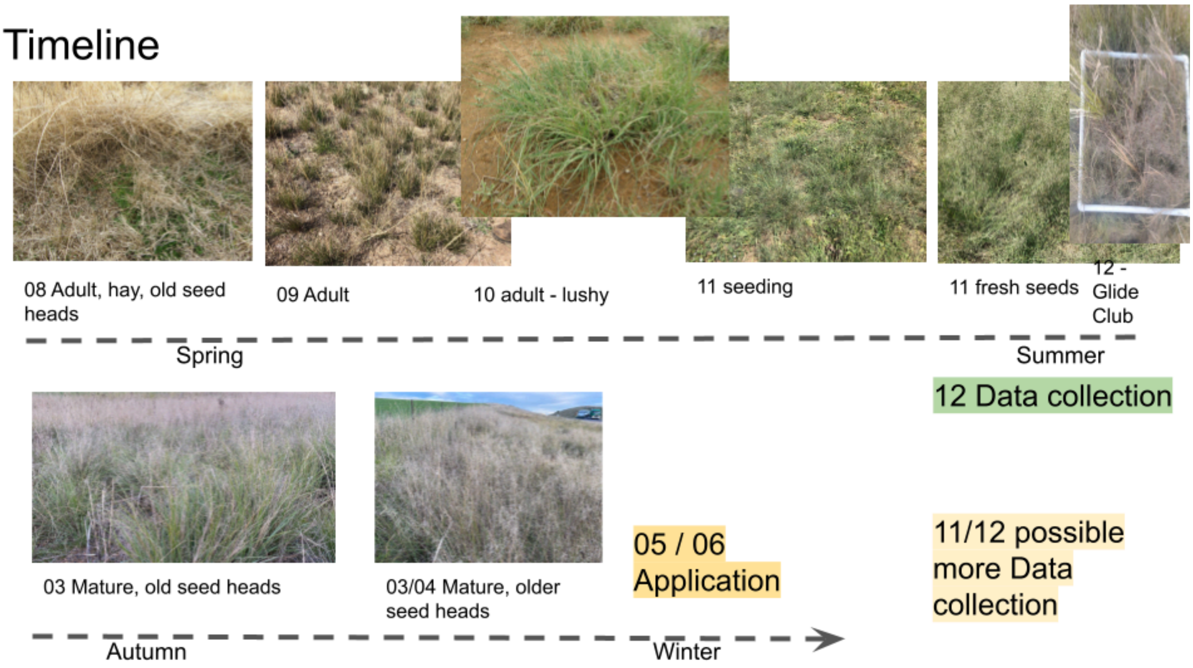
ALG Report

Nicolas Mandel, v1: 2024 07 17
v2: 2024 07 18
Code available at: https://github.com/WeeddetectionAI/ALG

Introduction

African Lovegrass (ALG), Eragrostis curvula, is a grass indigenous to Africa, which has spread to Australia and is considered an invasive species, as it threatens livestock foodstuff. ALG is itself not edible and diminishes the volume of edible content in a pasture. It requires monitoring and treatment through herbicides or other control measures, however, before it can be treated it needs to be identified.

From an elevated aerial perspective, such as from UAVs, it is difficult to identify, as the fine strands of the grass intertwine with other grasses. Hence, the outline of the grass is almost impossible to delineate, making for an interesting computational detection problem:

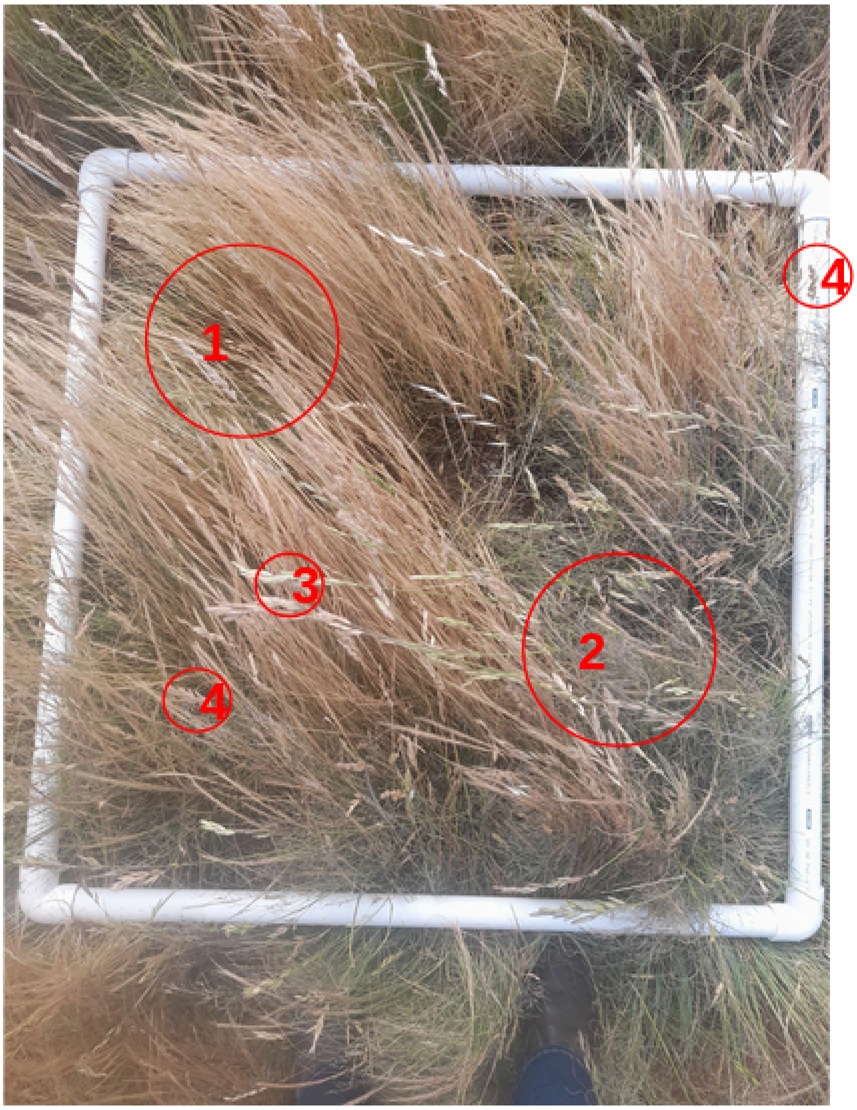


The Figure shows the growth cycle of ALG, from old seed heads through to maturity across the seasons.
it is evident that the grass is difficult to delineate and moves from a yellow-coloured status through green and lush states, to a grey-purple flowering status before repeating the cycle over.

Similar Species

As it is a grass, there are plenty of species that look alike and also appear in the same environments. Of these, some include:

- Soft Brome
- Prairie Grass
- Vulpia
- Barley Grass
- Wild Oats



The Figure shows a quadrat collected at the Glide Club site in december 2022, with multiple false positive species:

1. Vulpia
2. ALG
3. Prairie Grass
4. Soft Brome

To the uneducated eye, these may be almost indistinguishable. In machine learning, these are termed “false positives”, as they resemble ALG and may be identified as such.

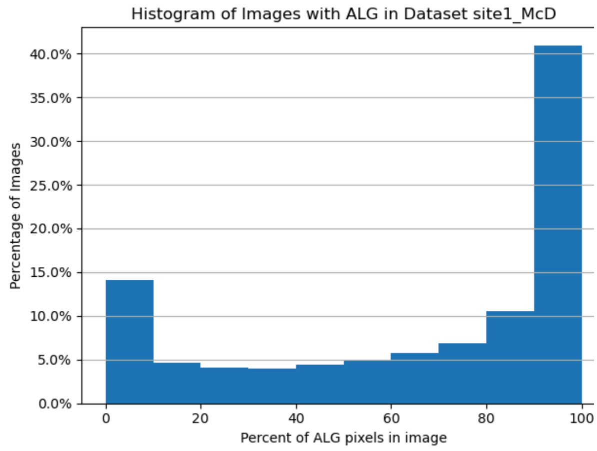
Datasets

The datasets were generated by using models trained on Multispectral (MS) imagery, overlaid with RGB orthomosaics. One model was developed for each growth stage, one for site 1 and tested on site 2, and another on site 4 and tested on site 3. The pixel-wise labels were generated from the MS model and propagated to the overlaid higher-resolution RGB model by upscaling.

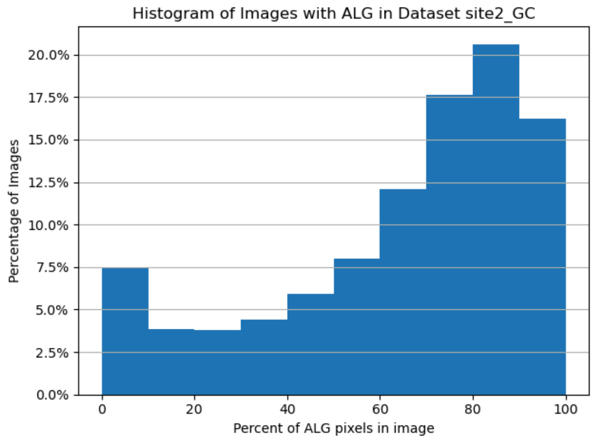
Datasets	Name	Date of RGB	Type of RGB	Type of MS	Date of MS	Growth Stage	Quadrats	GroundTruth Quadrats	Images before cleaning	Images after cleaning WHITE	Images after cleaning BLACK
Site 1	Mc Donald's	13/12/2022	Fuji 50	MicaSense Altum	13/12/2022	Flowering	10	Link to OneDrive	8900	8901	7747
Site 2	Gliding Club	14/12/2022	DJI_P1 40m	MicaSense Altum	15/12/2022	Flowering	10		15130	15130	10775
Site 3	Kuma Nature Reserve	06/12/2023	Fuji NR	MicaSense RedEdge	05/12/2023	Vegetative, no flowerheads	19		21584	21584	13719
Site 4	TSR	Dec 2023	Fuji NR	MicaSense RedEdge	05/12/2023	Vegetative, no flowerheads	20		51376	24292	24292

In total, 59 quadrats are available which have been validated on the ground. 56533 images in total are available, from 2 cameras and multiple GSDs, collected during two different years and vegetative states. The mask images are labeled in the following way: Pixel value 0 denotes ALG pixels, 127 non-alg vegetation and 255 denotes non-vegetation pixels. Datasets are cleaned by removing all images and their associated label files, where all pixels are either black or white - these are boundary areas from orthomosaics that have been tiled:
Dataset histograms are calculated by counting the amount of pixels with label “ALG” and displaying the buckets that these have. [1]

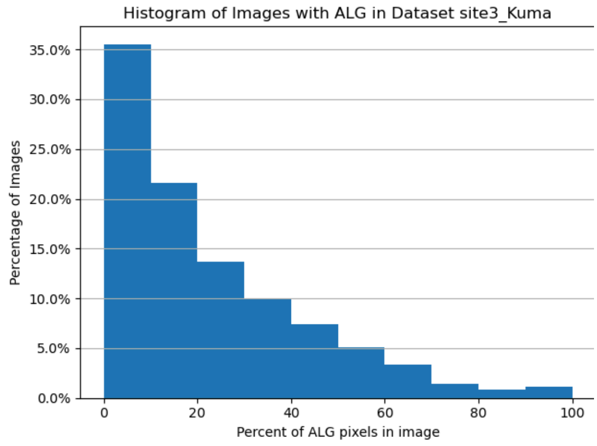
Site 1 - McDonald’s



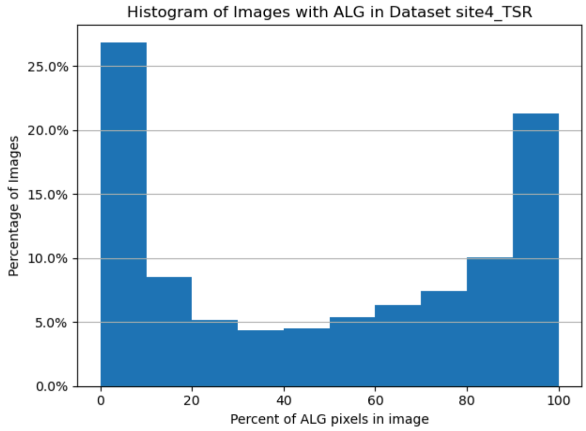
Site 2 - Gliding Club



Site 3 - Kuma

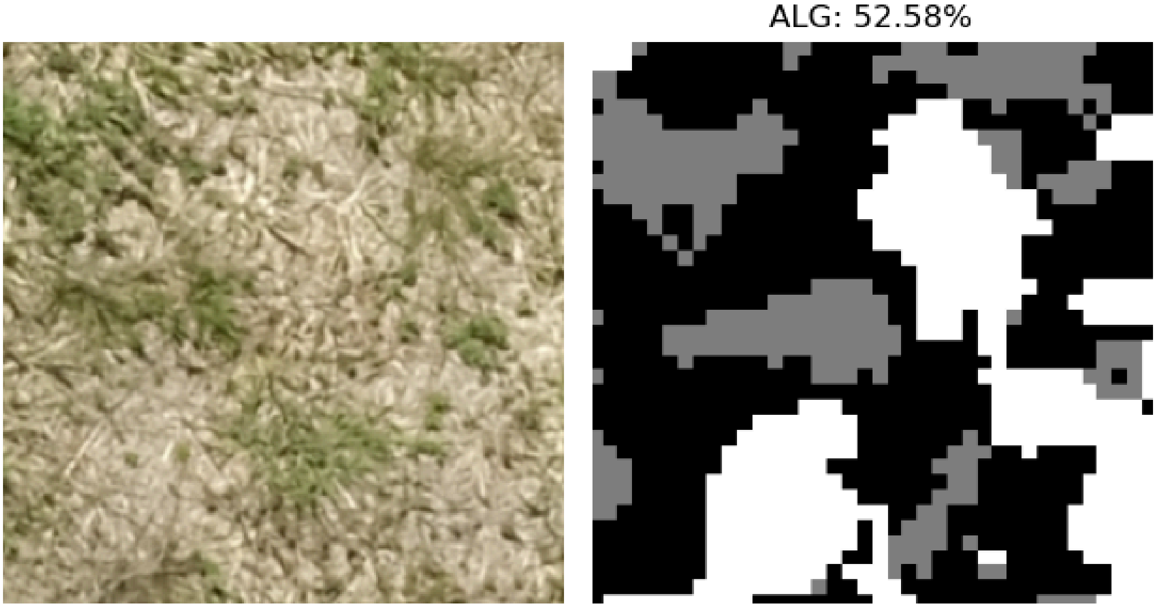


Site 4 - TSR



These histograms show the number of image with a certain percentage of ALG at each site. It can be observed from the histograms that e.g. for Kuma, site 3, there are a lot of images which have 0-10% of their pixels noted with class 0 for ALG, while for example for Glide Club there appear to be a lot of images which are almost entirely made up of ALG.

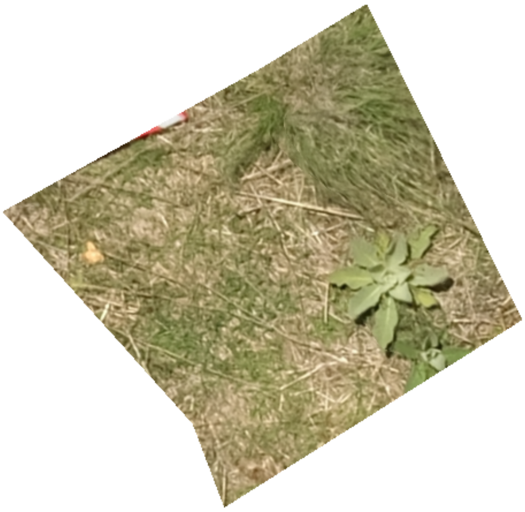
McD and TSR appear both to have a lot of images which are either fully ALG or not at all. This can be interpreted that McD and TSR either have large patches of ALG / non-ALG or even separate areas where there is either or. For Kuma this would indicate a lot of areas where there is no ALG at all, or if it is, then it is very sparse. The GC site appears to be almost entirely covered by ALG.



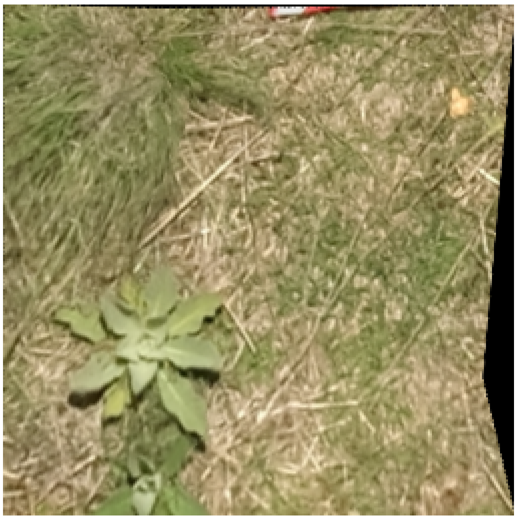
The figure shows an example ALG sample, where black values indicate ALG, gray indicates vegetation and white non-vegetation. 52.58 % of all pixels are considered ALG.

Test Datasets

The performance of models is tested on the square-corrected quadrats of the other site. So models trained on site 1 for flowering ALG are tested on the quadrats of site 2. Models trained for vegetative ALG on site 4 are tested on quadrats of site 3. The label for the quadrats is generated from the validation spreadsheets and calculating the percentage of ALG in the quadrat. The example shows the square-corrected quadrat 10 at Site 4 - TSR. 256 x 256



The uncorrected image, loaded into a contemporary deep learning library such as pytorch, would have a lot of 0-padding values around the borders. These would highly distort the prediction of the filters, hence not yielding representative results, which is the reason for the square-correction.



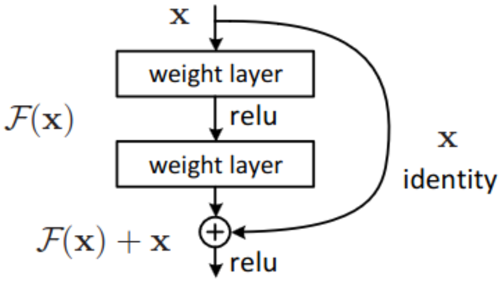
Models

A suite of models has been developed as part of this research. ALG is difficult to delineate and for monitoring and management purposes, it is not as important to know where exactly certain plants are located, but more that there is a presence and a spread. Treatment cannot be targeted at individual plants but must be done effectively over large areas.

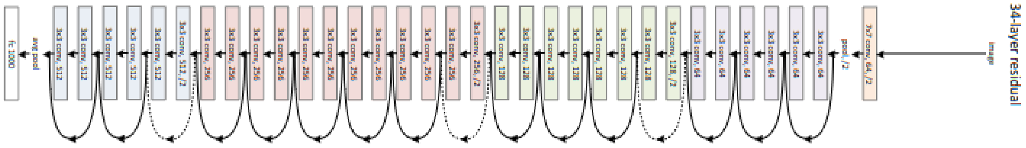
Therefore, classification of 256 x 256 patches was chosen as a suitable computer vision task. Training samples are described above. Models were developed on site 1 and site 4 and tested on the quadrat images from site 2 and 3 respectively, so flowering and vegetative datasets separately.

ResNet models, one of the most successful models in computer vision, were used for detection [2], which despite their age and being outperformed by more recent transformer models, have been shown effective in remote-sensing tasks on low volume data [3].

Central to ResNets are skip connections, which skip blocks and append the input to the output, so that the layers in between only need to learn the difference, similar to Gaussian integrators.


x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

This approach enabled deeper models and common ResNet variants are 18, 34, 50, 101 and 152. The figure shows a 34-layer network.



Preliminary experiments on a subset of the data showed that ResNets 34 to 101 were effective and fewer-layer models showed degrading performance, while models with more layers took too long to train for diminishing returns in performance.

Three variants of the training were used to asses the impact:

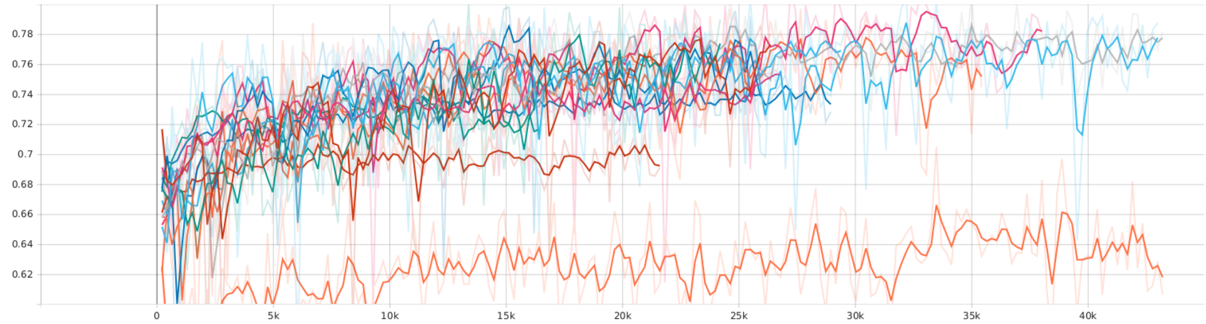
1. training a model from random weights
2. training a model from an imagenet variant
3. transfer learning from imagenet, with only the last layer learnable

The following image augmentations were used to reduce overfitting and enable the models to potentially generalise better:

- Flipping / Rotations with a probability of 0.5
- HSV, Gamma, Brightness Contrast and Gaussian noise with a probability of 0.5
- Elastic or Optical Distortions with a probability of 0.5
- Normalisation to
  - mean 0.485, 0.456, 0.406
  - sd 0.229, 0.224, 0.225

Models are trained with the Adam optimizer in a pytorch-lightning implementation with a learning rate of 0.001, batch sizes of 16 and early stopping with 20 epochs patience and a maximum of 200 epochs.

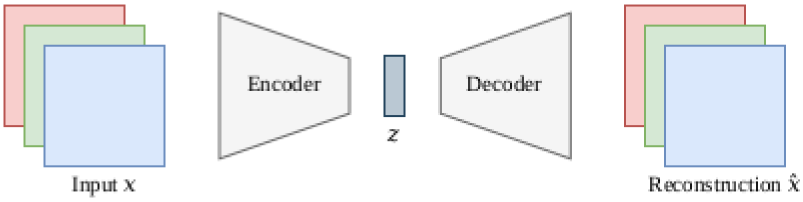
In preliminary experiments on site 4, all ResNet models exhibit a similar accuracy, between 60 and 80 %. Smaller models generally perform on par with larger models and often better, with accuracies in the high 70s range. Only finetuning the last layer does not yield major benefits, indicating that the domain of images is vastly different from the default imagenet domain, however, transfer learning yields results comparable to learning from scratch. While smaller networks appear to perform similarly independent of training regime, larger networks exhibit major differences, with only fine tuning the last layer performing poorly in comparison.



Resnet	Variant	Validation Accuracy	Test Accuracy	Model Epoch	
18	vanilla	0.93	0.6875	93	
	transfer	0.93	0.625	88	appears to converge earlier
	finetune	0.9	0.625	57	drop in accuracy
34	vanilla	**0.94**	0.75	136	
	transfer	**0.94**	**0.875**	116	appears to converge earlier
	finetune	0.89	0.625	76	drop in accuracy
50	vanilla	0.93	0.8125	186	
	transfer	0.93	0.6875	100	appears to converge earlier
	finetune	0.9	0.625	10	drop in accuracy
101	vanilla	**0.94**	0.6875	168	
	transfer	**0.94**	0.75	98	appears to converge earlier
	finetune	0.89	0.625	88	drop in accuracy
152	vanilla	0.93	0.8125	185	
	transfer	0.93	0.75	152	appears to converge earlier
	finetune	0.91	0.75	68	drop in accuracy

AutoEncoders

Autoencoders are models that attempt to encode and compress the data that they are presented with. This works by funnelling the information through a “bottleneck” layer z and calculating a loss on the reconstruction x_hat. This way, the bottleneck layer z is enabled to contain a more compact representation of the information in input x.



ResNets have been shown to be effective Autoencoders on the example of small-scale 32x32 pixel datasets [4]. Common sizes for z are 512, 256 or 1024, which is much smaller than 32x32x3 (R, G, B) - 3072 parameters.
Autoencoder variants with 32x32 pixels and 256 x 256 pixels have been used. While in the case of 256 x 256 pixels there is a lot more context visible, the information necessary to be compressed into the bottleneck layer is far larger and there are more parameters to be learned by the model and decoder. In this case, 196608 parameters would have to be compressed into the bottleneck layer.

Unlabeled Datasets

For all sites there are multiple camera raw images available. These are used to train the autoencoders. Raw images are from the following sites, cameras GSDs and resolutions:

Site	**Camera**	**GSD**	**Resolution**	**MB per image**	**Image Count**
Site 1 - McD	DJI P1		8192 x 5460	14	227
	Fuji	50 m	4000 x 3000	6	249
	Phase 1		11664 x 8750	80	100
Site 2 - GC	DJI P1	40 m	8192 x 5460	20	231
	DJI P1	80 m	8192 x 5460	20	70
	Phase 1		11664 x 8750	90	57
Site 3 - Kuma	DJI P1		5280 x 3956	5	14
	Fuji		11648 x 8736	63	99
	Phase 1		11664 x 8750	80	100
Site 4 - TSR	DJI P1 007		5280 x 3956	5	4
	DJI P1 008		5280 x 3956	6	5
	Fuji		11648 x 8736	63	99
	Phase 1		11664 x 8750	82	100

Because image resolution is large and memory loading is difficult for files of that size at runtime, between two and three thousand 256 x 256 crops are generated, which are then used as sample inputs with random crops of 32x32 at runtime to train autoencoders.

Orthomosaic-Base-Datasets

Sites	Camera	GSD	Image Resolution	Image Size in MB	Image Count
Site 1 - McD	Fuji GFX100	0,22 cm	4000 x 3000	6	249
Site 2 - GC	DJI P1 40m	0,48 cm	8192 x 5460	16	231
Site 3 - Kuma	Fuji GFX100	0,22 cm	11648 x 8736	70	165
Site 4 - TSR	Fuji GFX100	0,24 cm	11648 x 8736	70	285

For specific autoencoders, only the base datasets were used that also created the orthomosaics, to stay within the same camera generation space.

Experiments

In normal cases, the volume of labels that is available for this kind of research is limited and the quality is questionable, since all labels are required to be generated by qualified labelers [5]. ALG is hard to identify by hand and only 59 images have been validated on the ground, while for this research there are 56533 labeled images available, even though these have been generated by MS imagery.

For this research, the impact of training with lower volume datasets is tested, by only using subsets of the available labeled data of 10 or 1 %.

Furthermore, it is tested whether pre training with autoencoders yields a benefit, so that model capacity is improved.

Results

Training happens on one set of sites and test accuracy is presented on the validated quadrats of the other dataset. E.g for flowering images, training happens on all MS-generated images from site 1 and only the validated quadrats with numbers estimated by hand on site 2, while for vegetative training on site 4 and testing on site 3. Combined fuses both datasets into a single training and testing regime. This is not just the average of both because of the imbalance of image volume in training and testing. Validation accuracy represents the accuracy during training epochs on a subset of training data withheld from training. Test accuracy is on the quadrats from the other site. It is given as percentage of images correctly identified.

		Flowering			**Vegetative**			**Combined**		
Resnet	**Variant**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**
34	transfer	**0.93**	**0.9**	72	**0.94**	0.4737	73	**0.93**	0.5172	**53**
50	transfer	**0.93**	**0.9**	78	0.93	0.4737	119	**0.93**	0.5862	76
101	transfer	**0.93**	0.7	174	**0.94**	0.4737	104	0.92	0.5172	109
18	ae-32	**0.93**	0.7	125	0.91	0.5263	**20**	**0.93**	0.5162	101
18	ae-256	**0.93**	**0.9**	**33**	0.93	0.4211	107	0.92	0.5862	97
34	ae-32	0.92	**0.9**	37	0.93	**0.5789**	56	**0.93**	**0.6207**	66

The table shows that the models have a much higher accuracy detecting flowering ALG than vegetative. ResNets pretrained with autoencoders achieve comparable accuracies, often with less training epochs, despite being smaller in general (and therefore requiring less weights). It is evident that vegetative ALG is much more difficult to detect with RGB imagery. Autoencoder pre training does not improve the test accuracy significantly for vegetative data. However, training epochs after pre training with autoencoders are much lower than training transfer learning models pre trained on imagenet.

Note that the models trained with autoencoders are smaller than the other models. The largest Autoencoder model is the smallest normal model.

Reducing the Volume of Data

10 %

		Flowering			**Vegetative**			**Combined**		
Resnet	**Variant**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**
34	transfer	0.94	**0.9**	**15**	**0.94**	0.3684	64	**0.93**	**0.5862**	147
50	transfer	**0.95**	**0.9**	39	0.93	**0.5789**	97	**0.93**	0.5172	109
101	transfer	0.94	0.7	95	0.91	0.5263	**37**	0.92	0.4483	97
18	ae-32	0.93	0.4	64	0.9	0.3684	58	0.91	0.4483	**60**
18	ae-256	0.92	0.8	18	0.91	0.4737	40	0.89	0.4483	**60**
34	ae-32	0.92	**0.9**	22	0.91	0.3158	53	0.91	0.5172	66

Reducing the volume of data to 10% yields marginal increases in the validation accuracy, which is an indicator for overfitting to the training data. Test performance for Autoencoder pretraining drops significantly, while training time reduces by a large margin to less than a hundred epochs.

1 %

		Flowering			**Vegetative**			**Combined**		
Resnet	**Variant**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**	**Validation Accuracy**	**Test Accuracy**	**Model Epoch**
34	transfer	**1**	0.8	23	0.92	**0.4737**	16	**0.91**	0.4138	12
50	transfer	**1**	0.8	23	**0.96**	**0.4737**	13	**0.91**	0.4828	23
101	transfer	**1**	0.5	32	0.92	0.3684	19	**0.91**	0.5172	29
18	ae-32	0.94	**0.9**	11	**0.96**	0.4211	31	**0.91**	**0.5862**	17
18	ae-256	**1**	0.7	40	0.94	0.3684	**7**	**0.91**	0.5517	42

34	ae-32	0.88	0.3	**9**	0.94	**0.4737**	17	0.86	0.3103	**9**

Reducing the volume of data to 1% of the original data reduces training time until convergence even further. The perfect validation accuracy for flowering data is a clear indicator of overfitting, while the test accuracy drops slightly.

Summary

Reducing the volume of data does not yield a significant drop in test accuracy and for the binary case, a performance fluctuating around 50% indicates a performance as good as tossing a coin. While it appears that flowering ALG can be detected fairly well, even with only 1% of the volume of data available, even large volumes of data do not enable models to reliably detect vegetative ALG from RGB images. This is an indicator that the information contained in the additional channels that MS has available has a far larger discriminative power than the visual channels RGB.#

Cross-Testing developed models

Cross testing developed models is used to confirm performance of model across unseen data. E.g. models developed on flowering ALG are tested on vegetative datasets.

Dataset Developed				**Flowering**	**Vegetative**	**Combined**
Volume	**Dataset**	**Resnet**	**Variant**	**Test Accuracy**	**Test Accuracy**	**Test Accuracy**
100.00%	**Flowering**	34	transfer	**0.9**	**0.6842**	**0.7586**
7747		50	transfer	**0.9**	**0.7368**	**0.7931**
		101	transfer	0.7	0.5789	0.6207
		18	ae-32	0.7	**0.6842**	**0.69**
		18	ae-256	**0.9**	0.2631	0.4828
		34	ae-32	**0.9**	0.5789	**0.69**
24,292	**Vegetative**	34	transfer	**0.9**	0.4737	0.621
		50	transfer	0.7	0.4737	0.552
		101	transfer	0.8	0.4737	0.586
		18	ae-32	0.8	0.5263	0.552
		18	ae-256	0.8	0.4211	0.621
		34	ae-32	0.8	0.5789	0.655
32,039	**Combined**	34	transfer	0.6	0.4737	0.5172
		50	transfer	**0.9**	0.421	0.5862
		101	transfer	0.8	0.3684	0.5172
		18	ae-32	0.8	0.3684	0.5162
		18	ae-256	**0.9**	0.421	0.5862
		34	ae-32	**0.9**	0.4737	0.6207
10.00%	**Flowering**	34	transfer	**0.9**	0.5789	0.6897
774		50	transfer	**0.9**	**0.78947**	**0.82759**
		101	transfer	0.7	**0.7368**	**0.72414**
		18	ae-32	0.4	**0.7368**	0.6207
		18	ae-256	0.8	**0.7368**	**0.757**
		34	ae-32	**0.9**	0.579	0.69
2,429	**Vegetative**	34	transfer	0.8	0.3684	0.5172
		50	transfer	0.7	0.5789	0.621
		101	transfer	0.7	0.5263	0.5862
		18	ae-32	0.7	0.3684	0.4828
		18	ae-256	0.7	0.4737	0.55172

		34	ae-32	0.8	0.3158	0.4827
3,203	**Combined**	34	transfer	**0.9**	0.421	0.5862
		50	transfer	0.8	0.3684	0.5172
		101	transfer	0.7	0.3158	0.4483
		18	ae-32	0.8	0.2632	0.4483
		18	ae-256	0.6	0.3684	0.4483
		34	ae-32	0.7	0.42105	0.5172
1.00%	**Flowering**	34	transfer	0.8	**0.6842**	**0.7241**
77		50	transfer	0.8	**0.7368**	**0.7586**
		101	transfer	0.5	0.6316	0.5862
		18	ae-32	**0.9**	0.474	**0.6208965517**
		18	ae-256	0.7	0.36842	0.482757931
		34	ae-32	0.3	**0.7368**	0.5861793103
242	**Vegetative**	34	transfer	0.7	0.4737	0.55
		50	transfer	0.8	0.4737	0.5862
		101	transfer	0.7	0.3684	0.4828
		18	ae-32	0.8	0.4211	0.5517551724
		18	ae-256	0.7	0.3684	0.4827448276
		34	ae-32	0.7	0.4737	0.5517344828
319	**Combined**	34	transfer	0.7	0.2631	0.4138
		50	transfer	0.8	0.3684	0.4828
		101	transfer	**0.9**	0.2631	0.5172
		18	ae-32	**0.9**	0.42	0.5862
		18	ae-256	**0.9**	0.3684	0.5517
		34	ae-32	0.3	0.3158	0.3103

The largest three values per dataset volume are always highlighted, with the dataset used for training listed in the second column. It is clearly visible that datasets developed on flowering ALG data outperform both models including vegetative data, even on the vegetative data.

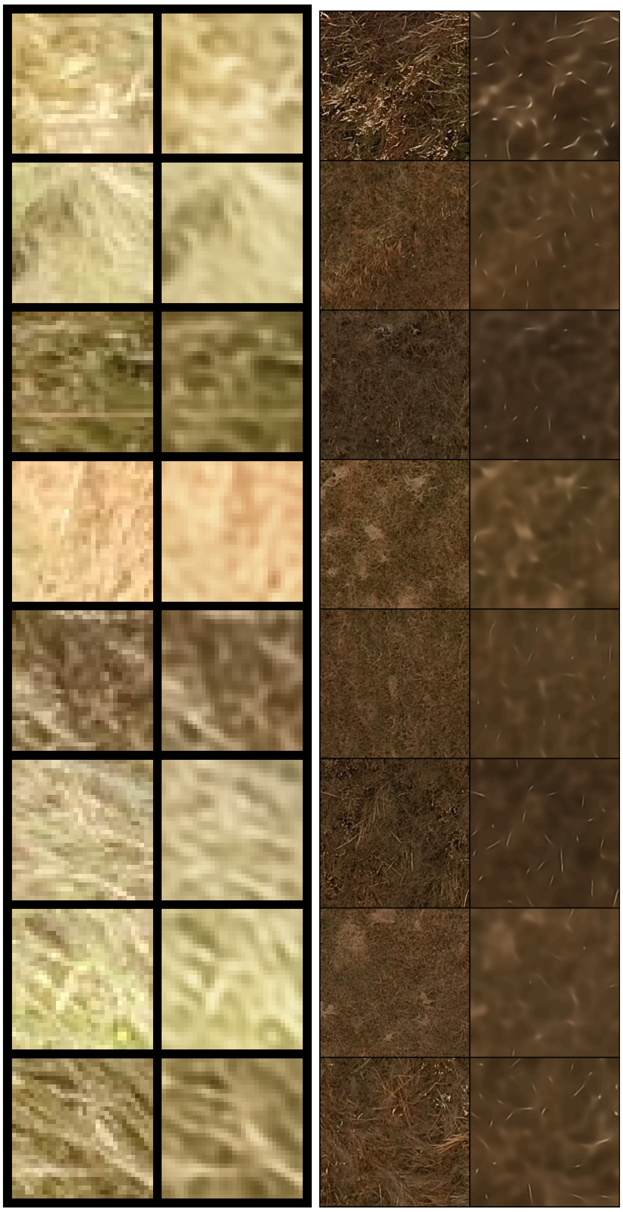
This may be caused by quality of labels or by the fact that vegetative data is not as expressive. It may also indicate a mismatch between testing and training data for the case of vegetative images.

A clear improvement for using self-supervised autoencoders for pretraining cannot be seen, despite always contributing to the top 3 and reducing training epochs. This may also be caused by the fact that the training dataset for autoencoders include images from vegetative ALG and cameras that were not used to generate orthomosaics.

Another reason may be that Autoencoders usually attempt to reproduce entire images, e.g. the loss is not adopted for fine-grained textures, rather overall colour matching, as seen in the following figure. It can be suspected that the compression into the bottleneck layer z removes fine-grained edges and features, which may hold the discriminative power to identify ALG in the wild. The combination of fine-grained plant strands with different colours for different weeds may be key to identifying ALG.

Autoencoder images

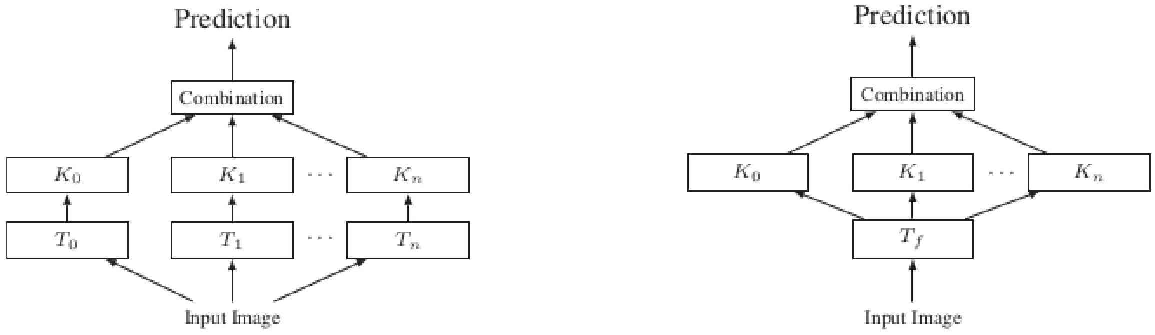
The images show the original input image on the left and the reproduction from the autoencoder on the right. The left image shows the 32 x 32 patches that have been condensed into a 512-dimensional latent space, while the right shows the 256 x 256 images.



Sub-Ensembles

Sub-Ensembles can be used as an approximation to Deep Ensembles, where multiple models are combined and trained on different subsets of the data, to make more robust and generalizable predictions and also allow the opportunity to estimate how certain the model is about its prediction [6], [7].
When all predictors are in agreement, it is less probable that the prediction is wrong and vice versa, when a large factor of disagreement is present, it can be assumed that the data is not represented well.
This can also be used to estimate when an incoming sample is out-of-distribution (OOD) e.g. when a sample is not represented at all in the dataset, which can be very helpful to identify different growth stages, environments or other things that cause the sample to be considered “not part of known data”.

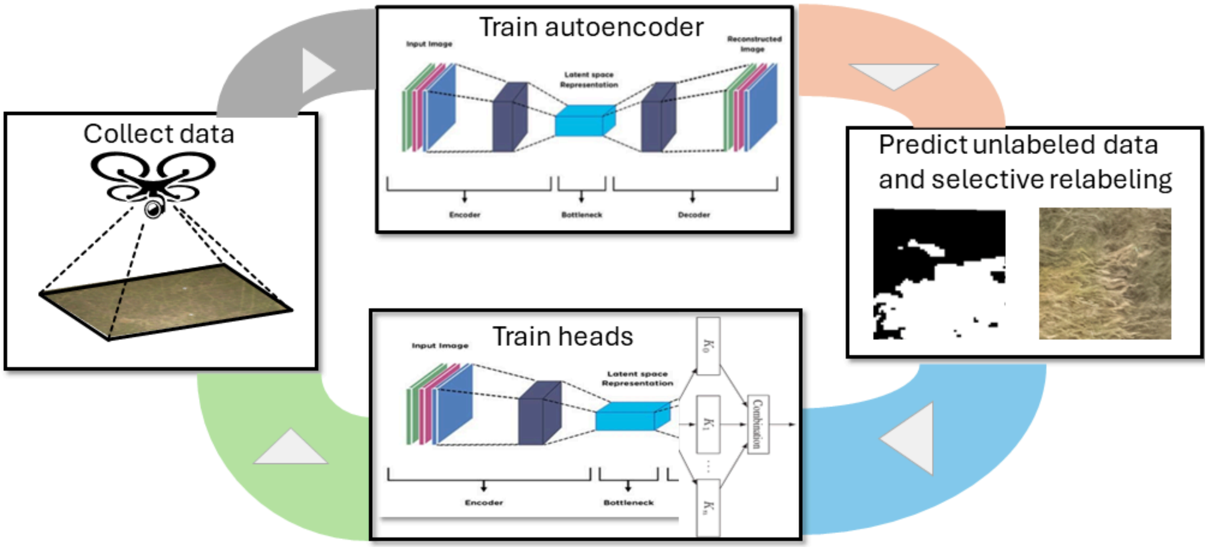
Instead of training entire models, which are computationally heavy, Sub-Ensembles approximate this behaviour by only training different end-parts for prediction.



(a) Deep Ensembles

(b) Deep Sub-Ensembles

The figure shows the inherent difference. where T denotes a trunk network, a backbone, and K the predictors. The structure is computationally much more efficient, because the same trunk network is used and requires less training and has fewer parameters, so less prone to overfitting and it can be trained and run on consumer-grade hardware.
Furthermore, the separation of trunk and classifier allows for different combinations of both to be experimented with, making it an ideal candidate for experiments with autoencoders as dense embeddings of images.



The figure shows the cycle how the data collection cycle works. New data is collected and the raw data used to train an autoencoder. The autoencoder is used as a backbone to train multiple heads with previously labeled data. The resulting model is then used to predict samples and decide which ones should be added to the labeled training dataset. After the new samples have been added to the labeled dataset, the cycle is repeated.
The process of repeating a cycle multiple times over to show samples selected by a degree of difficulty is commonly referred to as “curriculum learning” and has been shown effective in multiple settings where dataset or model discrepancy is common [8].

Methods

This is tested with the raw datasets that have been used to generate orthomosaics, as well as the labeled samples. At first, 100 random samples from Site 1 are added to the training dataset. An autoencoder is trained on raw images from Site 2, and then 10 heads are trained on the basis of that autoencoder backbone with the labels from site 1. The model predictions for all samples from site 2 are used to evaluate the accuracy and 100 labels with the highest cross-entropy H are added to the labeled dataset. This cycle is repeated for the next sites.

$$H(x) = -\Sigma\, p_c(y_c\,|\,x) \cdot log(p\,(y_c|\,x))$$

Orthomosaic Raw Datasets

Sites	Camera	GSD	Image Resolution	Image Size in MB	Image Count
Site 1 - McD	Fuji GFX100	0,22 cm	4000 x 3000	6	249
Site 2 - GC	DJI P1 40m	0,48 cm	8192 x 5460	16	231
Site 3 - Kuma	Fuji GFX100	0,22 cm	11648 x 8736	70	165
Site 4 - TSR	Fuji GFX100	0,24 cm	11648 x 8736	70	285

The table shows the images and cameras used to generate the orthomosaics and also their file sizes. The large size of raw images is prohibitive for runtime data loading, therefore 2000

256 x 256 patches are used as input to autoencoder training. From these, random 32 x 32 crops are taken for predictions.

Variations and Ablations

Multiple variations and ablations are run to assess the impact of certain features:

- A baseline variation that trains normally on the labeled samples and has a single prediction head - denoted “single” in the **heads** -column
- A baseline variation that trains on the entire dataset labeled by MS imagery, denoted “full” in the **selection**-column
- Another baseline is a variant that does not sample by cross-entropy H, but takes 100 random labeled samples to train. Denoted “100 random2 in the **selection** column
- Another variant takes a denoising autoencoder as a backbone, including prediction of hidden image patches [9] - denoted with ae denoise in the backbone column.
- A variant that only trains an autoencoder without denoising, denoted by ae in the **backbone** column
- A variant that does not train any encoder at all but only uses labeled samples. - denoted “none” in the **backbone** column
- Another variant does not fix the backbone for the training of the first head, allowing for modifications between the unlabeled raw images and outputs generated by orthomosaics. Denoted by “retrain previous” in the **resnet** column

Results

		Base	**Growth state**	**Flowering**	**Vegetative**	**Vegetative**
		68.87	**Class Percentage**	74.55%	11.87%	50.51%
			Site	**2**	**3**	**4**
resnet	**selection**	**backbone**	**heads**	**GC**	**Kuma**	**TSR**
18	full	ae	single	57.86%	34.39%	**80.61%**
		ae denoise	single	47.96%	33.87%	49.53%
		none	single	69.95%	28.49%	72.85%
18	100 random	ae	single	52.06%	16.63%	51.38%
		ae denoise	single	26.83%	78.69%	49.53%
		none	single	40.03%		
		ae	subensemble 10	74.55%	9.52%	50.51%
		ae denoise	subensemble 10	74.62%	11.96%	72.16%
		none	subensemble 10	66.38%	11.96%	73.04%
	100 entropy-based	ae	subensemble 10	74.55%	37.10%	49.49%
		ae denoise	subensemble 10	74.61%	**90.55%**	49.49%
		none	subensemble 10	74.74%	81.89%	56.33%

retrain previous	100 random	ae	subensemble 10	74.57%	11.87%	51.21%
		ae denoise	subensemble 10	75.35%	12.26%	49.98%
	100 entropy-based	ae	subensemble 10	74.59%	11.90%	50.51%
		ae denoise	subensemble 10	**76.45%**	33.44%	50.49%

The table shows inconclusive results. In some instances, autoencoder, denoising autoencoders or Sub-Ensembles improve the results significantly, in others not.

The results indicate that there appear to be inconsistencies in the training and further research is necessary.

One possibility is that the order of sites matters. The predictions for Site 3 come from training from Site 4, therefore inverting the order may have an impact on the quality and further research is necessary.

Conclusions

Detecting ALG in the wild is a difficult task. Since it is a grass, there are many false positives and detecting boundaries from other grass species is difficult. However, the detection of single strands is not a requirement but only the presence, since it usually presents in patches. Identifying these is possible in a flowering state, but much more difficult in a vegetative state. Complex approaches to modelling uncertainty and using self-supervised pre training have not yielded conclusive results to improving the quality and further work is necessary.

Future work can focus on integrating the information from MS imagery into the RGB prediction pipeline, as well as improving the capabilities to detect vegetative ALG, as this performance appears poor overall.

Bibliography

[1] P. Keerthinathan *et al.*, “African Lovegrass Segmentation with Artificial Intelligence Using UAS-Based Multispectral and Hyperspectral Imagery,” *Remote Sensing*, vol. 16, no. 13, Art. no. 13, Jan. 2024, doi: 10.3390/rs16132363.

[2] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778. Accessed: Jul. 09, 2024. [Online]. Available: https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html

[3] T. Zhang *et al.*, “Consecutive Pre-Training: A Knowledge Transfer Learning Strategy with Relevant Unlabeled Data for Remote Sensing Domain,” *Remote Sensing*, vol. 14, no. 22, Art. no. 22, Jan. 2022, doi: 10.3390/rs14225675.

[4] C. S. Wickramasinghe, D. L. Marino, and M. Manic, “ResNet Autoencoders for Unsupervised Feature Learning From High-Dimensional Data: Deep Models Resistant to Performance Degradation,” *IEEE Access*, vol. 9, pp. 40511–40520, 2021, doi: 10.1109/ACCESS.2021.3064819.

[5] R. S. Geiger *et al.*, “‘Garbage in, garbage out’ revisited: What do machine learning application papers report about human-labeled training data?,” *Quantitative Science Studies*, vol. 2, no. 3, pp. 795–827, Nov. 2021, doi: 10.1162/qss_a_00144.

[6] M. Valdenegro-Toro, “Sub-Ensembles for Fast Uncertainty Estimation in Neural Networks,” presented at the Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 4119–4127. Accessed: Jul. 08, 2024. [Online]. Available: https://openaccess.thecvf.com/content/ICCV2023W/LXCV/html/Valdenegro-Toro_Sub-Ensembles_for_Fast_Uncertainty_Estimation_in_Neural_Networks_ICCVW_2023_paper.html

[7] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles,” Nov. 03, 2017, *arXiv*: arXiv:1612.01474. doi: 10.48550/arXiv.1612.01474.

[8] P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe, “Curriculum Learning: A Survey,” *Int J Comput Vis*, vol. 130, no. 6, pp. 1526–1565, Jun. 2022, doi: 10.1007/s11263-022-01611-x.

[9] J. Li, P. Chen, S. Yu, Z. He, S. Liu, and J. Jia, “Rethinking Out-of-distribution (OOD) Detection: Masked Image Modeling is All You Need,” Apr. 11, 2023, *arXiv*: arXiv:2302.02615. doi: 10.48550/arXiv.2302.02615.