A Faster R-CNN Approach for Bird Species Classification

Aiswarya Kattappilly Ravi  
*Computer Science dept.   
Liverpool John Moores University*Liverpool, United Kingdom  
csmakatt@ljmu.ac.uk

Abstract: Presently, birdwatching is a popular activity, however in order to identify the species, one requires the assistance of a bird book. I created a deep learning platform to aid users in identifying bird species using a programmed based on the idea of picture recognition to give birdwatchers a helpful tool to appreciate the beauty of birds. By contrasting the model with a trained model, it would also identify the input image and then anticipate the type of bird. As an output, the specifics would be provided. The ability to study ecological processes at larger scales holds considerable potential thanks to developments in artificial intelligence for computer vision. Ecology revolves around the distribution and behavior of individuals, and deep neural networks can be used to distinguish individual items in pictures through computer vision. But creating supervised models for ecological monitoring is difficult because it needs a lot of training data that has been labelled by humans, it needs sophisticated technological know-how and computational infrastructure, and it is prone to overfitting.

The paper focuses on the classification of bird species using the Faster R-CNN and SSD (Single Shot MultiBox Detector) models, which are popular object detection architectures. The objective is to accurately identify and classify various bird species from input images.

The study begins by preprocessing a large dataset of bird images, including tagging for different bird species. This dataset is then split into training and testing subsets for model development and evaluation. The Faster R-CNN and SSD architectures are employed for training the models, leveraging their ability to detect and classify objects within images. During training, the models are fine-tuned using transfer learning techniques, initializing the network weights with pre-trained models such as ResNet-50 or ResNet-101. This approach allows the models to leverage learned features from large-scale datasets like ImageNet, improving their ability to extract relevant features for bird classification.

The trained models are evaluated using the testing dataset, measuring their accuracy, precision, recall, and F1-score for bird classification. The results demonstrate the effectiveness of both Faster R-CNN and SSD in accurately detecting and classifying bird species. Overall, the study showcases the potential of Faster R-CNN and SSD models for bird classification tasks. The results highlight their ability to detect and classify different bird species accurately, providing valuable insights for bird monitoring, conservation, and ecological research.

Keywords—Faster R-CNN,SSD

# Introduction

BC Autumn Watch has launched a computer vision challenge with the aim of developing a deep learning model for object identification. The challenge revolves around tracking the diverse bird species that visit gardens during the autumn season. To capture bird images, a common camera trap is utilized, which captures still photographs at a resolution of 1024 x 768 pixels. However, labelling the bird species in these images poses a significant challenge due to the inherent difficulty in distinguishing between different species. Even experienced bird watchers often struggle to accurately identify birds when presented with a photograph. This challenge pushes the boundaries of both human and computer visual processing capabilities. While bird species share common elements, their shapes and appearances can vary greatly, leading to substantial intraclass variance. Factors such as lighting conditions, background variations, and different bird attitudes contribute to the complexity of the task. Addressing these challenges requires advanced computer vision techniques and deep learning models capable of robustly capturing the subtle differences and nuances in bird species' visual characteristics.

The identification and classification of bird species are traditionally carried out manually by ornithologists, who are experts in the field of birds. However, this approach is time-consuming as it requires in-depth knowledge of bird biology, species characteristics, songs, distribution patterns, and ecological impacts [1] in order to accurately identify the species. The task of species identification poses a significant challenge even for experts in the field. In the domain of image processing, one of the main challenges is to extract the main object, which is the bird, from the image while excluding unwanted objects such as background, trees, and other elements. This segmentation problem adds complexity to the task of bird species identification using visual images.

Several studies have focused on bird classification and identification using visual bird images and bio acoustical data [2]–[4]. These studies aim to predict the bird species by leveraging both visual characteristics and bioacoustics signals. Many techniques have been proposed that utilize bioacoustics sounds, songs, and signals to accurately identify bird species. However, the research in this area is still ongoing, and there are several ongoing studies and advancements in bird classification techniques. The challenge lies in effectively combining visual and bioacoustics information to improve the accuracy and reliability of bird species identification. By exploring and integrating both modalities, it is possible to enhance the understanding of bird diversity and behaviour, leading to more comprehensive and accurate bird classification models.

Object detection algorithms use computer vision techniques to predict objects in images or videos and locate them with bounding boxes [5]. Deep learning, particularly with convolutional neural networks (CNN) [6], has improved the accuracy and meaningfulness of these predictions.

Despite the promise of deep learning approaches, there are some limitations that need to be addressed. One such limitation is the requirement for a large amount of labeled training data, which can be challenging to obtain, especially for specific bird species. Another limitation is the interpretability of deep learning models, as they often function as black boxes, making it difficult to understand the underlying reasoning for their predictions. Additionally, deep learning models often have significant computational requirements and training time.

To address these challenges, the proposed approach in this research is to utilize the Faster R-CNN and SSD models for bird classification. These object detection models have demonstrated success in detecting and classifying objects within images. The rest of this paper is organized as follows:

Section II cover the challenge presented by BBC Autumn watch, which involves computer vision and the role of deep learning in object detection.

Section III describes the methodology used to classify the birds.

Section IV include discussion about the model’s performance.

Section V contains details about the practical implementation.

# background

The challenge presented by BBC Autumnwatch involves the application of computer vision and deep learning techniques in the field of object detection. The goal of BBC Autumnwatch is to track and identify the various bird species that visit gardens during the fall season. To accomplish this, they plan to utilize a common camera trap that can capture still photographs at a resolution of 1024 x 768 pixels.

Computer vision is a multidisciplinary field that focuses on enabling computers to interpret and understand visual information from images or videos. It involves developing algorithms and techniques that mimic human vision capabilities, allowing computers to analyse, process, and extract meaningful insights from visual data. The goal of computer vision is to enable machines to perceive and comprehend the visual world in a manner similar to humans. This involves tasks such as object detection, image classification, image segmentation, tracking, scene understanding, and more. By leveraging computer vision, machines can interpret and make sense of visual data, enabling them to perform a wide range of applications and tasks [7].

Computer vision algorithms often employ various techniques such as image processing, pattern recognition, machine learning, and deep learning. Image processing techniques involve manipulating and enhancing images to improve their quality or extract relevant features. Pattern recognition techniques enable machines to recognize and classify objects or patterns within images. Machine learning algorithms are utilized to train models on large datasets, enabling them to learn and make predictions based on visual input. Deep learning, a subset of machine learning, has revolutionized computer vision by leveraging artificial neural networks to automatically learn hierarchical representations of visual data, enabling more accurate and robust analysis and understanding.

In recent years, computer vision has experienced remarkable advancements driven by the rapid progress in hardware capabilities, the availability of extensive annotated datasets, and the development of sophisticated algorithms. These factors have contributed to significant breakthroughs in the field, allowing machines to push the boundaries of visual perception, understanding, and analysis. The continuous evolution of computer vision has opened up new possibilities and empowered machines to extract meaningful insights from visual data, enabling them to perform complex tasks with improved accuracy and efficiency. This progress has not only transformed various industries but also has the potential to revolutionize how we interact with and benefit from technology in our daily lives [8]-[10].

Deep learning, a branch of machine learning, has transformed the field of object detection by leveraging artificial neural networks to recognize and comprehend intricate patterns in data. In the realm of object detection, deep learning algorithms play a crucial role in automating the process of identifying and localizing objects within images [11].

One of the key components in deep learning for object detection is the utilization of convolutional neural networks (CNNs). CNNs are adept at extracting pertinent features from images, allowing the algorithms to discern and classify objects present in the visual data. This is achieved by applying convolutional filters to capture patterns at different spatial scales, followed by pooling layers that down sample the extracted features [12].

There are two primary approaches to deep learning-based object detection: one-stage detection and two-stage detection. One-stage detectors, such as You Only Look Once (YOLO)[13] and Single Shot Detector (SSD) [14], are renowned for their high inference speed. These algorithms do not require a separate region proposal stage, which makes them faster but may sacrifice some accuracy.

Faster R-CNN, for instance, combines a region proposal network (RPN) with a CNN-based classifier to generate precise bounding box predictions. YOLO and SSD, on the other hand, employ a single-stage detection approach, where the models directly predict object classes and bounding box coordinates in a single pass over the image. This design choice ensures real-time performance and is particularly beneficial in applications where speed is crucial [15].

The power of deep learning in object detection has paved the way for numerous real-world applications. Self-driving cars rely on object detection to perceive and react to the surrounding environment, ensuring safe navigation. Facial recognition systems employ object detection to identify and track faces in images or video streams. Video surveillance systems utilize deep learning-based object detection to detect and monitor objects or individuals of interest in real-time [16].

By harnessing the capabilities of deep learning algorithms, object detection has become more accurate, efficient, and versatile. These models can handle complex and dynamic environments, making our lives easier and safer in various domains, including transportation, security, and computer vision research.

. In the context of the BBC Autumnwatch challenge, deep learning-based object detection methods can play a crucial role in automatically identifying and classifying bird species based on the captured images. By leveraging computer vision and deep learning techniques, it is possible to overcome the challenges associated with manually identifying bird species, which can be time-consuming and prone to errors due to the ambiguity in distinguishing different species.

The utilization of deep learning algorithms in object detection allows for more accurate and meaningful predictions, enabling the automatic identification of bird species based on visual cues. By leveraging the power of deep learning models, it becomes possible to analyse images, locate instances of birds, and provide precise bounding box annotations, facilitating bird species tracking and monitoring during the fall season.

Both Faster R-CNN and SSD have their advantages. Faster R-CNN adopts a two-stage approach, first proposing potential regions of interest and then classifying and refining those regions. On the other hand, SSD is a one-stage detector that performs detection and classification simultaneously, resulting in faster inference times. By applying Faster R-CNN or SSD to the task of bird species classification, ornithologists and researchers can benefit from automated and accurate identification of bird species. These algorithms can assist in processing large amounts of visual data efficiently, enabling faster analysis and facilitating biodiversity research, conservation efforts, and ecological studies related to birds.

# Methodology

The Exploratory Data Analysis (EDA) phase of this research involves analysing a dataset consisting of 2400 images. The dataset is divided into three classes of bird species, namely TurdusMerula, ErithacusRubecula, and PeriparusAter, with each class containing 800 images. Through EDA, I aim to gain insights into the characteristics and distribution of the images, examine any potential patterns or variations among the classes, and explore the overall composition of the dataset. This analysis will provide a solid foundation for further investigation and model development in my study on bird species identification using deep learning techniques. Fig1. Shows some images from the dataset.



Fig. 1 Images from bird dataset

After tagging the images, some of the steps performed were cleaning of the data, visualising the resolution of the images, and removing the bad images from the directory.

### Preprocesing

#### Analyse resolution distribution

To analyse the resolution of images, the dimensions are visualized using joint plots (Fig 2). Most images in the dataset exhibit resolutions that fall within the range of 1000 pixels on both the x and y axes. Considering this observation, it would be prudent to select a pre-trained model that aligns with the prevalent resolution. Notably, the TensorFlow Object Detection API offers a diverse range of models that have been trained on various resolutions, including models specifically trained on 1024 x 1024 images. By leveraging a pre-trained model trained on a similar resolution, can potentially enhance the model's performance and facilitate better feature extraction for the given dataset.

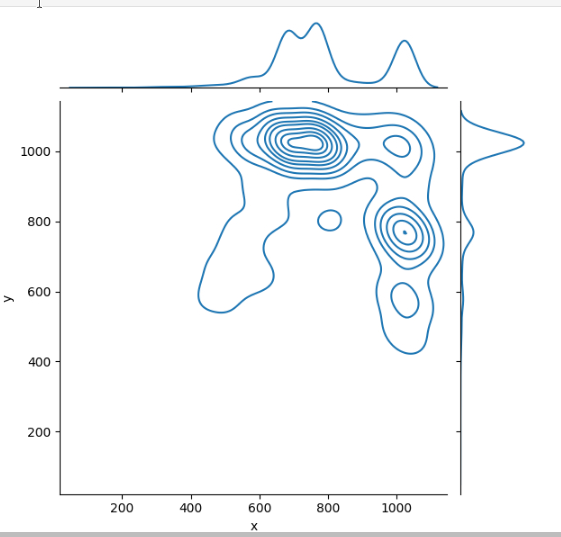
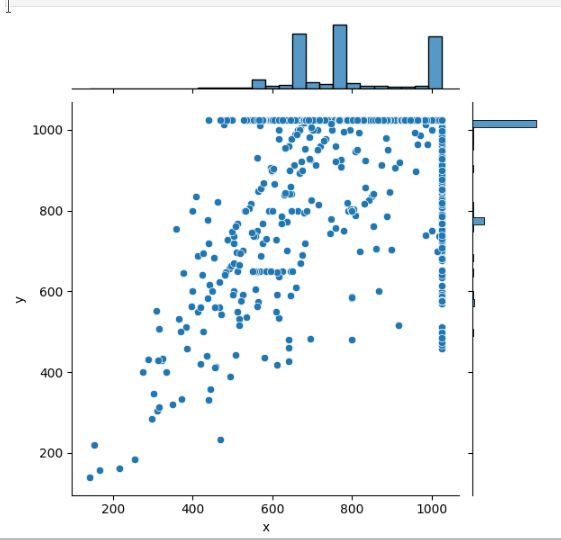


Fig 2 Resolution of the images are plotted on graph

#### Xml to csv

xml\_to\_csv function plays a crucial role in extracting valuable information from XML files and converting it into a structured data frame. By iterating through the XML files within a designated directory, the function retrieves key attributes such as the filename, dimensions (width and height), class labels, and bounding box coordinates for object detection. These extracted values are then organized into a data frame named xml\_df(Fig. 2), which adopts a tabular format with appropriate column names. This data frame serves as a valuable resource for further analysis, allowing researchers to gain insights into the dataset's characteristics, distributions, and potential patterns related to the bird species of interest. By incorporating the xml\_to\_csv function into the EDA workflow, a comprehensive understanding of the dataset's composition and properties can be established.

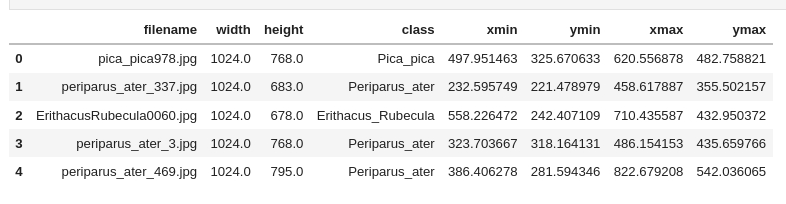


Fig. 3 Features extracted from the xml file

#### Remove unwanted images

While tagging, there may be some images with no object or without a clear image of objects. So, I named them as Bad\_image and have to remove these images from dataset. To check that bad images, I printed unique values of class in data frame (Fig 4) and listed out all entries from the dataframe (Fig. 5).



Fig. 4 Unique class values

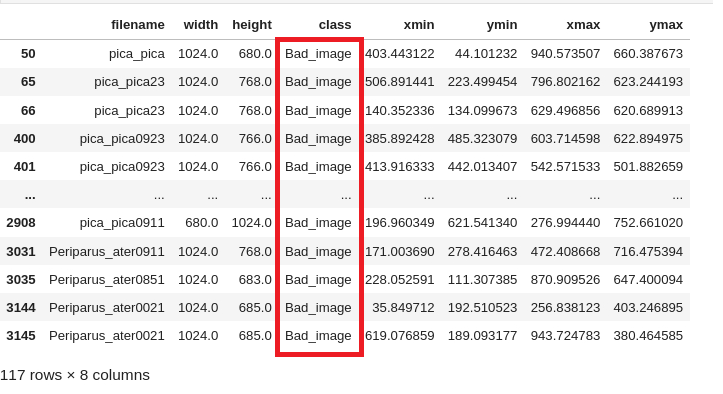


Fig. 5 Entries with Bad\_image value

To clean the dataset, I removed all xml and image files which belongs to Bad\_image. Now, my dataset is ready for training.

### Training

There are several steps include in the training process of bird dataset:

#### Clean the XML

To clean the XML files, a new directory called "cleaned" will be created within the "images" directory. All the XML files will be copied to this "cleaned" directory. Each XML file will then be processed individually by removing any empty spaces or unnecessary characters. The cleaned XML files will be saved back to the "cleaned" directory, replacing the original files. This cleaning process ensures that the XML files are ready for subsequent steps such as data pre-processing and model training.

#### Split the dataset into test and train

The dataset will be split into training and testing sets to evaluate the performance of the model. A desired ratio, such as 90% for training and 10% for testing, will be defined.

#### Create TF Record

TFRecord files (Fig. 6) are created for both the test and train data in object detection to ensure a consistent and standardized format for the data. They facilitate efficient data loading during training and evaluation, as TFRecord files are optimized for fast retrieval. Additionally, TFRecord files allow for easy application of data augmentation and pre-processing techniques to enhance the training process. By separating the test and train data into separate TFRecord files, unbiased evaluation of the model's performance on unseen data is enabled. Furthermore, TFRecord files simplify data management, as they provide a compact and portable way to store, transfer, and share large datasets. Overall, TFRecord files play a crucial role in streamlining the object detection workflow, improving training efficiency, and enabling fair evaluation of model performance.

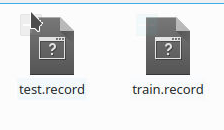


Fig. 6 TFRecord of test and train data

#### Model Development

To classify the bird species accurately, Here I am using 2 models: Faster RCNN and SSD.

1. Faster R-CNN

Faster R-CNN is a leading object detection algorithm that combines deep convolutional neural networks with a region proposal network. It was introduced in 2015 and has since become a popular and effective approach for detecting objects in images. Faster R-CNN employs a two-stage architecture where the region proposal network generates potential object bounding box proposals, and the object detection network performs accurate classification and regression on these proposals. By sharing convolutional features, Faster R-CNN achieves high detection accuracy while maintaining fast inference times. It has been widely applied in various object detection tasks and has greatly advanced the field of deep learning-based object detection.

1. Architecture

* The model consists of two main stages: the first stage performs region proposal generation using a grid anchor generator, while the second stage refines the proposals and performs object classification and bounding box regression.
* The image resizer ensures that input images are resized to a fixed shape of 1024x1024 pixels. The feature extractor utilizes the faster\_rcnn\_resnet101\_keras architecture, which includes a batch normalization layer that is trainable during fine-tuning.
* The model uses a two-stage approach for prediction. In the first stage, anchor boxes are generated at different scales and aspect ratios, and non-maximum suppression is applied based on the score and intersection-over-union (IoU) thresholds. The second stage involves a mask-RCNN box predictor for further refinement of the proposals, with options for dropout and fully connected (FC) hyperparameters.

1. Hyperparameters and Configuration of the config file.

The TensorFlow Object Detection API relies on a configuration file to specify the hyperparameters for object detection task.

* num\_classes: This parameter determines the number of classes the model will be trained on. Since I am training the model on 3 classes, I will set this value to 3.
* batch\_size: This parameter controls the number of samples that the network processes in each training iteration. As my images are high resolution and Faster R-CNN requires significant computational power, I will keep the batch size at 1.
* num\_steps: This parameter defines the number of training steps for the model. I will initiate the training process with 15,000 steps. During training, I can monitor the loss using TensorBoard and analyze its behavior to determine if the model requires further training or convergence.
* fine\_tune\_checkpoint: This field specifies the path to the checkpoint file of the pretrained model. It is important to provide the correct path to ensure the model is initialized with the appropriate weights.
* fine\_tune\_checkpoint\_type: This parameter should be set to "detection" as we are fine-tuning a detection model.
* label\_map\_path: This field indicates the path to the label\_map.pbtxt file, which contains the mapping of class names to their corresponding numeric labels.
* input\_path: This parameter represents the path to the training TFRecord file. TFRecord files are used to efficiently store and read training data. By modifying these fields in the configuration file, I can customize the model training process according to the requirements of object detection task.

1. SSD

SSD (Single Shot MultiBox Detector) is a highly efficient and accurate object detection algorithm introduced in 2016. Unlike traditional two-stage detection methods, SSD takes a single-shot approach, enabling it to predict object class labels and bounding box coordinates simultaneously in real-time. By utilizing multiple feature maps with different resolutions, SSD can effectively capture objects of various scales and aspect ratios. This allows for efficient and accurate detection without the need for time-consuming region proposal generation. SSD has gained popularity for its ability to achieve high detection accuracy while maintaining fast inference speeds, making it well-suited for applications that require real-time object detection, such as video surveillance and robotics.

1. Architecture

* This model is specifically designed for object detection tasks and has been trained on the COCO dataset, which consists of 90 different object classes.
* The input images are resized to a fixed shape of 640x640 pixels using a fixed\_shape\_resizer.
* The model utilizes the "ssd\_resnet50\_v1\_fpn\_keras" feature extractor, which is a variant of the ResNet-50 network architecture with a feature pyramid network (FPN) added for multi-scale feature representation.
* The box\_coder employs the Faster R-CNN style box encoding, and the matcher utilizes the argmax\_matcher for matching predicted boxes with ground truth boxes based on their intersection over union (IoU) overlap.
* The box\_predictor includes convolutional layers for predicting class scores and bounding box regression values. Anchors of different scales and aspect ratios are generated using a multiscale\_anchor\_generator.
* The post\_processing step applies batch non-maximum suppression to filter out redundant detections based on a score threshold and an IoU threshold.
* The model employs weighted\_smooth\_l1 for localization loss and weighted\_sigmoid\_focal for classification loss.

1. Hyperparameters and Configuration of the config file.

* Batch Size: The batch size will be increased from 1 to 2. Since SSD models are computationally less demanding, this system's 25 GB GPU memory can accommodate a larger batch size, which can potentially improve training efficiency.
* Fine-Tune Checkpoint: The path to the fine-tune checkpoint will be updated to the pretrained weights specific to the SSD model being used. This will ensure that the model starts from a pre-trained state and benefits from transfer learning.
* Number of Steps: The training will start with 10,000 steps. During the training process, the loss will be monitored using TensorBoard to assess the model's progress. Based on the loss analysis, further training steps can be determined if necessary.

1. Inference and Evaluation

In the context of object detection models, inference refers to the process of using a trained model to make predictions on new, unseen images. During inference, the model takes an input image and generates bounding box predictions along with corresponding class labels and confidence scores for the detected objects within the image. This allows us to identify and locate objects of interest in the given image.

Evaluation, on the other hand, is the process of assessing the performance and accuracy of an object detection model. It involves comparing the model's predicted bounding boxes and class labels with the ground truth annotations of the objects in the test dataset. Common evaluation metrics for object detection models include mean Average Precision (mAP), which measures the precision-recall trade-off, as well as metrics like precision, recall, and F1-score.

A picture containing text, number, line, font

Description automatically generatedA screenshot of a graph

Description automatically generated with medium confidence

Fig. 7 Evaluation plot for Faster Rcnn

A screenshot of a computer

Description automatically generated with medium confidenceA screenshot of a graph

Description automatically generated with low confidence

Fig. 7 Evaluation plot for SSD

1. Faster R-CNN:

* Average Precision (AP) is relatively higher compared to SSD, indicating better overall object detection performance.
* AP values for IoU thresholds of 0.50, 0.75, and across all IoU thresholds (0.50:0.95) are reasonably high, ranging from 0.651 to 0.905.
* AP varies across different object sizes, with the highest AP achieved for large objects (0.681).
* Average Recall (AR) values are also relatively high, ranging from 0.696 to 0.731.

1. SSD:

* Average Precision (AP) values for SSD are significantly lower compared to Faster R-CNN, indicating poorer object detection performance.
* AP values are generally low, ranging from 0.018 to 0.233.
* The AP for small objects is particularly low (0.000), suggesting difficulty in detecting small-sized objects.

Average Recall (AR) values for SSD are also lower compared to Faster R-CNN, ranging from 0.042 to 0.386. Based on the evaluation metrics, it appears that Faster R-CNN outperforms SSD in terms of object detection accuracy, with higher AP and AR values across various IoU thresholds and object sizes. Therefore, I would recommend using Faster R-CNN as the preferred object detection model for the given task.

# Discussion

BBC Autumnwatch is a nature program that aims to capture and showcase the beauty of wildlife during the autumn season. The computer vision task set by Autumnwatch could involve various objectives related to wildlife analysis and understanding. Potential tasks may include:

* Wildlife Detection and Classification: Automatically detecting and classifying different wildlife species present in the scenes.
* Behaviour Analysis: Analysing the behaviour of animals captured in the footage to understand their interactions and movements.
* Counting and Population Estimation: Estimating the population and count of specific wildlife species within a given area.
* Habitat Monitoring: Monitoring changes in wildlife habitats using computer vision techniques.
* Scene Understanding: Identifying and understanding the environment and ecological context in which the wildlife is captured.

I choose faster rcnn to resolve the above risk. Because of :

* Accuracy: Faster R-CNN is a widely recognized object detection model known for its accuracy. If the challenge involves detecting and classifying wildlife species accurately, Faster R-CNN can be a suitable choice.
* Flexibility: Different computer vision tasks may require different models. For example, if the challenge involves scene understanding and habitat monitoring, semantic segmentation or instance segmentation models may be more appropriate.
* Real-time Processing: If the challenge requires real-time or near real-time processing of video footage, Faster R-CNN's efficiency in inference could be a valuable advantage.
* Data Availability: The availability of annotated datasets is crucial for training deep learning models. The chosen methodology should align with the availability of relevant training data for the specific task.
* Computational Resources: Training and running complex models like Faster R-CNN may require significant computational resources. If the available resources are limited, simpler models or transfer learning from pre-trained models might be more suitable.
* Ethical Considerations: If the challenge involves wildlife monitoring, ethical considerations must be taken into account to ensure minimal disturbance to the animals.

In conclusion, the choice of methodology for addressing the computer vision task set by BBC Autumnwatch should be based on the specific objectives of the challenge, the available resources, and the nature of the data. Faster RCNN is a better methodology for them.

# Future work

1. Step 1: Tag set of 800 images of Periparus Ater, Erithacus Rubecula and Pica pica. Then keep the xml files in image folder for training.
2. Step 2: Data Pre-processing

#### Analyse the resolution of images

Analyzing the resolution of images is crucial for image classification as it directly impacts the level of detail and clarity in the images. Higher resolution images provide finer details and better visual quality, enabling classification algorithms to distinguish between different classes more accurately. In contrast, low-resolution images lack important details and may lead to misclassification. By carefully considering image resolution, a balance can be struck between capturing sufficient detail for accurate classification and optimizing computational efficiency, ensuring that the classification system performs effectively while utilizing available resources efficiently.

#### Visualize the resolutions

Resolutions will be visualised using graphs and identify the average width and hight of dataset.

#### Remove bad quality Images

During the tagging of images, there may include non- quality images or image without objects. These types of images will tag as Bad\_Image in first step and removed in Data pre-processing step.

### Step 3 : Training

1. Clean the xml files and put into the cleaned folder.
2. Split the dataset into 90:10 as train and test data.#
3. Create TFRecord : Tensorflow Object Detection API requires the input data to be in TFRecord format. The TFRecord format is a simple format for storing a sequence of binary records.Here create TF record for train and test.Data in the TFRecord format can take up less space than the original data.TensorFlow can read data in the TFRecord format with parallel I/O operations.Hence, the efficiency will high.
4. Config the config file and hyper parameters: I will use two models. Faster RCNN and SSD. So have to set configuration file and hyper parameters for them.

Parameters which will set for Faster RCNN are: num\_classes, batch\_size, num\_steps, fine\_tune\_checkpoint, fine\_tune\_checkpoint\_type, label\_map\_path and input\_path

Parameters which will set for SSD are: Batch Size, Fine-Tune Checkpoint and Number of Steps.

1. Train the model by setting the epoch values and save the model for future discussion.
2. Evaluation data will save for future discussion.
3. Step 4: Evaluation

Tensor board will used to analyse the loss and accuracy for each model

1. Step 5: Inference

Using the saved model with highest accuracy, predict the species of birds for testing the model.

# References

1. A. Marini, J. Facon and A.L. Koerich, "Bird species classification based on color features", 2013 IEEE International Conference on Systems Man and Cybernetics, pp. 4336-4341, 2013.
2. U.D. Nadimpalli, R.R. Price, S.G. Hall and P. Bomma, "A comparison of image processing techniques for bird recognition", Biotechnology progress, vol. 22, no. 1, pp. 9-13, 2006.
3. C. Huang, F. Meng, W. Luo and S. Zhu, "Bird Breed Classification And Annotation Using Saliency Based Graphical Model", Journal of Visual Communication and Image Representation, vol. 25, no. 6, pp. 1299-1307, 2014.
4. F. Briggs, R. Raich and X.Z. Fern, "Audio classification of bird species: A statistical manifold approach", Data Mining 2009. ICDM'09. Ninth IEEE International Conference on, pp. 51-60, 2009.
5. 6. S. Singha and B. Aydin, "Automated Drone Detection using YOLOv4", vol. 5, no. 3, pp. 95, September 2021.
6. Tareq Abed MOHAMMED, Saad Albawi and Saad Al-Zawi, "Understanding of a convolutional neural network", IEEE Access, vol. 6, pp. 978-1, 2017.
7. B. Zhang, "Computer vision vs. human vision," 9th IEEE International Conference on Cognitive Informatics (ICCI'10), Beijing, China, 2010, pp. 3-3, doi: 10.1109/COGINF.2010.5599750.
8. C. R. Dyer, "Parallel algorithms and architectures for image analysis and computer vision," Sixth Multidimensional Signal Processing Workshop,, Pacific Grove, CA, USA, 1989, pp. 7-, doi: 10.1109/MDSP.1989.96986.
9. D. Barik and M. Mondal, "Object identification for computer vision using image segmentation," 2010 2nd International Conference on Education Technology and Computer, Shanghai, China, 2010, pp. V2-170-V2-172, doi: 10.1109/ICETC.2010.5529412.
10. Xin Zhang; Shuo Xu, “Research on Image processing Technology and computer vision”, 2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL).
11. M. Shanmuganathan and C. Nalini, "Deep Learning Approaches to Image Classification," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2020, pp. 263-268, doi: 10.1109/ICIRCA48905.2020.9182927.
12. C. Hary and S. Mandala, "Object Detection Analysis Study in Images based on Deep Learning Algorithm," 2022 International Conference on Data Science and Its Applications (ICoDSA), Bandung, Indonesia, 2022, pp. 226-231, doi: 10.1109/ICoDSA55874.2022.9862922.
13. K. Li, "Applications of Deep Learning in Object Detection," 2022 International Conference on Computers, Information Processing and Advanced Education (CIPAE), Ottawa, ON, Canada, 2022, pp. 436-442, doi: 10.1109/CIPAE55637.2022.00097.
14. S. Liu, H. Shi and Z. Guo, "Remote sensing image object detection based on improved SSD," 2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and Applications (CVIDL & ICCEA), Changchun, China, 2022, pp. 421-424, doi: 10.1109/CVIDLICCEA56201.2022.9824392.
15. X. Yin, Y. Yang, H. Xu, W. Li and J. Deng, "Enhanced Faster-RCNN Algorithm for Object Detection in Aerial Images," 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 2020, pp. 2355-2358, doi: 10.1109/ITAIC49862.2020.9339038.
16. Zhang Lei, Zhang Yongsheng and Yu Ying, "Research on data augmentation of aerial image object detection", Journal of Surveying and Mapping Science and Technology, vol. 36, no. 05, pp. 505-510, 2019.
17. Liu Delian, "Research on aerial image object detection metho", Xidian University, 2008.
18. R. Girshick, "Fast R-CNN", Proc. IEEE Int. Conf. Comput. Vis, pp. 1440-1448, 2015.
19. A. Krizhevsky, I. Sutskever and G. Hinton, "ImageNet classification with deep convolutional neural networks", NIPS, 2012.
20. M. Lin, Q. Chen and S. Yan, "Network in network", ICLR, 2014.
21. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", ICLR, 2015.