i, \boldsymbol{C}(\boldsymbol{g}_j)) = -\log \frac{\exp\left(\boldsymbol{g}_i\cdot\boldsymbol{C}(\boldsymbol{g}_j)/\tau\right)}{\sum_{\{\boldsymbol{c}_j^k\}}\exp\left(\boldsymbol{g}_i\cdot\boldsymbol{c}_j^k/\tau\right)} \tag{2}$$

where $\{\boldsymbol{c}_j^k\}$ denotes the set of k centroids from the other branch. Thus, the loss of a dual-branch CLD in Figure 2 is:

$$\mathcal{L}_g(\boldsymbol{g}_1, \boldsymbol{C}(\boldsymbol{g}_2)) + \mathcal{L}_g(\boldsymbol{g}_2, \boldsymbol{C}(\boldsymbol{g}_1)) \tag{3}$$

Module	PyTorch-like Augmentation
Base Aug.	RandomCrop(224)*
	RandomHorizontalFlip(p=0.5)
	GaussianBlur([0.1, 2.0])
Color Aug.	ColorJitter(0.4, 0.4, 0.4, 0.1)
	RandomGrayscale(p=0.2)
Rot. Aug.	RandomRotation()**

Table 1: Overview of the employed random augmentation strategies. * To avoid information loss on tiny animals, we apply random crops without resizing. ** We randomly rotate the images by $\{90°, 180°, 270°\}$.

3.3. Augmentation Strategies

State-of-the-art contrastive learning [4, 6, 5] between multiple *views* of the data employs stronger augmentation strategies to improve performance. The choices of different *views* have a marked impact on the performance of self-supervised pretraining [5, 41, 39, 35]. For different branches in CLD, *e.g.*, I_1 and I_2 in Figure 2, we apply multiple augmentations to the same input image [39]. We keep each branch invariant to one specific augmentation transformation. For example, I_1 and I^+ are always augmented by the same color but different rotation augmentation while I_2 and I^+ are always augmented by same rotation but different color augmentation. Augmentation parameters are sampled randomly and independently from the stochastic augmentation module $\mathcal{T}$ as outlined in Table 1. We project the queries and key into one embedding space, keeping the embedding space invariant to all augmentations. We aim to add extra geometric transformations on top of the CLD framework. The loss of our proposed augmentation strategies has the same form of Equation (1) and (2).

3.4. Total contrastive loss

We combine CLD [37] with MoCo v2 [5] and construct a total contrastive loss over *views* I_1, I_2, and I^+ with CLD weight λ in a mini-batch. We apply different temperatures τ_q and τ_g for instance and group branches, respectively. The total contrastive loss is [37]:

$$\begin{aligned}\mathcal{L}_{tot} =& \frac{1}{2}\left[\mathcal{L}_q(\boldsymbol{q}_1, \boldsymbol{k}^+) + \mathcal{L}_q(\boldsymbol{q}_2, \boldsymbol{k}^+)\right] \\ &+ \lambda \times \frac{1}{2}\left[\mathcal{L}_g(\boldsymbol{g}_1, \boldsymbol{C}(\boldsymbol{g}_2)) + \mathcal{L}_g(\boldsymbol{g}_2, \boldsymbol{C}(\boldsymbol{g}_1)\right]\end{aligned} \tag{4}$$

where $\boldsymbol{q}_{\{1,2\}}$ and $\boldsymbol{g}_{\{1,2\}}$ are feature representations issued from augmented versions of the original samples, generated following the procedure described in Section 3.3.

4. Experiments

4.1. Study Area and Data

In this work, we use the data from [19], consisting of RGB aerial images acquired with a SenseFly eBee* UAV over the Kuzikus Wildlife Reserve in Namibia† by the SAVMAP consortium‡. The UAV's flight height varied between 120 and 160m, resulting in a resolution of 4 to 8 cm with the given camera (Canon PowerShot S110). The images were annotated with bounding boxes for animals in a crowdsourcing campaign led by MicroMappers§; these annotations were then refined in several iterations by the authors. This resulted in a total of 1183 animals. We derived the **K**uzikus **W**ildlife **D**ataset **Pre**-training (KWD-Pre) and **K**uzikus **W**ildlife **D**ataset **L**ong-**T**ail distributed (KWD-LT) for pre-traning and fine-tuning/downstream task.

Technical Challenges. Most contrastive models are trained on curated dataset with unique characteristics, *e.g.*, ImageNet [7]. In these datasets, images contain only a single object which is located in the center of the image (*object-centric*). And objects have *discriminative* visual features. The datasets also have uniformly distributed classes. In contrast, domain-specific datasets (*e.g.*, our KWD) contain less discriminative visual feature, making it hard to distinguish between similar and small objects *e.g.*, small trees, wildlife from the top view.

KWD-Pre. We apply the same patches creating procedure as described in [19]. We randomly crop 15 patches for every original 4000×3000 image. The size of each patch is 256×256 pixels to save memory and have a larger batch size. We randomly crop 15 extra patches if one image contains animals. Cropping this way increases the chances of extracting patches containing animals for training, but we do not retain any labeled information nor bounding box location. As this can be seen as a form of weak supervision in the patch extraction process, we do not know whether each patch contains animal(s) or not while applying random cropping. So the prior knowledge of classes and locations is not exploited by self-supervised learning.

KWD-LT. The original images are taken by UAVs on different dates and times [19]. We first split the original data into train, test, and validation set with a ratio of 8:1:1. Then, for the background class, we apply a random cropping procedure (512×512 pixels) to the original images and verify each patch to make sure it contains no animal. For the foreground (wildlife) class, we apply a random cropping procedure (224×224 pixels) around the ground truth bounding boxes to make sure each patch contains whole animal(s) body. We choose three different random seed to random cropping procedures of train, test, and validation set to make sure the cropping position is different. The train set is class imbalanced and long-tail distributed with a foreground-to-background ratio of $\frac{1}{18}$. The test and validation set are class balanced. In the experiments below, we evaluate fine-tuning on KWD-LT with different percentages of annotated animals to investigate the benefit of SSL for reduced annotation efforts.

4.2. Experimental Setup

Models. For the supervised models, we use a ResNet-50 [14], pretrained on ImageNet [7]. Sup1 freezes the output of ResNet-50 average pooling layer. Sup2 fine-tunes the pretrained ResNet-50 on KWD-LT with full labels, as shown in Figure 2. We apply the MoCo v2 as our contrastive baseline model, denoted as MCC0. Instead of using a RandomResizeCrop, we apply a PyTorch RondomCrop to input images to keep more information. MCC1 and MCC2 are all MoCo v2 model with CLD. We apply our augmentation strategies to MCC2. We set the λ, number of clusters to 0.25 and 32 respectively. Detailed information of different models are outlined in Table 2.

Optimizer. We use stochastic gradient descent for self-supervised pretraining and downstream task fine-tuning. For the self-supervised pretraining, we apply the same cosine decay scheduler as proposed in [37]. For the semi-supervised fine-tuning, the initial learning rate is 0.01 and we likewise apply a cosine decay schedule [4]. For the downstream task, we set the initial learning rate as 30 and we apply same strategy proposed in [13].

Base Encoder, Projection Head. We apply a ResNet-50 without pretrained on ImageNet as our base encoder. We simply remove the last fully-connected layer and use the output of average pooling as feature vector $\boldsymbol{z}_i$. We adopt a Multi-Layer Perceptron (MLP) head following [4, 5], which is a 2-layer MLP (2048-dimensional hidden layer, with ReLU). We share the hidden layer and apply a different final

*https://www.sensefly.com
†https://kuzikus-namibia.de
‡http://lasig.epfl.ch/savmap
§https://micromappers.wordpress.com

Name	Backbone	pretraining			Fine-tuning	CLD	Augmentation Strategies
		Supervised	Unsupervised	Dataset			
Sup1	ResNet-50	✓		ImageNet			
Sup2	ResNet-50	✓		ImageNet	✓		
MCC0	MoCo v2		✓	KWD-Pre			
MCC1	MoCo v2		✓	KWD-Pre		✓	
MCC2	MoCo v2		✓	KWD-Pre		✓	✓

Table 2: Overview of models we use.

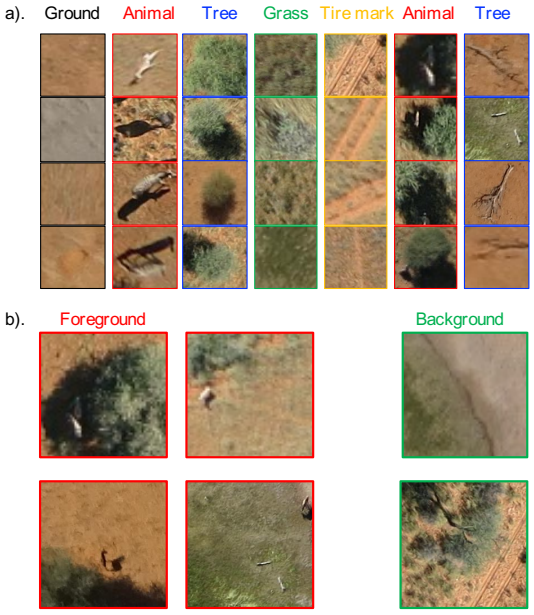

Figure 3: Overview of KWD-LT dataset. (a) denotes all possible elements in the dataset. Especially the animal elements in the right are animals beneath the tree and the tree elements in the right are dead tree trunks. This is extremely hard to be recognized [19]. It is also difficult to distinguish between dead tree trunks and animal. Intuitively, all examples in the KWD-LT dataset consist of randomly combined elements in (a). (b) is examples of KWD-LT dataset. Instances are image-level annotated. Foreground represent images containing wildlife.

layer of the MLP head to MoCo branch and CLD branch. The dimension of the unit-length feature representation q_i and $k^{\{+,-\}}$ is 128.

Hyperparameter Choice. For fair comparison and avoiding hyper-parameters tuning redundancy, we select the CLD weight λ and number of cluster by linear classification on frozen features with labeled data. According to our prior knowledge, the images in the KWD dataset are primarily composed of artefacts as follows: animal, tree, grass, tire mark (road), animal beneath tree, and dead tree trunk, as illustrated in Figure 3. Ideally, the number of all possible clusters is therefore $C_6^0+C_6^1+C_6^2+C_6^3+C_6^4+C_6^5+C_6^6 = 2^6 = 64$. Among all elements, animals beneath tree are hard to recognize; also dead tree trunks are easy confused with animals. Meanwhile, with a batch size of 64, it might not yield all 64 combinations, but we still use 64 clusters to give the model enough freedom. We train the MCC1 model with different hyper-parameters for 200 epochs and select the best hyper-parameter combination based on accuracy on the validation set of the downstream recognition task.

Downstream Recognition Task. We verify different models by applying linear classification on encoded image representations. We follow the same common linear classification protocol as [13]. We first perform self-supervised pretraining on KWD-Pre dataset. Then we perform two kinds of experiments: (1) *frozen* features: we freeze the output features of the global average pooling layer of a ResNet and train a linear classifier (a fully-connected layer followed by softmax) [13] in a supervised way on our KWD-LT downstream task dataset; (2) *end-to-end*: we fine-tune the base encoder and linear classifier by softmax loss instead of contrastive loss. And we report the linear classification top-1 accuracy on the KWD-LT validation set, as well as recall and precision for foreground class.

5. Results and Discussion

5.1. Hyperparameter Choice

The results for the ablation studies on the number of clusters for CLD and λ are shown in Tables 3 and 4. We can clearly see that the model performs almost equally when $\lambda = 0.25$ or $\lambda = 1$. Hence, we chose the default $\lambda = 0.25$ value as in [37]. However for the number of clusters, there does not seem to be an obvious trend. we speculate that (1) grouping projected *representations* by k-means clustering is hard to perform well, (2) the model is not able to recognize animals beneath the tree or/and to distinguish between dead tree trunks and animals (number of clusters = 16 or 32). When number of clusters = 16, 32, or 64, the accuracy is equally good. Therefore, we chose the best recall with

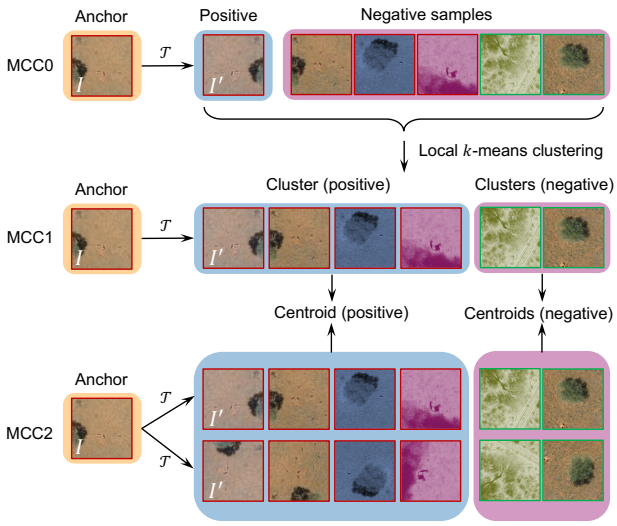

Figure 4: Illustration of positive and negative samples in MCC0, MCC1, and MCC2 scenarioes. Anchor I is the original image patch on KWD dataset. I' is the augmented view of I. The red/green bounding boxes represent the foreground/background classes. In MCC0, negative samples might contain images from the same category of the anchor (positive samples). That causes the false repulsion. In MCC1 and MCC2, after applying local k-means clustering to all samples, positive and negative samples can be grouped into different categories. With CLD, there is less repulsion between samples in the same category.

Model	λ	Acc	Prec	Rec
MCC1	0.1	86.2	98.6	73.4
MCC1	**0.25**	**88.4**	98.9	**77.4**
MCC1	0.5	86.4	98.9	73.5
MCC1	1	88.1	98.5	77.2
CLD	1	60.9	98.9	22.1

Table 3: Hyper-parameter λ selection. All models are pretrained on KWD-Pre using the MCC1 strategy.

number of cluster = 32.

5.2. Main Results

Results of self-supervised pretraining. The possible repeated and highly correlated patches slow the training process and lower the performance of SSL pretraining. As it breaks the instance discrimination presumption described in Section 1. For fair comparison, we evaluate the performance of self-supervised pretraining by linear classification of *frozen* features with full labels. The linear classifier accuracy in Table 5 shows that SSL pretraining on target dataset

Model	Clusters	Acc	Prec	Rec
MCC1	**16**	88.3	100.0	76.5
MCC1	30	87.3	99.7	75.0
MCC1	**32**	**88.4**	98.9	**77.4**
MCC1	48	87.2	99.7	74.9
MCC1	**64**	**88.4**	100.0	76.9

Table 4: Number of clusters selection. All models are pretrained on KWD-Pre using the MCC1 strategy with $\lambda = 0.25$.

Model	Epochs*	Acc	Prec	Rec
Sup1	-	86.7	96.5	76.1
Sup2	200	88.6	99.3	77.7
MCC0	200	82.2	97.7	65.6
MCC1	200	88.4	98.9	77.4
MCC2	150	90.8	99.6	81.9

Table 5: **Linear classifier** top-1 accuracy (%), foreground class precision and recall (%) on *frozen* features with full labels, comparison of self-supervised learning on KWD-Pre (MCC0, MCC1, MCC2) and supervised pretraining on ImageNet (Sup1, Sup2). * We adopt the peak performance epoch to compare different methods.

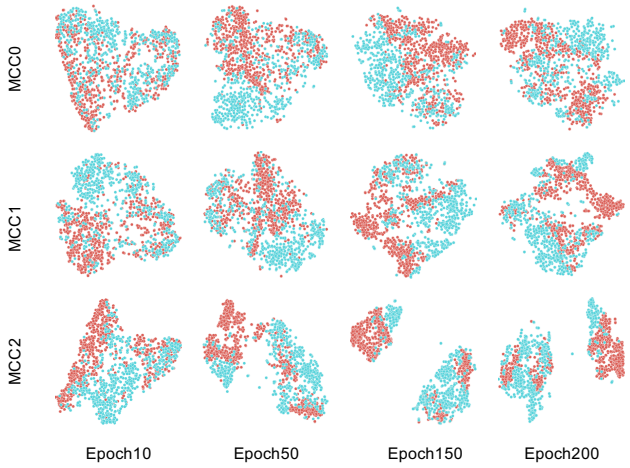

Figure 5: **t-SNE feature visualization** of MCC0, MCC1, and MCC2 on KWD-LT. MCC2 has earlier and better separation between foreground and background classes (indicated by color) than MCC0 and MCC1.

is a strong competition of supervised pretraining on ImageNet. MoCo with CLD (MCC1) perform equally as well as supervised fine-tuning (Sup2). With controlled and designed augmentation for our scenario, MCC2 outperforms Sup2 by 2.2%. MCC2 at epoch 150 outperforms MCC0 at

Pretraining dataset	SSL strategy	Fine-tuning (on KWD-LT)	1% labels			10% labels			20% labels		
			Acc	Prec	Rec	Acc	Prec	Rec	Acc	Prec	Rec
ImageNet	-	*end-to-end*	50.4	0	0	69.8	98.8	40.2	80.5	97.3	62.8
		frozen features	54.4	100.0	9.0	69.6	99.3	39.3	76.2	99.5	52.7
KWD-Pre	MCC0	*end-to-end*	68.0	98.5	35.9	76.8	99.7	53.4	77.9	99.7	55.4
		frozen features	74.0	97.1	49.1	76.9	96.8	55.1	77.4	97.3	56.1
KWD-Pre	MCC1	*end-to-end*	70.5	98.5	40.9	76.0	99.7	51.6	88.5	99.3	77.4
		frozen features	71.9	94.8	46.0	82.5	98.2	66.0	83.8	98.2	68.8
KWD-Pre	MCC2	*end-to-end*	78.9	98.0	58.7	90.7	99.8	81.3	**91.9**	100	83.7
		frozen features	**83.4**	97.0	68.6	**91.7**	98.8	84.5	90.1	99.1	81.4

Table 6: **Animal recognition** accuracy when using a portion of the available labeled samples in the final classifier (*frozen* features) or in the base encoder (*end-to-end*) and when varying the type of pretraining, the self-supervised strategy and the type of fine tuning. Prec = Precision, Rec = Recall.

epoch 200 by 8.6% (82.2% vs. 90.8%) with a faster converging. We can tell that SSL pretraining with controlled augmentation can improve the performance of rare wildlife recogniztion. However, the accuracy of MCC2 in epoch 200 is lower than that in epoch 150. That might imply that adding geometric transformation might cause the problem of overfitting. And feature visualization in Figure 5 show that CLD with geometric augmentation (MCC2) converges faster and better towards a more distinctive feature representation than MCC0 and MCC1.

Results of Recognition Task. Self-supervised pretraining on our dataset can utilize annotations far more efficiently than supervised pretraining on ImageNet. As shown in Table 6, fine-tuning the encoder *end-to-end* will completely destroy the capacity of ImageNet pretrained model. Even though we freeze the feature representations, the model perform poorly with small fraction of labeled instances. However, fine-tuning only linear classifier on *frozen* features with 1% and 10% annotations outperforms fine-tuning the encoder *end-to-end*. When we only train the linear classifier with 1% of labels, MCC2 outperforms MCC0 by 9.4% and MCC1 by 11.5%. For MCC1, fine-tuning encoder with 10% annotations outperforms the recognition accuracy with full annotations. For MCC2, we need 20% annotations to get a better result. Meanwhile, for MCC1, fine-tuning the encoder with 20% of labeled instances will have the same performance with fine-tuning ImageNet pretrained model (Sup2) with full labeled instances. Whereas for MCC2, only 10% labeled data is required to outperform the Sup2 supervised model. The results show that SSL model can learn more information through geometric invariant mapping and capturing geometric invariant information can benefit UAV top view imagery task.

Influence of Label Fraction. The results of Sup1, Sup2, MCC0, and MCC1 in Table 5 and of these *frozen* features in Table 6 show that increasing the fraction of labels used can improve the performance of linear classification. But it has a bottleneck performance of 88.6%. This is different in training linear classifier on MCC2. Although adding extra geometric augmentation can boost performance. Feeding all annotated data can decrease the performance of *frozen* features in MCC2, as shown in Table 5 and 6. That could be caused by imbalanced classes: the model overfits to the background class and is over-confident.

6. Conclusion

In this paper, our proposed strategy reduces the label requirements in the wildlife recognition tasks. The problem is introduced by applying supervised learning to automated animal censuses in largely remote areas with aerial imagery, in which scenario the annotations are expensive to be obtained. The contrastive self-supervised pretraining with domain-specific geometric transformation outperforms the performance of fine-tuning ImageNet pretrained model with full labels. Results show that the geometric invariant mapping method can capture information more efficiently of wildlife in UAV images than method without geometric augmentation. Extensive experiments further prove the effectiveness of recognizing rare wildlife with reduced labels.

Acknowledgments. The authors would like to thank the Kuzikus Wildlife Reserve, Namibia[¶] for the access to the aerial data and the ground reference used in this study and Mei Sun of IfU, ETH Zürich, Xia Li of CVL, ETH Zürich for their helpful discussion.

References

[1] Kumar Ayush, Burak Uzkent, Chenlin Meng, Kumar Tanmay, Marshall Burke, David Lobell, and Stefano Ermon.

[¶] https://www.kuzikus-namibia.de

Geography-aware self-supervised learning. *arXiv preprint arXiv:2011.09980*, 2020. 2

[2] Andrew Balmford, Leon Bennun, Ben Ten Brink, David Cooper, Isabelle M Côté, Peter Crane, Andrew Dobson, Nigel Dudley, Ian Dutton, Rhys E Green, et al. The convention on biological diversity's 2010 target. *Science*, 307(5707):212–213, 2005. 1

[3] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020. 2, 3, 4, 5

[5] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv: 2003.04297*, 2020. 2, 3, 4, 5

[6] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 4

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 1, 5

[8] Jian Ding, Enze Xie, Hang Xu, Chenhan Jiang, Zhenguo Li, Ping Luo, and Gui-Song Xia. Unsupervised pretraining for object detection by patch reidentification. *arXiv preprint arXiv:2103.04814*, 2021. 3

[9] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015. 2, 3

[10] Jasper AJ Eikelboom, Johan Wind, Eline van de Ven, Lekishon M Kenana, Bradley Schroder, Henrik J de Knegt, Frank van Langevelde, and Herbert HT Prins. Improving the precision and accuracy of animal population estimates with aerial image object detection. *Methods in Ecology and Evolution*, 10(11):1875–1887, 2019. 1

[11] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[12] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3, 4, 5, 6

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 5

[15] Neal Jean, Sherrie Wang, Anshul Samar, George Azzari, David Lobell, and Stefano Ermon. Tile2vec: Unsupervised representation learning for spatially distributed data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019. 2

[16] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[17] Jian Kang, Ruben Fernandez-Beltran, Puhong Duan, Sicong Liu, and Antonio J Plaza. Deep unsupervised embedding for remotely sensed images based on spatially augmented momentum contrast. *IEEE Transactions on Geoscience and Remote Sensing*, 59(3):2598–2610, 2020. 2

[18] Benjamin Kellenberger, Diego Marcos, Sylvain Lobry, and Devis Tuia. Half a percent of labels is enough: Efficient animal detection in UAV imagery using deep cnns and active learning. *IEEE Transactions on Geoscience and Remote Sensing*, 57(12):9524–9533, 2019. 1

[19] Benjamin Kellenberger, Diego Marcos, and Devis Tuia. Detecting mammals in UAV images: Best practices to address a substantially imbalanced dataset with deep learning. *Remote Sensing of Environment*, 216:139–153, 2018. 1, 2, 5, 6

[20] Benjamin Kellenberger, Diego Marcos, and Devis Tuia. When a few clicks make all the difference: Improving weakly-supervised wildlife detection in UAV images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019. 1

[21] Junnan Li, Pan Zhou, Caiming Xiong, and Steven Hoi. Prototypical contrastive learning of unsupervised representations. In *International Conference on Learning Representations (ICLR)*, 2021. 3

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014. 1

[23] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3

[24] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3

[25] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 3

[26] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 3, 4

[27] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature

learning by inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 3

[28] Hadsell Raia, Chopra Sumit, and LeCun Yann. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006. 3

[29] Christophe F Randin, Michael B Ashcroft, Janine Bolliger, Jeannine Cavender-Bares, Nicholas C Coops, Stefan Dullinger, Thomas Dirnböck, Sandra Eckert, Erle Ellis, Néstor Fernández, et al. Monitoring biodiversity in the anthropocene using remote sensing in species distribution models. *Remote Sensing of Environment*, 239:111626, 2020. 1

[30] Nicolas Rey, Michele Volpi, Stéphane Joost, and Devis Tuia. Detecting animals in african savanna with UAVs and the crowds. *Remote Sensing of Environment*, 200:341–351, 2017. 1

[31] Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *International Conference on Learning Representations (ICLR)*, 2021. 2

[32] Vladan Stojnic and Vladimir Risojevic. Self-supervised learning of remote sensing scene representations using contrastive multiview coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2021. 2

[33] Chao Tao, Ji Qi, Weipeng Lu, Hao Wang, and Haifeng Li. Remote sensing image scene classification with self-supervised paradigm under limited labeled samples. *IEEE Geoscience and Remote Sensing Letters*, 2020. 2

[34] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[35] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive representation learning? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2, 3, 4

[36] Xin-Yi Tong, Gui-Song Xia, Qikai Lu, Huanfeng Shen, Shengyang Li, Shucheng You, and Liangpei Zhang. Land-cover classification with high-resolution remote sensing images using transferable deep models. *Remote Sensing of Environment*, 237:111322, 2020. 2

[37] Xudong Wang, Ziwei Liu, and Stella X. Yu. Unsupervised feature learning by cross-level instance-group discrimination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 3, 4, 5, 6

[38] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 3, 4

[39] Tete Xiao, Xiaolong Wang, Alexei A Efros, and Trevor Darrell. What should not be contrastive in contrastive learning. In *International Conference on Learning Representations (ICLR)*, 2021. 2, 3, 4

[40] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 2, 3

[41] Nanxuan Zhao, Zhirong Wu, Rynson W. H. Lau, and Stephen Lin. What makes instance discrimination good for transfer learning? In *International Conference on Learning Representations (ICLR)*, 2021. 4