An explainable deep vision system for animal classification and detection in trail-camera images with automatic post-deployment retraining

Golnaz Moallem*a, 1, Don Pathiragea, Joel Reznicka, James Gallagherb, Hamed Sari-Sarrfa

a Electrical and Computer Engineering Department, Texas Tech University, Lubbock, TX, USA 79409

b Texas Parks and Wildlife Department, Mason, TX 76856

Keywords: Automatic wildlife monitoring; Automatic animal detection; Automatic animal localization; convolutional neural networks (CNN) automatic retraining; convolutional neural networks (CNN); Faster R-CNN; Explainability

Abstract

This paper introduces an automated vision system for animal detection in trail-camera images taken from a field under the administration of the Texas Parks and Wildlife Department. As traditional wildlife counting techniques are intrusive and labor intensive to conduct, trail-camera imaging is a comparatively non-intrusive method for capturing wildlife activity. However, given the large volume of images produced from trail-cameras, manual analysis of the images remains time-consuming and inefficient. We implemented a two-stage deep convolutional neural network pipeline to find animal-containing images in the first stage and then process these images to detect birds in the second stage. The animal classification system classifies animal images with more than 87% sensitivity and 96% specificity. The bird detection system achieves better than 93% sensitivity, 92% specificity, and 68% average Intersection-over-Union rate. The entire pipeline processes an image in less than 0.5 seconds as opposed to an average 30 seconds for a human labeler. We also addressed post-deployment issues related to data drift for the animal classification system as image features vary with seasonal changes. This system utilizes an automatic retraining algorithm to detect data drift and update the system. We introduce a novel technique for detecting drifted images and triggering the retraining procedure. Two statistical experiments are also presented to explain the prediction behavior of the animal classification system. These experiments investigate the cues that steers the system towards a particular decision. Statistical hypothesis testing demonstrates that the presence of an animal in the input image significantly contributes to the system's decisions.

---

1 : Corresponding author.

Email addresses: golnaz.moallem@ttu.edu (G. Moallem), D.Pathirage@ttu.edu (D. Pathirage), joel.reznick@ttu.edu (J. Reznick), james.gallagher@tpwd.texas.gov (J. F. Gallagher), hamed.sari-sarraf@ttu.edu (H. Sari-Sarraf)
1. Introduction

Trail-camera imaging is a non-intrusive method employed in ecological research and conservation to gather large-scale data about wildlife and habitat health [1]. However, the task of manually extracting information from this data is costly, labor intensive, and time-consuming. Moreover, without robust domain expertise, the validity of the produced data is uncertain [2]. Deep neural networks (DNNs) are currently viewed as the state-of-the-art for many computer vision tasks, having made great strides due to advances in computer hardware, network architectures, and the availability of very large datasets to learn from.

In this work, we propose a two-stage deep learning pipeline for the analysis of wildlife imagery in the Texas Parks and Wildlife Department (TPWD) dataset. In the first stage, a DNN classifies the TPWD images into ‘Animal’ and ‘No-Animal’ categories. Then, a second DNN detects and localizes birds in the set of ‘Animal’ images. Furthermore, this system is managed by an automatic retraining algorithm which maintains performance as data drifts over time. We also present statistical experiments to address model explainability, i.e., insights into network predictions and behavior.

The paper has the following contributions:

1. A sophisticated method of sampling and modeling data when the available data is significantly varying but limited.
2. A method of detecting data drift and triggering automatic retraining of models using a structural similarity measurement-based index.
3. A set of statistical experiments for interpreting a network’s decision-making process.

The proposed framework is designed in two stages to efficiently label Animal images for further processing. In this framework, the No-Animal images are filtered out in the first stage and only the Animal images undergo the detection process. Since the DNN classifier performs noticeably faster than the DNN detector (details are reported in the following sections), feeding the Animal-labeled images to the DNN detector considerably reduces the overall analysis time.

Section 2 describes the TPWD dataset. Section 3 elaborates on the training and performance of the classification DNN, i.e. animal classification system. Section 4 introduces an automatic procedure designed for the automatic retraining of the animal classification system. Section 5 presents two statistical experiments explaining the predictions of the animal classification system. The training process and performance of the detection DNN, i.e. bird localization systems, is demonstrated in Section 6. Sections 7 and 8 respectively present discussion and conclusion of the study.

2. Dataset

The TPWD dataset is derived from a project investigating the use of prescribed fire to manage wildlife habitat at small scales. While the Northern Bobwhite Quail was the focal species, it was also important to document changes in habitat use by other species of wildlife, with a particular focus on other species of birds.
Traditional wildlife-count techniques would have been difficult to conduct on numerous locations; therefore, trail cameras were used to study wildlife activities at several sites with solar powered water fountains that attract wildlife to the trail camera focal area. Cameras were set to be a standard distance above the fountain (1.52 m) with the same distance from the camera to the fountain (3.05 m). The first year of the study (2014) generated approximately 700,000 images. These images were manually classified by one individual over the course of about 9 months. Given the large size of the image dataset, the large rate of incoming input images, and the need for recurrent image classification, it was necessary to automate the classification process with high sensitivity and accuracy levels.

This research developed deep neural network (DNN) models for detecting animals, especially birds, in trail-camera TPWD imagery. A significant portion of the work undertaken in this endeavor went into generating useful training and testing datasets from the images provided by TPWD. The images were produced from a set of observation sites which resemble each other in their layout. At each location, a motion-triggered camera placed above the ground was centered and focused on a fountain (artificial watering hole) that attracts animals in the vicinity. Over the course of seven days, the camera continuously monitored the scene for activity, taking images when motion was detected and occasionally, at periodic intervals for diagnostic purposes. Images were recorded for 7-day periods in May, June, and July of each year. For night-time imaging, an infrared (IR) flash was used to illuminate the scene without disturbing the animals. The night-time images are captured by an IR sensitive detector on the camera. Typical examples of day and night images are shown in Figure 1.

Given the significant difference in appearance between day and night images, we trained two separate DNNs to analyze the two sets of images. The day-time and night-time datasets used in training and testing these networks were formed from a validated subset of 23,429 volunteer-labeled images, of which only 1,582 contained animals. Figure 2 shows an example of an annotated image in which the animals are labeled and localized with bounding boxes.
Figure 2: Images containing animals were annotated with bounding boxes and associated with labels denoting the class the animal in each bounding box belongs to, e.g., mammal, bird, reptile, etc. Images without animals were marked as ‘empty’.

3. Animal Classification System

Several other works have employed DNN models for classifying wildlife images from camera-trap projects. We began our work by assessing the results of two such papers by Norouzzadeh et al. [3] and Tabak et al. [4] which outline methods for classification of larger mammals (compared to those in the TPWD images) in images from the SnapShot Serengeti (SS) project [5]. We applied the DNN models produced from these works to analyze images from the TPWD dataset. Despite the shared domain relevance between the datasets and similar classification tasks, the networks performed poorly on a benchmark set of TPWD images — in that nearly all images containing no animals produced false positive predictions.

To expedite the development of a more performant DNN, we also explored methods of leveraging transfer learning from a larger, already annotated dataset having better domain overlap with the TPWD images. For this, we trained models on images from the iWildCam 2018 challenge dataset [6] which tracks animals and geographies that are more comparable to those observed in the TPWD images. Like the SS networks, these models too generated mainly false positive predictions. Observing this pattern, we speculated that the presence of the watering fountain, common to all the TPWD images, may be triggering false positive detections. To verify this, we applied inpainting with Nvidia’s Inpainting DNN [7] (see Figure 3) to remove the watering fountain from images with no animals and observed that the networks began to classify such images as true negatives.
Figure 3: Watering fountain in the images was masked out using Nvidia’s Inpainting DNN [7].

Given the apparent bias with existing DNNs toward background scene information, even in cases with significant domain overlap, it was evident that new models needed to be trained specifically on the TPWD images. This necessitated the laborious endeavor of annotating the TPWD images to generate training and testing datasets.

3.1. Dataset Generation Procedure

Initial experiments with random sampling of the labeled TPWD images to generate balanced datasets of Animal and No-Animal classes resulted in models that were highly sensitive to the background content and day-time shadow patterns, which occur naturally in the scene. Consequently, the developed models were again producing mainly false positive detections. We therefore aimed to develop models which better accounted for the variation in background content and shadow patterns by applying a more appropriate procedure for generating training data.

Furthermore, due to the severe imbalance between Animal and No-Animal examples in the TPWD images (1,582 Animal and 21,847 No-Animal), special emphasis was also placed on ensuring the sampling procedure produced balanced and representative modeling datasets to prevent the DNNs from becoming biased towards background information or a particular class. To amplify the number of animal examples in the training data, Animal images were also flipped horizontally. This method of data augmentation has the added benefit of producing models that are invariant to whether an animal appears in a left or a right profile in the image. For the No-Animal class, an equal number of background images from each observation site was incorporated into the datasets.

As our models expect an input image size of 299 × 299 pixels the original 3264 × 2250 images needed to be resized, but simply resizing these large images can lead to problems, e.g., pixels of very small animals such as birds (the majority of animal examples in day images) will be
decimated or lost after resizing. To address this problem, day-time images for both training and testing were first cropped using a $1500 \times 1500$ window centered over the watering fountain — the region of the image where animal activity is highest as determined from the bounding box labels. However, the cropping resulted in an albeit acceptable 9% loss (where at least 50% of a bounding box lies outside the crop) of useable animal-containing examples from the dataset. This step was nonetheless necessary to preserve pixels pertaining to birds after resizing images to the expected network input shape. The proposed automatic window cropping algorithm is shown in Figure 4. In contrast, as there was little to no bird activity in the night-time images, the same procedure was not necessary for the night-time images.

An additional criterion was used in selecting No-Animal images for the day-time dataset — they needed to be well-representative of the various lighting conditions and shadow patterns that occur at each location. This was accomplished by employing a time-of-capture based sampling of images for each location in the dataset. Animal examples were sampled from a histogram with 15-minute interval bins and the No-Animal examples from a histogram with 3-minute interval bins. Time-of-capture sampling was not used for the night-time datasets as there was minimal variation in the background due to the very consistent illumination provided by the IR flash.

**Figure 4:** The automatic window cropping algorithm

1. **Step 1:** Mask the image using color information and morphological techniques
2. **Step 2:** Find the object with closest centroid (red dot) to the center of the image (blue dot) using image processing technique
3. **Step 3:** Crop a $1500 \times 1500$ window centered on the determined object
3.2. Transfer Learning and Architecture Selection

To speed up the development of models specific to our task and dataset, we applied concepts from transfer learning to existing state-of-the-art network architectures. In contrast to other works mentioned in this paper, we found this step necessary as our datasets were not large enough to train models from the ground up. The architecture for our models was selected by comparing pre-trained network performances on ImageNet [8], a large object classification dataset consisting of over 14 million images for more than 20,000 classes. The assumption is that the convolutional layers of a pre-trained, well-performing network on ImageNet will also be suitable for datasets in our domain as these layers learn features such as edges and textures that are common in all object detection tasks. This reduced the training complexity of our networks to just fine-tuning the fully-connected layers for the new classification task which is favorable in cases where data is limited. The size and parameter count of these networks’ fully connected layers also influenced architecture selection as computational resources required to re-train a model were limited. The main limiting factor here was the amount of VRAM available in our GPU (11GB) to handle the volume of data necessary for re-training our models. Following these considerations, a pre-trained Xception architecture [9] was selected. The Xception architecture achieves greater than 90% Top-5 Accuracy on ImageNet with lower model and computational complexity than other networks, making it a good candidate for modeling our classification task [10].

3.3. Animal Classification Results

We trained two separate Xception networks, one for day-time images and one for night-time images. After determining whether an image is from day-time (high hue value) or night-time (near-zero hue value) by comparing them in the HSL (hue, saturation, lightness) color-space, our models take the input image and output two probabilities for whether the image contains an animal or not. Our best day-time network was modeled on a 70:30 training-testing split of 2,942 images from the three observation sites that saw the most animal activity. The night-time network was similarly trained on 1,726 images from eight observation sites. Our models achieved 93.97% and
97.61% classification accuracy on the benchmark dataset for day-time and night-time images, respectively. Details of each model’s performance is presented in Table 1. Using the time-of-capture sampling strategy to incorporate the variation in shadow patterns and background content of observation sites into the day-time training data alleviated the problem of large false positive detections caused by training on only randomly sampled data. In contrast, the visual uniformity within and between observation sites from the night-time images made the task of training a classification network simpler.

Table 1: Statistical measures of the models’ performance are presented in the form of Sensitivity and Specificity for each observation site. TP, TN, FP, and FN refer to true positive, true negative, false positive, and false negative, respectively.

| Obs. Site | ‘No-Animal’ Class | ‘Animal’ Class | Total |
|-----------|-------------------|----------------|-------|
| Site No. 1 | Site No. 2 | Site No. 3 | Total | Site No. 1 | Site No. 2 | Site No. 3 | Total |
| # of Images | 268 | 262 | 311 | 841 | # of Images | 65 | 59 | 131 | 255 |
| TNs | 261 | 254 | 294 | 809 | TPs | 58 | 54 | 109 | 221 |
| FPs | 7 | 8 | 17 | 32 | FNs | 7 | 5 | 22 | 34 |
| Specificity | 97% | 97% | 95% | 96% | Sensitivity | 89% | 91% | 83% | 87% |

| Obs. Site | TP | TN | FP | FN | Sensitivity | Specificity |
|-----------|----|----|----|----|-------------|-------------|
| Site No. 1 | 20 | 17 | 3 | 0 | 100% | 85% |
| Site No. 2 | 18 | 20 | 0 | 1 | 95% | 100% |
| Site No. 3 | 38 | 39 | 0 | 0 | 100% | 100% |
| Site No. 4 | 9 | 9 | 0 | 0 | 100% | 100% |
| Site No. 5 | 34 | 32 | 2 | 0 | 100% | 94% |
| Site No. 6 | 7 | 6 | 1 | 0 | 100% | 86% |
| Site No. 7 | 5 | 5 | 0 | 0 | 100% | 100% |
| Site No. 8 | 18 | 16 | 0 | 0 | 100% | 100% |
| Overall | 149 | 144 | 6 | 1 | 99% | 96% |

4. Automatic Retraining Procedure

A crucial characteristic of a reliable and robust deep learning system is its ability to generalize and to respond in a stable fashion to drift in the incoming data. Ideally, once deployed, a robust system continuously monitors the incoming data and detects any drift in the data that may lead to performance degradation and if necessary, triggers a retraining procedure.

In the TPWD images, the observation sites can look noticeably different with the passage of time and changes in environmental conditions (Figure 6). In these images, data drift manifests itself as background changes in the observation sites. More specifically, the drifted images contain any component that the model (1) is not trained for, and, (2) can switch the model’s classification decision. Such components hereafter will be referred to as notable background changes. These components are mainly temporally and statistically dependent background objects that are added,
eliminated, or have appearance or location transformation unseen to the model. Examples of such components may be change in the background vegetation state, displacement of existing objects (e.g., big rocks, cardboards, buckets, or watering fountain), and the introduction of new objects into the scene. All such factors can potentially transform the background scene in a significant way and cause the deployed model to produce false positives.

To assess the post-deployment health of the classification system, the Animal Classification System (ACS) was trained on a subset of TPWD images that were captured in July of 2017. The dataset generation and training procedures were done as discussed in section III. This trained system will hereafter be referred to as ACS 2017. We tested ACS 2017 on a collection of randomly selected images captured from all observation sites later in 2019.

![Figure 6: One of the observation sites (site No. 1) in chronological order: (a) July 2017, (b) May 2019, (c) June 2019, and (d) July 2019](image)

The evident deterioration in the performance of the model reported in Table 2 indicate that the incoming images gradually drift as the appearance of the background changes over time. The degradation of the results is more pronounced in the day images, in which the background plays a more prominent role, than in the night images where the background appears as being more uniform.

Table 2: Post-deployment performance of ACS 2017 on the 2019 test set. Deterioration of performance of this model as compared to that reported in Section III is reported in parenthesis.

| Day-time Model | Night-time Model |
|----------------|------------------|
| Sensitivity    | Specificity      | Y-index    | Sensitivity    | Specificity      | Y-index    |
| 81.4% (-5.4%)  | 77.7% (-19.8%)   | 61% (-23%) | 94.8% (-4.2%)  | 95.8% (-0.1%)   | 91% (-4%) |

As these results confirm our assumptions about the background appearance changes, we developed a technique to quantify the background changes. The deployed model has only learned the training images and therefore knows the background and background components from the training images. Therefore, the drifted images may also be referred to those images that manifest notable background changes compared to the training images. One can compare the state of the background in the incoming images to all the background states available in the training set to quantify any notable background changes.

However, the temporally- and statistically-independent components in the background such as animal presence, shadow patterns and vegetation movements do affect the background comparison, even for similar background states. Consequently, a one-to-one comparison of the
background states of the incoming and training images is not practical. To resolve this problem, mean images were introduced which essentially eliminate the temporally- and statistically-independent components among frames. Mean images of observation sites for a specific time interval were calculated by averaging all the cropped images taken from the corresponding observation site during that time interval.

Based on the observations made on over 10,000 images in our dataset, the background did not go through notable changes from one sunrise to sunset. Therefore, the mean of both incoming and training images from the sunrise to the sunset within a day for each observation site were estimated and used as the daily background state of the corresponding site.

The goal was to determine if the model is trained for the background state in the incoming images. If not, trigger the retraining process. The automatic retraining triggering was accomplished through the following steps per observation site:

1. For a day worth of incoming images, estimate the mean image \( \bar{I}_{one\_day} \).

2. Assuming there are \( N \) background states available in the training set of the deployed model, compare \( \bar{I}_{one\_day} \) to these training background states \( \bar{I}_{BG\_state\_1}, \ldots, \bar{I}_{BG\_state\_N} \). Each background state is the mean of the training images captured on a single day.

3. If \( \bar{I}_{one\_day} \) is similar to one of \( \bar{I}_{BG\_state\_1}, \ldots, \bar{I}_{BG\_state\_N} \), then that means the model is trained for the background state in that day and theoretically can perform the classification task adequately for those images. Otherwise, it requires retraining.

The measure of similarity between \( \bar{I}_{one\_day} \) and background states should be properly quantified; we set up a technique employing the structural component of the similarity index (SSIM) [11] as defined by:

\[
s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x \sigma_y + C_3},
\]

where \( x \) and \( y \) are images being compared, \( \sigma_{xy} \) is the cross-correlation of \( x \) and \( y \), \( \sigma_x \) and \( \sigma_y \) are the standard deviation of \( x \) and \( y \), respectively, and \( C_3 \) is the regularization constant. This component contains the structural similarity information defined as the luminance- and contrast-independent characteristics that account for the structure of objects in the field of view [11].

To determine the similarity between two mean images, local SSIM-Structure values were calculated for corresponding sub-regions in the mean images. Because the key background feature, i.e., the watering fountain, occupies a 500 \( \times \) 500 neighborhood in all 1500 \( \times \) 1500 images, local structural calculations were performed within 500 \( \times \) 500 windows with a stride of 250 pixels and the results of this procedure were stored in a 5 \( \times \) 5 SSIM-Structure matrix.

Figure 7 demonstrates two examples of corresponding sub-regions from mean images of observation site No. 3 over different time intervals. The presence of rocks and vegetation in the sub-region from July 2019 shown in Figure 7(b) causes discrepancies in the structure of the scene
that result in a relatively low SSIM-Structure value. On the other hand, sub-regions in Figure 7(c) and 7(d) contain similar structural components and, hence, produce a high SSIM-Structure value.

| Observation site No. 3 |
|------------------------|
| Averaging Time Interval: July 2017 | Averaging Time Interval: July 2019 | SSIM-Structure |
| (a) | (b) | 0.37 |
| (c) | (d) | 0.84 |

*Figure 7: Examples of corresponding sub-regions from two mean images and associated SSIM-Structure values*

We chose the standard deviation of the SSIM-Structure matrix for measuring the dissimilarity of mean images. This measure is referred to as the Retraining Trigger Index (RTI). Figure 8 displays the heatmap of estimated RTI values for several pairs of $\bar{I}_{one\_day}$ images from observation site No. 1. As expected, all the diagonal components are zero, because an image is compared to itself and so all the elements of the SSIM-Structure matrix are ones. Therefore, the standard deviation of this matrix defined as the RTI is zero. Moreover, the RTI values associated with intra-monthly pairs are noticeably smaller than those of inter-monthly pairs since in terms of vegetation growth, month-to-month background changes are more drastic compared to background changes that occur within a month. The low RTI value of Pair 1 validates the visual similarity between the two mean images. The higher RTI of Pair 2 compared to Pair 1 shows that Pair 2 manifests more local dissimilarities as visually evident. However, both these pairs have RTI values less than 0.1 and neither carries a notable background change. On the other hand, Pairs 3 and 4, associated with RTI values above 0.1, exhibit notable background changes, e.g. vegetation state change and displacement of the fountain.

Inspecting 602 mean image pairs visually and monitoring their associated RTIs, we found that RTI values higher than 0.1 indicate a notable background change. Accordingly, the retraining triggering procedure is illustrated in Figure 9. Every time an $\bar{I}_{one\_day}$ triggers retraining, a subset of images associated with that $\bar{I}_{one\_day}$ is formed based on the corresponding temporal histogram sampling (see Section I). This subset is then appended to the model’s training set, and, the model is retrained with the enhanced training set.
Figure 8: Estimated RTI values for pairs of $\bar{I}_{\text{one-day}}$ images in observation site No. 1. Higher values are shown by lighter color as lower values are illustrated by darker colors.

Figure 9: The algorithmic flowchart of the automatic retraining triggering procedure. For the similarity test of $\bar{I}_{\text{one-day}}$ and $\bar{I}_{\text{BG state}_i}$, the RTI value is estimated and thresholded.
To demonstrate how this retraining triggering system works, we deployed the ACS 2017 model accompanied with eight training background states on two sets of one-day images:

1. Figure 10 illustrates the steps of the retraining triggering algorithm for the $I_{\text{one-day}}$ from July 2019. The background state of $I_{\text{one-day}}$ is compared with all the training background states ($I_{\text{BG, state}_1}, \ldots, I_{\text{BG, state}_8}$) available in ACS 2017 and the RTI values are estimated. One may easily observe that the $I_{\text{one-day}}$ has notable background changes through visual inspection; and the estimated RTI values are all greater than 0.1, which are in-line with the visual observation. The algorithm recommends that the model needs retraining. To find out if this recommendation is plausible, we deployed the ACS 2017 model on a subset of the images associated with $I_{\text{one-day}}$, which resulted in a poor 60% specificity and 67% sensitivity. The retrained model achieved specificity and sensitivity of 100% for the same image subset, which further confirms the algorithm’s recommendation.

2. Figure 11 demonstrates the progression of the retraining triggering technique for the $I_{\text{one-day}}$ captured in July 2017. The computed RTI for the first background state ($I_{\text{BG, state}_1}$) is 0.03. The RTI value being less than 0.1, the algorithm suggests that retraining is not needed and the model is trained for the background state of $I_{\text{one-day}}$. To further evaluate the recommendation of the algorithm, the ACS was deployed on the images affiliated to $I_{\text{one-day}}$ that lead to 81% sensitivity and 95% specificity. These results validate the algorithms suggestion.

*Figure 10: Retraining triggering procedure on the single-day images captured in July 2019*
The model does not need retraining!

Figure 11: Retraining triggering procedure for the single-day images taken in July 2017

Based on these experiments and observations, the automatic retraining triggering algorithm employing the introduced RTI is a reliable technique for triggering the ACS when a measurable drift in the incoming images appears.

5. Explainability

The CNNs have demonstrated remarkable success with various image classification tasks [12-15]. As the current study suggests an adequately trained ACS achieves a significant success rate with the animal detection challenge for images of various animal species over several observation sites. However, it is not quite clear as to how a well-trained CNN arrives at a particular decision; specifically, what criteria and features in an input image are captured by the CNN to determine a classification label.

This major shortcoming in the interpretation of a CNN classification mechanism is originates from the black-box nature of such networks. This subject has been recently addressed in several works [16-27]. There have been several visualization tools and libraries developed for explaining deep Neural Networks [19, 21, 22]. Moreover, heatmap visualization approaches have been used in explaining the decisions of the deep neural networks [17, 25-27]. Meanwhile, these methods propose a general explanation for how a trained system works; this section introduces an exclusive interpretation of the CNN classifier in ACS using a frequentist statistical approach. We proposed two statistical experiments to investigate the rationale of the network behind its correct decisions, as follows:

I. True-Positive (TP) experiment, which investigates the motive behind the classifier’s decisions for TP images
II. True-Negative (TN) experiment, which examines the rationale behind the classification of TN images

The following presents a detailed description of these experiments. Both experiments are executed on 1500 × 1500 cropped day-time images.

5.1. True-Positive (TP) Experiment

In the True-Positive experiment, the performance of ACS on TP images is analyzed. TP images are images in the testing set containing an animal in the field of view that are correctly classified into the ‘Animal’ category. For this experiment, we hypothesize that:

Null hypothesis ($H_0$): The ACS significantly bases the classification decision (Animal/No-Animal) on the presence of an animal in the input image.

The alternative hypothesis is, therefore defined as:

Alternative Hypothesis ($H_a$): The ACS bases the classification decision regardless of the presence of an animal in the input image.

The data preparation phase for this experiment is rather cumbersome, yet doable if performed in an organized manner. Table 3 and Table 4 describe such a workflow. Every TP image is paired with a No-Animal image based on the temporal and structural aspects. We simply refer to the paired image as the twin image. The algorithm for finding the twin image is demonstrated in Table 4. The dissimilarity index (DISI), defined in Equation (1), quantifies the degree of temporal and structural similarity. The DISI value for the TP image and a No-Animal test image consists of two terms: (1) the time stamp difference associated with the two images, this signifies the temporal similarity, and (2) the similarity index of the two images as discussed in detail in [11]. Finding the twin image based solely on temporal similarity is not sufficient because not only temporal features such as shadow patterns contribute to the appearance of the observation site, but also other environmental features, e.g. cloud overcast, rain, wind, etc., transform the appearance of the field of view. Such features can be properly quantified by the similarity index.

| Table 3: The algorithm designed for True-Positive Experiment
|---|
| **Experiment I: True-Positive Experiment**
| **Input:** $N_{tp}$ testing images correctly labeled as “Animal”, i.e., TP images
| **for TP images** $i = 1, 2, ..., N_{tp}$ **do**
| 1. Find the twin image using Algorithm I
| 2. Feed the twin image to the ACS and collect the estimated label
| **end**
| **Using** the collected statistics:
| **Establish** the t-test stat, $t = \frac{\bar{x} - \mu}{s/\sqrt{n}}$
| $\bar{x}$: sample means
| $s^2$: sample variance
| $n$: sample size
| $\mu$: specified population means
\( t \): Student-t quantile with \( n-1 \) degrees of freedom

\( p \text{-value} \): corresponding calculated probability, defined as the probability of finding the observed results when the null hypothesis (\( H_0 \)) is true

Reject the null hypothesis for a significance level \( \alpha = 0.05 \) if the calculated \( p \)-value is less than \( \alpha \).

Otherwise, the experiment fails to reject the null hypothesis; this simply means that the data support the null hypothesis. A significant \( p \)-value indicates strong support for the null hypothesis.

**Table 4: The algorithm designed for finding the twin image**

**Algorithm I: Finding the twin image**

**Input:** one TP image & all the No-Animal images from the same observation site

**Define** dissimilarity index (DISI)

\[
\text{DISI} = \frac{1}{60 \text{ sec}} \times (T_{TP \text{ image}}(\text{sec}) - T_{No-\text{Animal \ image}}(\text{sec})) + (1 - \text{SIM}_{TP \text{ image, No-\text{Animal \ image}}})
\]

DISI: dissimilarity index, dimensionless

\( T \): Timestamp associated with an image, second

\( \text{SIM} \): Similarity Index \([11]\), dimensionless

1/60: conversion factor

for No-Animal images \( i = 1, 2, \ldots \) do

1. Calculate the DISI for the \( i^{th} \) No-Animal image and the TP image
2. Record the DISI for the corresponding No-Animal image

end

No-Animal image with minimum DISI \( \rightarrow \) twin image

Figure 12 demonstrates an example of a TP image and its twin image.

![Figure 12: An example of a TP image (Left) and its twin image (Right). As shown, the two images are almost the same with respect to the background components (watering fountain placement, vegetation, rocks, etc.) and the shadow patterns. The only noticeable difference is the bird in the TP image (circled in red).](image-url)
The results of this experiment for all TP images indicate that in at least 93% of the cases, the twin image received a “No-Animal” label (See Table 5). We take the following steps to test the null hypothesis:

1. We assume that an acceptable performance for the ACS on twin images is 0.95. We simply refer to this statistic as the “success rate,” denoted as \( p = 0.95 \). The total number of samples is 221.
2. We may think of this process as a series of \( N_{tp} = 221 \) binomial samples, for which the expected success rate of correct labeling is 0.95. The samples are assumed to be independent.

An investigation of the underlying binomial distribution justifies a Normal approximation to the binomial distribution. The binomial standard deviation, defined as \( \sqrt{N_{tp}p(1-p)} = 10.5 \) is higher than the threshold level of 10 [28]. This observation justifies a safe application of the normal-based t-test.

A one-sided t-test with a 0.05 significance level confirms a 0.95 minimum success rate of correct labeling. The one-sided t-test fails to reject the null hypothesis with a strong p-value of 0.21 and an upper bound confidence value of 0.964. Therefore, the ACS significantly relies on the presence of an animal to pass an ‘Animal’ label. Note that, if the expected success rate is dropped to 0.94, the t-test would still fail to reject the null hypothesis; however, the p-value notably increases to 0.42.

*Table 5: The results of the True-Positive experiment. As demonstrated, for all three observation sites, ACS estimates the ‘No-Animal’ label for the selected twin images at least 93% of the cases.*

| Site       | Total twin images | No. of twin images correctly classified | ACS Specificity on twin images |
|------------|-------------------|----------------------------------------|-------------------------------|
| site No. 1 | 58                | 54                                     | 93%                           |
| site No. 2 | 54                | 51                                     | 94%                           |
| site No. 3 | 109               | 102                                    | 94%                           |
| Total      | 221               | 207                                    | 94%                           |

Consequently, it can be confidently concluded that the ACS significantly emphasizes the presence of an animal in an image to pass an Animal/No-Animal decision.

### 5.2. True-Negative (TN) Experiment

True-Negative (TN) images are ‘No-Animal’ images in the testing set that are correctly labeled. The TN experiment investigates the rationale behind the ACS decision for the TN images. Similar to the TP experiment, we establish a hypothesis testing procedure for the assessment of the ACS decision for the TN images:

Null hypothesis \( (H_0) \): The ACS learns the observation sites’ temporally- and statistically-dependent and independent background components, such as the background objects, shadow patterns, movement of vegetation caused by wind; Subsequently, the presence of an animal is considered as a disturbance to the learned patterns of the observation sites.
Table 6 elaborates on the algorithm for testing the null hypothesis. Again, the data preparation phase does require some attention. To introduce a disturbance in the TN images, a template of an animal is used, for which three examples are illustrated in Figure 13. Templates of two different bird species in various sitting positions are extracted from random observation sites. The bird species are chosen for imposition since birds are the smallest animals in the field of view and, thus, challenging to recognize. The animal visiting location distribution for each observation site is estimated by recording the center of the annotation bounding boxes for all animals. These distributions are demonstrated for three observation sites in Figure 14. The hypothetical birds are introduced to the TN images based on the samples from these spatial distributions.

Table 6: The algorithm designed for True-Negative Experiment

| Experiment II: True-Negative Experimental Procedure |
|-----------------------------------------------------|
| **Input:** \( N_{tn} \) testing images correctly labeled as ‘No-Animal’, i.e., TN images |
| **Construct** visiting location distribution of animals in the observation sites. The center of the annotation bounding box is considered as the visiting location of the corresponding animal. |
| **Extract** three templates of two different bird species in different gestures from the TPWD images |
| **for TN images** \( i = 1, 2, \ldots, N_{tn} \) **do** |
| 1. Introduce the first template to the \( i^{th} \) TN image at a location sampled from the constructed location distribution |
| 2. Feed the new image to the ACS and collect the statistics. |
| 3. Repeat steps 1 and 2 for the second template |
| 4. Repeat steps 1 and 2 for the third template |
| **end** |
| **Using** the collected statistics: |
| **Establish** the t-test stat similar to the previous experiment |
| **Reject** the null hypothesis for a significance level \( \alpha = 0.05 \) if the calculated p-value is less than \( \alpha \). |
| **Otherwise**, the experiment fails to reject the null hypothesis; this simply means that the data support the null hypothesis. A significant p-value indicates strong support for the null hypothesis. |

![Figure 13: The three templates used in the TN experiment](image.png)
For every TN image, the bird template is positioned in a location that is randomly sampled from the observation site’s estimated animal visiting location distribution. This process is repeated for all three bird templates. Examples of disturbed TN images are shown in Figure 15.

The results of the ACS performance on the disturbed TN images are demonstrated in Table 7. The imposition of bird templates alters the classification label in at least 98% of the cases.

Following the proposed workflow in Table 6, we test the null hypothesis:

1. The assumed success rate of the ACS for disturbed images is $p = 0.95$. The total number of samples is 809.
2. We model this process as a series of $N_{tn} = 809$ binomial samples, for which the expected success rate is 0.99. The samples are assumed to be independent.
3. The binomial standard deviation, defined as $\sqrt{N_{tn}p(1-p)} = 38.4$, that is significantly larger than the threshold level of 10 [28]. This observation indicates that the binomial distribution can be approximated by a Normal distribution.

The one-sided t-test fails to reject the null hypothesis with a strong p-value of 1 and an upper bound confidence value of 0.996. Therefore, we conclude that ACS learns the background components and patterns of the observation sites and the variabilities associated with them, and the presence of an animal in fact disturbs the learned patterns of the observation sites. Thus, the classification label is determined based on whether the learned pattern is disturbed.

The p-value associated with the TN experiment is noticeably higher ($p = 1$) than the one for the TP experiment ($p = 0.21$), even though the p-value of the TP experiment is acceptable. The main reason is that we have a significantly larger sample population for the TN experiment. The notable difference of p-values is also due to the more considerable contribution of background images in the ACS training set.

Figure 16: Birds, which make up around 65\% of the total animal population in the TPWD dataset (1,069 birds among 1,592 animals), pose a significant challenge to automatic detection because of their size (first row), camouflage (second row), position and range of activities (third row). All birds are circled in red.
6. Bird Localization

The ACS efficiently classifies images into ‘Animal’ Versus ‘No-Animal’ categories. Birds form more than 65% of the animal population in the TPWD database and are most challenging to localize both manually and automatically due to their relative size, unpredictable position, and camouflage that allows them to blend in with the background (See Figure 1).

To tackle this challenge, a Bird Detection System (BDS) based on the Faster Region-based Convolutional Neural Network [29] was designed to localize the birds in the animal class as labelled by the ACS subsystem.

Although several published works deal with the problem of bird detection [30-37], none were found to address the aforementioned challenges in a satisfactory manner. For example, those described in [30, 35, 37] present approaches for the detection of bird parts (not birds), while others, e.g., [31, 34], focus on detection from aerial images in which the birds have significantly different radiometric and geometric appearances than those in the TPWD images.

Perhaps the most relevant of existing works to that presented here are by Simons et al. [32] and Wang et al. [33]. The authors of [32] present a cascade object detector to detect and count birds in trail camera images. However, they choose not to pursue a deep learning approach and cite the small size of their training dataset as the reason. As will be detailed in the next section, our work successfully employs a deep learning strategy that was trained on a dataset even smaller than that presented in [32].

Wang et al. [33] present a modified YOLO network for bird detection that is trained on the 2012 PASCAL VOC dataset. A close inspection of this dataset revealed that the birds are much more prominent in their respective images than is the case in our dataset. Specifically, while the birds occupy, on average, 18% of the image in the data used in [33], this number is as low as 1% for our images. We, therefore, concluded that the YOLO model presented in [33] could not be used to accurately localize the birds in the TPWD images.

We trained a Faster Region-based Convolutional Neural Network (Faster R-CNN) [29] to detect and localize birds in the positive images (i.e., those labelled as having animals in the ACS module). The trained network receives a preprocessed positive image and localizes the birds by estimating a bounding box per bird. In the preprocessing step, a $1500 \times 1500$ window from the input image is cropped and centered around the watering fountain. The Faster R-CNN model consists of two sub-models. The first sub-model, i.e., proposal network, learns to find region proposals in the input image that are likely to contain a bird. Redundant RPs are eliminated by employing non-maximum suppression based on their proposal scores. The second sub-model that is a detection network ranks the selected RPs by assigning a detection score to every chosen RP. Finally, regions with the highest scores are outputted as bounding boxes containing birds.

To train BDS, we generated a training set containing two subsets of images. The first subset, i.e., a positive subset, includes 80% of the single-bird images in the TPWD database. Each image is paired with a bounding box localizing the bird in the corresponding image. The second subset, i.e., negative subset, is a group of FPs collected using the hard-negative mining method.
[38]. Figure 17 shows examples from each of the two subsets used in training and Table 8(a) reports the details of the training set.

Figure 17: Examples of images in the positive and negative subsets of the training set

The specifications of the training procedure are shown in Table 8(c). The optimizer used is Stochastic Gradient Descent with Momentum of 0.9 and a learning rate of 0.001. For training the proposal network, a binary class label is assigned to each RP. Two kinds of RPs are assigned a positive label: (i) the RP/RPs that have the highest Intersection-over-Union (IoU) with a ground-truth box or (ii) an RP for which exists a ground-truth box with an IoU larger than 0.6. A negative label is allocated to a non-positive RP that has IoU of less than 0.5 with any ground-truth box.

As illustrated in Figure 18, the trained BDS is capable of detecting single and multiple birds with varying size, color, and gesture from all observation sites.
Table 8: Details of (a) Training set, (b) Testing set and (c) Specifications of the training procedure

| (a) Training Set Details | (c) Training Process Specifications |
|--------------------------|-------------------------------------|
| **Total No. of Images**  | **SGDM**                            |
| **1,007**                | **Momentum = 0.9,**                  |
| **No. of Single Bird Images** | **Learning Rate = 0.001**        |
| **855**                  |                                      |
| **No. of Background Samples** |                                      |
| **152**                  |                                      |

| (b) Testing Set Details | |
|------------------------|------------------|
| **No. of Single-Bird Images** | **244**          |
| **No. of Multi-Bird Images** | **164**          |
| **No. of No-Bird Images** | **1,619**        |
| **Total No. of Images**  | **2,027**        |
| **Total No. of Birds**   | **567**          |

To quantitively assess the performance of BDS, a test set is formed that contains the remaining 20% of single bird images along with all the multi-bird images in the TPWD database. Table 8(b) tabulates the details of this test set. TPs, TNs, FPs, and FNs that are used to estimate the sensitivity and specificity of BDS are defined as follows. TPs are number of localized birds for which the IoU of the estimated and ground-truth bounding boxes is greater than 0.4. TNs are the number of no-bird images for which the model does not output an estimated localization. FPs are assessed in two ways: (i) localized birds for which the IoU of estimated and ground-truth bounding boxes is less than 0.4, and (ii) number of background regions localized as a bird. Lastly, FNs are the number of birds not localized.

BDS performed with 93.7% sensitivity and 92.5% specificity on the test set. Details are presented in Table 9.

Table 9: Details of the performance of BDS on the generated testing set

| BDS Performance Statics | Avg. IoU | Avg. Localization Time |
|-------------------------|---------|------------------------|
| **TP**                  | **TN** | **FP** | **FN** | **Sensitivity** | **Specificity** |
| 526                     | 1,534  | 125   | 35     | 93.7%          | 92.5%          |
|                         |       |       |        | 68%            | 0.4 s          |

(Tested on AMD Ryzen 7, 3.6 GHz)
7. Discussion

The models from prior works in the literature performed poorly on the TPWD images ([3] [6]); this indicated that the DNN models that are trained to perform similar tasks may not successfully generalize despite shared domain relevance. Furthermore, we found that a carefully selected dataset was necessary to train a model to handle the variation in lighting conditions and backgrounds of observation sites; this was evident when direct training on the TPWD data via random sampling proved to be insufficient for producing a well-performing model. The seasonal and incidental changes to the scenery of the observation sites, such as vegetation growth, rearrangement of the existing objects, and the addition of new objects into the scene, deteriorated the performance of the animal classification system. A significant portion of the work in this paper was focused on recognizing these drifts in the images, efficiently forming training datasets which are well-representative of these variations in the data and continuously retraining the animal
classification system. On a typical 8-core CPU (AMD Ryzen 7 3700X), the classification and bird detection tasks take approximately 0.05 and 0.4 seconds per image, respectively. An image is processed through the entire pipeline in less than 0.5 seconds while a human labeler may take at least 30 seconds on average to accurately label each image. The automatic pipeline presents a speed-up of the process by up to 60 times.

As for the future work, animal species classification may be added to the proposed pipeline; this task requires gathering more animal containing examples to generate a multiclass dataset. Moreover, we may extend the current system into one multi-stage network, e.g., a network that handles both sorting of animal vs no-animal images and then performing localization and species level classification of animal images. There is also potential for the systems to be included in a semi-automatic labeling procedure where a second opinion from our models can be used assist anyone labeling new images.

8. Conclusion

We presented a pipeline for automatic animal activity detection in trail-camera images taken from fields monitored by the Texas Parks and Wildlife Department. Common wildlife monitoring methods involve intrusive and tedious tasks, and also are labor intensive to conduct. Hence, in order to capture the wildlife activity in a comparatively inexpensive and non-intrusive manner, trail-camera imaging is utilized. A two-stage deep learning pipeline comprising an animal classification system and a bird detection system was implemented. The animal classification system categorizes the images into ‘Animal’ and ‘No-Animal’ classes and then the ‘Animal’ images are processed to detect birds through the bird detection system. The animal classification system achieved a sensitivity and specificity of more than 87% and 96%, respectively. The bird detection system detects the birds with more than 93% sensitivity and 92% specificity with an average IoU of more than 68%. These findings revealed that the proposed pipeline proves useful in fast and accurate classification and detection of animals in TPWD trail-camera images. We addressed the importance of post-deployment updates to the CNN-based animal classification system as the image features alter due to continuous seasonal changes in the wildlife habitat. We equipped the animal classification system with an automatic retraining algorithm which tracks temporal data drifts. We proposed a novel method for inspecting drift in the incoming images and triggering the retraining process based on the drift recognition. Finally, we designed and implemented two statistical experiments to explain predictions of the animal classification system. These experiments explored the image features that influence the system’s decision. The test results statistically supported a significant contribution of animal presence to the animal classification system’s decision.

Acknowledgements

The authors would like to thank the members of the Applied Vision Lab at Texas Tech University for their assistance in image annotation, especially Peter Wharton, Rupa Vani Battula, Shawn Spicer, Farshad Bolouri, Colin Lynch, and Rishab Tewari. This research was funded by a grant from the Texas Parks and Wildlife Department.
References

[1] A. F. O'Connell, J. D. Nichols, and K. U. Karanth, *Camera traps in animal ecology: methods and analyses*. Springer Science & Business Media, 2010.

[2] K. R. Swinnen, J. Reijniers, M. Breno, and H. Leirs, "A novel method to reduce time investment when processing videos from camera trap studies," *PloS one*, vol. 9, no. 6, p. e98881, 2014.

[3] M. S. Norouzzadeh et al., "Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning," *Proceedings of the National Academy of Sciences*, vol. 115, no. 25, pp. E5716-E5725, 2018.

[4] M. A. Tabak et al., "Machine learning to classify animal species in camera trap images: Applications in ecology," *Methods in Ecology and Evolution*, vol. 10, no. 4, pp. 585-590, 2019.

[5] A. Swanson, M. Kosmala, C. Lintott, R. Simpson, A. Smith, and C. Packer, "Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna," *Scientific data*, vol. 2, no. 1, pp. 1-14, 2015.

[6] S. Beery, G. Van Horn, O. Mac Aodha, and P. Perona, "The iwildcam 2018 challenge dataset," *arXiv preprint arXiv:1904.05986*, 2019.

[7] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 85-100.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009: Ieee, pp. 248-255.

[9] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251-1258.

[10] S. Bianco, R. Cadene, L. Celona, and P. Napoletano, "Benchmark analysis of representative deep neural network architectures," *IEEE Access*, vol. 6, pp. 64270-64277, 2018.

[11] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600-612, 2004.

[12] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2285-2294.

[13] S. Yu, S. Jia, and C. Xu, "Convolutional neural networks for hyperspectral image classification," *Neurocomputing*, vol. 219, pp. 88-98, 2017.

[14] R. Shang, J. He, J. Wang, K. Xu, L. Jiao, and R. Stolkin, "Dense connection and depthwise separable convolution based CNN for polarimetric SAR image classification," *Knowledge-Based Systems*, p. 105542, 2020.

[15] G. Liu, Z. Yin, Y. Jia, and Y. Xie, "Passenger flow estimation based on convolutional neural network in public transportation system," *Knowledge-Based Systems*, vol. 123, pp. 102-115, 2017.

[16] W. Samek, T. Wiegand, and K.-R. Muller, "Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models," *arXiv preprint arXiv:1708.08296*, 2017.
[17] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller, "Explaining nonlinear classification decisions with deep taylor decomposition," Pattern Recognition, vol. 65, pp. 211-222, 2017.

[18] R. Pierrard, J.-P. Poli, and C. Hudelot, "Learning fuzzy relations and properties for explainable artificial intelligence," in 2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), 2018: IEEE, pp. 1-8.

[19] M. Melis, A. Demontis, M. Pintor, A. Sotgiu, and B. Biggio, "secml: A Python Library for Secure and Explainable Machine Learning," arXiv preprint arXiv:1912.10013, 2019.

[20] J. Choo and S. Liu, "Visual analytics for explainable deep learning," IEEE computer graphics and applications, vol. 38, no. 4, pp. 84-92, 2018.

[21] Captum, I. Facebook, 2019. [Online]. Available: https://captum.ai

[22] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in Proceedings of the IEEE international conference on computer vision, 2017, pp. 618-626.

[23] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," arXiv preprint arXiv:1703.04730, 2017.

[24] A. Adefurin et al., "Genetic variation in the alpha1B-adrenergic receptor and vascular response," The pharmacogenomics journal, 2017.

[25] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," Digital Signal Processing, vol. 73, pp. 1-15, 2018.

[26] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, "Evaluating the visualization of what a deep neural network has learned," IEEE transactions on neural networks and learning systems, vol. 28, no. 11, pp. 2660-2673, 2016.

[27] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," arXiv preprint arXiv:1312.6034, 2013.

[28] S. M. Ross, A first course in probability. Pearson Prentice Hall Upper Saddle River, NJ, 2006.

[29] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in Advances in neural information processing systems, 2015, pp. 91-99.

[30] A. Pankajaksh and A. Bhavsar, "Faster RCNN-CNN-Based Joint Model for Bird Part Localization in Images," in Proceedings of 3rd International Conference on Computer Vision and Image Processing, 2020: Springer, pp. 197-212.

[31] S.-J. Hong, Y. Han, S.-Y. Kim, A.-Y. Lee, and G. Kim, "Application of deep-learning methods to bird detection using unmanned aerial vehicle imagery," Sensors, vol. 19, no. 7, p. 1651, 2019.

[32] E. S. Simons and M. K. Hinders, "Automatic counting of birds in a bird deterrence field trial," Ecology and evolution, vol. 9, no. 20, pp. 11878-11890, 2019.

[33] Q. Wang, C. Rasmussen, and C. Song, "Fast, Deep Detection and Tracking of Birds and Nests," in International Symposium on Visual Computing, 2016: Springer, pp. 146-155.

[34] L. B. Boudaoud, F. Maussang, R. Garello, and A. Chevallier, "Marine Bird Detection Based on Deep Learning using High-Resolution Aerial Images," in OCEANS 2019-Marseille, 2019: IEEE, pp. 1-7.

[35] X. Yu, F. Zhou, and M. Chandraker, "Deep deformation network for object landmark localization," in European Conference on Computer Vision, 2016: Springer, pp. 52-70.
[36] S. Huang, M. Gong, and D. Tao, "A coarse-fine network for keypoint localization," in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3028-3037.

[37] D. Lin, X. Shen, C. Lu, and J. Jia, "Deep lac: Deep localization, alignment and classification for fine-grained recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1666-1674.

[38] T. Tang, S. Zhou, Z. Deng, H. Zou, and L. Lei, "Vehicle detection in aerial images based on region convolutional neural networks and hard negative example mining," Sensors, vol. 17, no. 2, p. 336, 2017.