Operational Data Augmentation in Classifying Single Aerial Images of Animals

Emmanuel Okafor and Rik Smit
Institute of Artificial Intelligence
and Cognitive Engineering
University of Groningen,
The Netherlands
e.okafor@rug.nl, riksmithh@gmail.com

Lambert Schomaker and Marco Wiering
Institute of Artificial Intelligence
and Cognitive Engineering
University of Groningen,
The Netherlands
{l.r.b.schomaker, m.a.wiering}@rug.nl

***Abstract*—In deep learning, data augmentation is important to increase the amount of training images to obtain higher classification accuracies. Most data-augmentation methods adopt the use of the following techniques: cropping, mirroring, color casting, scaling and rotation for creating additional training images. In this paper, we propose a novel data-augmentation method that transforms an image into a new image containing multiple rotated copies of the original image in the operational classification stage. The proposed method creates a grid of $n \times n$ cells, in which each cell contains a different randomly rotated image and introduces a natural background in the newly created image. This algorithm is used for creating new training and testing images, and enhances the amount of information in an image. For the experiments, we created a novel dataset with aerial images of cows and natural scene backgrounds using an unmanned aerial vehicle, resulting in a binary classification problem. To classify the images, we used a convolutional neural network (CNN) architecture and compared two loss functions (Hinge loss and cross-entropy loss). Additionally, we compare the CNN to classical feature-based techniques combined with a k-nearest neighbor classifier or a support vector machine. The results show that the pre-trained CNN with our proposed data-augmentation technique yields significantly higher accuracies than all other approaches.**

***Keywords*—Data Augmentation, Convolutional Neural Networks, Classical Feature Descriptors, Supervised Learning, and Aerial Image Classification**

I. Introduction

The use of unmanned aerial vehicles (UAV) has a lot of potential for precision agriculture as well as for livestock monitoring. A previous study [1] recommended that the combination between precision agriculture and remote sensing and UAV methods can be very beneficial for agricultural purposes. Other research [2]–[4] has examined this area of research with the use of UAVs for different tasks. A novel area of research is recognizing aerial imagery with the use of deep neural networks. The study in [5] demonstrates that the use of a convolutional neural network for ground-to-aerial localization yielded a good performance on some datasets. Another interesting study is the use of deep reinforcement learning for active localization of cows [6]. Next to the task of localization, there exists some recent research on the use of UAVs for motion detection and tracking of objects. The study in [7] analysed the merits of the use of optical flow with a coarse segmentation approach for aerial motion detection of animals from several videos. Furthermore, in [8] the authors extended the idea of using UAVs with object detection and tracking algorithms for monitoring wildlife animals. Another approach is detection and tracking of humans from UAV images using local feature extractors and support vector machines [9].

978-1-5090-5795-5/17/$31.00

The idea of data augmentation (DA) has been successfully applied to UAV data as well. In [10], the authors studied augmentation of drone sounds using a publicly available dataset that contains several real-life environmental sounds. Furthermore, the research in [11] explored the use of a DA method for training a deep learning algorithm for recognizing gaits. Another interesting use of DA is the development of a model for 3D pose estimation using motion capture data [12].

Most of the previous data-augmentation techniques transform a training image to multiple training images using techniques such as: cropping, mirroring, color casting, scaling and rotation. In this paper, we propose a novel data-augmentation method that transforms a single input image to another image containing $n \times n$ rotated copies of the original image. This method enhances the amount of information in an image, especially if the image contains a single object like in our study (cow or non-cow background). The aim of this paper is to assess if this novel data-augmentation method leads to higher classification accuracies when combined with different machine learning techniques such as convolutional neural networks or classical feature descriptors on a novel dataset containing aerial images of animals.

Contributions: This paper proposes a novel data-augmentation technique that transforms a train or test image into a novel single image with multiple randomly rotated copies of the input image. To combine the different rotated images, the proposed method puts them in a grid and adds realistic background pixels to glue them together. This approach presents some merits: 1) It provides more informative images which may aid to yield higher accuracies, 2) It does not require an increase in the number of training images compared to other conventional data-augmentation methods, and 3) The

method can also be used to perform data augmentation on test images in the operational stage. The utility of the proposed approach is evaluated by using a CNN which is derived from the original GoogleNet [13] architecture by keeping only several inception modules. For training this CNN we evaluate if there are differences in using the cross-entropy loss function (softmax classifier) compared to using a Hinge loss function. Furthermore, we compared the CNNs to several classical computer vision techniques using original images and data-augmented images. All techniques were used to investigate the recognition accuracies of aerial images of cows in natural scenes, for which we created our own dataset with an unmanned aerial vehicle.

The results show that using fine-tuned CNN models with the proposed data-augmentation technique leads to significantly better results than all other approaches.

Paper Outline: Section II describes the used UAV dataset and the proposed data-augmentation technique. Section III discusses the methods used for classifying the aerial imagery. The experimental results of the derived CNN and the classical techniques are reported in Section IV. Finally, the conclusion is presented in Section V.

II. Dataset and Data Augmentation

A. Dataset Collection

We employed the DJI Phantom 3 Advanced Unmanned Aerial Vehicle (UAV) for collecting video frames of cows and natural backgrounds at different positions and orientations. An illustration of the UAV is shown in Figure 1.

Fig. 1. A photo of the UAV used for this study

We applied manual cut-outs with a fixed size of 100×100 pixels to obtain positive samples of images that contain a cow, while we employed an automatic extraction of negative samples which have no presence of cows in the image. We flew the drone three times over different fields containing cows in order to obtain different samples. A summary of the three subsets of the obtained images with the amount of positive and negative samples, the video streaming time, and the amount of unique objects is reported in Table I. The unique objects denote cows that are recorded at different time frames and therefore have different appearances in time. Figure 2 shows some samples of images of our aerial dataset.

B. Cross-Set Splits

We used cross-set splits whereby each recorded subset is considered as a separate fold. One subset is used for testing and the other subsets are used for the training set. This process is repeated for the three available subsets. The classical feature descriptors combined with supervised learning algorithms and the derived CNN technique are employed for determining the existence of cows in the natural images. We maintain the same dataset splits for all the experiments using the CNN and the feature extraction techniques. The classical techniques employ two image resolutions; 100×100 and 250×250 pixels, and for the experiments carried out with the derived CNN we only used 250×250 pixels.

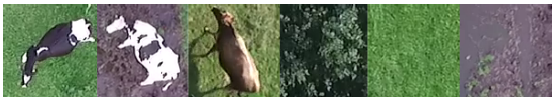


Fig. 2. Sample images of the aerial dataset, showing the presence of cow (positive samples) and non-cow (negative samples). Please note that non-cow images are also diverse.

Algorithm 1 Multi-Orientation Data-Augmentation Algorithm

Input : Given images $I_i(x, y)$ from an input directory, where x, y denote the pixel row and column, and a grid size of $n \times n$.

Output : The data-augmentated versions of the images.

procedure CONSTRUCT A FILELIST WITH N IMAGES FROM AN INPUT DIRECTORY.
  for each image I_i, $i \in N$ **do**
    Initialize the total number of cells $n \times n = M$
    for each image I_i, for each cell $m \in M$ **do**
      Define the size of the image resolution I_i.
      Compute a pad-size $I_q = ceil(I_i)/2$.
      Compute a pad-array I_p using a pixel replication padding technique, given I_i, I_q, pad value set to 'replicate' and the pad direction set to 'both'.
      Rotate I_p with a random angle within the bound $[1, 180^o]$, this yields a new image I_r.
      Adjust the image I_r to I_a such that undesired background introduced during rotation is filled with artificial pixels from the nearest-neighbor pixels.
      Concatenate each I_a into M cells.
      $I_c = [I_a(m), .., I_a(m+3); ...; ..; .., I_a(M)]$
    end for
    Convert the cell structure of I_c into a matrix I_m.
    Resize the image I_m to a size 250×250 pixels.
    Store each $I_m(i)$ into an output directory
  end for
end procedure

C. Multi-Orientation Data Augmentation

We propose a new offline data-augmentation (OFL-DA) algorithm that transforms an input image to a new single image containing multiple randomly rotated versions put in $n \times n$ cells. The use of a larger value for n leads to a new image containing more different poses. The value of n was set to 4 in the experiments, because using higher values of n resulted in making the cow images look very small. An illustration of the proposed data-augmentation method and the overall classification system using the CNN is shown in

TABLE I
STATISTICS OF VIDEO RECORDS AND ANNOTATED DATASETS

	Video ID	Time (s)	Unique Objects	Positive Samples	Negative Samples
1	Subset 1	11	10	37	225
2	Subset 2	43	82	475	2094
3	Subset 3	22	10	50	1100

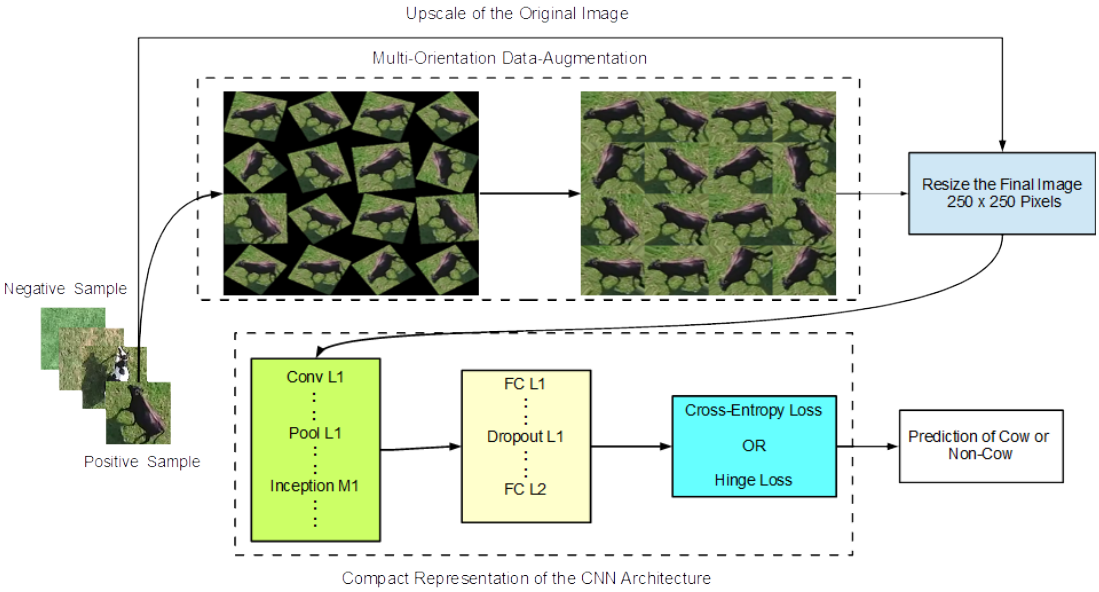


Fig. 3. Block diagram illustrating the proposed method and overall system using the CNN. The column (':') symbol between different layers represents the connections of neural network layers within the derived CNN architecture. The data-augmented image on the top-left is a multi-orientation image without padding and the image on the top-right is the resulting multi-orientation image with padding.

Figure 3. The pseudo-code in Algorithm 1 explains the various transformations of the original image to obtain the multi-orientation image.

After inserting the images in the newly created image, background pixels are added to glue them together. This is done by using the nearest-neighbor pixels around the edges of the images. We will also perform experiments with OFL-DA without rotations (OFL-DA-NR), but we do this only for the classical feature-based techniques.

III. IMAGE RECOGNITION METHODS

A. Three Inception Module CNN Architecture

This architecture is directly derived from the famous GoogleNet architecture as proposed in [13]. We eliminated all the layers after the inception 4a module, except for layers which lead to the first classifier. We do this because the problem under study is more of a binary classification problem and the dataset is quite small. We want to know how the reduced architecture can handle this problem compared to the original GoogleNet. Another modification made with respect to the original GoogleNet architecture is the use of Nesterov's Accelerated Gradient Descent (NAGD) rather than using the conventional stochastic gradient descent (SGD) to update the weights in the deep neural network. The NAGD optimization update rule [14] is described in equations 1 and 2:

$$u_{i+1} = \mu u_i - \alpha_L \nabla L \left(W_i + \mu u_i \right) \tag{1}$$

$$W_{i+1} = W_i + u_{i+1} \tag{2}$$

where $L \in \{L^h, L^c\}$ is the loss function, μ is the momentum value, α_L is the learning rate, u_i is the momentum variable, ∇ is the rate of change in L, i is the iteration number and W_i denote the learnable weights. We employed randomly initalized weights for the scratch CNN and pretrained weights from the ImageNet dataset for the fine-tuned CNN (GoogleNet architecture). In addition to our modification, we remark that the original GoogleNet (in Caffe framework) uses a simple online data-augmentation that involves cropping (with a default crop size of 224×224 pixels), i.e. cutting out several patches from an input image at 5 positions (as five in a dice), and additionally flipping (horizontal reflection) to obtain more samples. During training of the CNN model, it automatically flips each cropped image to double the effective dataset size. The cropping means an act of extracting some portions from an input image. In our customized CNNs, we considered the original and two additional crop sizes: 125×125 and 250×250 pixels. The crop size of 250×250 implies the single actual size of the input image. Furthermore, we evaluated flip and non-flip conditions. All the input images to the CNN have image sizes

TABLE II
THREE INCEPTION MODULE CONVOLUTIONAL NEURAL NETWORK ARCHITECTURE

Layer Type	Patch Size / Stride	Output Size	Depth	Number of Convolutional Filters (6 possible conv)	Blob Parameters
Conv 1	$7 \times 7/2$	$112 \times 112 \times 64$	1		16.06M
Max Pool 1	$3 \times 3/2$	$56 \times 56 \times 64$	0		4.01M
Conv 2	$3 \times 3/2$	$56 \times 56 \times 192$	2	0, 64, 192, 0, 0, 0	12.04M
Max Pool 2	$3 \times 3/2$	$28 \times 28 \times 192$	0		3.01M
Inception 3a		$28 \times 28 \times 256$	2	64, 96, 128, 16, 32, 32	4.01M
Inception 3b		$28 \times 28 \times 480$	2	128, 124, 192, 32, 96, 64	7.53M
Max Pool 3	$3 \times 3/2$	$14 \times 14 \times 480$	0		1.88M
Inception 4a		$14 \times 14 \times 512$	2	192, 96, 208, 16, 48, 64	2.01M
Average Pool 1		$4 \times 4 \times 512$	0		163.84K
Top Conv-1	$1 \times 1/1$	$4 \times 4 \times 128$	1		40.96K
FC 1 / 70% Dropout layer		$1 \times 1 \times 1024$	1 / 0		20.48K
FC 2		$1 \times 1 \times 2$	1		0.04K
Cross Entropy (Softmax) / Hinge Loss		$1 \times 1 \times 2$	0		

of 250×250 pixels. For the OFL-DA image, each cell of the 4×4 grid contains a copy of the input image in a reduced size and the method fills up empty spaces with nearest neighbor pixels.

The derived three inception module CNN architecture is described in Table II. This architecture involves the use of three inception modules that allow the concatenation of filters of different dimensions and sizes into a single new filter [15]. In each inception module, there exist six convolution layers and one pooling layer. Moreover, there exist several rectifiers (ReLUs) which are placed immediately after the convolutional and fully-connected layers. Furthermore, there exist four pooling layers excluding those within the inception modules, two bottom convolutional layers and one top convolutional layer which comes after the average pooling layer. We use one top-1 loss function which employs either the Hinge loss or the cross-entropy loss (for the Softmax classifier). The L_1-norm Hinge loss L^h used in our study can be defined as:

$$L^h(x_i) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \left(\max \left(0, 1 - y_i^k z_k(x_i) \right) \right) \qquad (3)$$

where $y_i^k = \{1, -1\}$, $y_i^k = 1$ if x_i belongs to the target class of the k-th class output unit, and $y_i^k = -1$ if x_i does not belong to the target class. The variable N denotes the total number of training images in a batch. K accounts for the number of class labels and $z_k = x^T w$ is the final activation of the output units. Here, $x \in \mathbb{R}^D$ denote the D-dimensional features of the previous hidden layer, and the learnable weights of the last layer are $w \in \mathbb{R}^{D \times K}$.

The Cross-Entropy Loss L^c used in our study is defined as:

$$L^c(x_i) = -\frac{1}{N} \sum_{i=1}^{N} log \left(\frac{exp(z_d(x_i))}{\sum_{k=1}^{K} exp(z_k(x_i))} \right) \qquad (4)$$

Where d denotes the target class. The fraction within the log accounts for the softmax activation function [16], which computes the probability distribution of the classes in a multi-class classification problem. Note that in this study we are dealing with a binary classification problem and we use 2 output units in the CNN.

The CNN under study consists of two fully connected (FC) layers: FC 1 with a corresponding ReLU computes the hidden unit activations, which is immediately followed by a regularization dropout of 0.7, and the second FC 2 contains the output neurons which represent the negative and positive class. The working operations of the CNN are well explained in the paper [13].

1) CNN Experimental Setup: All experiments were run on the Caffe deep learning framework on a Ge-Force GTX 960 GPU model. The used experimental parameters are as follows: training display interval is set to 40, average loss is set to 40, learning rate is set to 0.001, learning policy is set to step, the step size is set to 4000 iterations, power is set to 0.5, gamma is set to 0.1, the momentum value is set to 0.9, weight decay is set to 0.0002, and maximum iteration is set to 10000, which generates a snapshot model after every 500 iterations (which represent an epoch). This resulted in 20 epochs for the entire training process. The mentioned parameters were not altered during all the experiments for the different model configurations. The training images from the combination of any of the two subsets as reported in Table I, is further split into the ratio 80% for training and 20% for validation. We employed a training batch size set to 20 and testing batch size set to 5 for all experiments, but with different test iterations. The altered parameters for the three subsets of the aerial dataset used with their corresponding splits are described in Table III.

TABLE III
CNN PARAMETERS AND DATASET SPLIT INFORMATION

Parameters	Subset 1	Subset 2	Subset 3
Test Images	262($\sim 7\%$)	2569($\sim 65\%$)	1150($\sim 29\%$)
Training Images	2975($\sim 74\%$)	1129($\sim 28\%$)	2264($\sim 57\%$)
Validation Images	744($\sim 19\%$)	283($\sim 7\%$)	567($\sim 14\%$)
Total Images	3981(100%)	3981(100%)	3981(100%)
Solver Test Iteration (Val/Train)	148	56	113
Test Iterations for Evaluation	52	514	230

We first performed experiments with both the original and our derived CNN trained from scratch on the original images. The preliminary results show that our proposed architecture requires less memory usage and a decrease in training computing time. This is summarized in Table IV. Additionally, our architecture obtains a similar level of performance compared to the original CNN.

TABLE IV
PRELIMINARY EXPERIMENT USING ORIGINAL AND OUR PROPOSED CNN ON THREE CROSS SPLITS OF THE AERIAL IMAGE DATASET

Evaluation/Methods	Derived CNN, NAGD	Original CNN, NAGD
Time (min)	$25.1 \leq t \leq 26.8$	$63.2 \leq t \leq 69.1$
Memory Usage (MB)	752	1079
Average Validation %	99.94	99.94
Average Test %	97.87	97.71
Time Improvement %	61.3 (decrease)	-

B. Classical Features Combined with Supervised Learning Algorithms

In this subsection, we describe the three feature extraction techniques which we use and combine with the k-nearest neighbor classifier and the support vector machine (SVM) with a linear kernel or a radial basis function (RBF) kernel.

1) Color Histogram: The color histogram (Color-Hist) is a feature extraction technique that analyses the pixel color values within an image. For this, the pixel color values of an image which exist as RGB (Red, Green and Blue) are first transformed to HSV (Hue, Saturation, and Value). After that, the value of each pixel in a channel is put in a histogram consisting of different bins. In the experiments, only the saturation channel with a bin size of 32 is used, because it obtained the best performance in preliminary experiments. The resulting feature vector containing 32 values is given to the supervised learning algorithms.

2) Histogram of Oriented Gradients: The histogram of oriented gradients (HOG) [17] feature descriptor analyses patches (local regions) from an image. Then histograms are constructed based on the occurrences of orientation gradients within the patches. The HOG descriptor can process gray or color image information. In this study, we only considered the gray option. The procedure for constructing the HOG is as follows: convert the color images of the aerial imagery into grayscale, then compute the gradients with two gradient kernels to compute the gradient values for each pixel from the grayscale image. The gradients for each pixel within a small block (cell) are put in bins [18], [19], where each bin defines a specific orientation range. The following parameters were used, because they worked best in preliminary experiments: a grid of 2×2 blocks is used, where each block is split into 2×2 cells. The number of orientation bins is set to 4. This results in a feature dimension size of 64. This feature vector is fed as input to the supervised learning algorithms.

3) The Combination of HOG and the Color Histogram: In this technique, the features from both the HOG and Color-Hist are combined to form the HOG-Color-Hist feature descriptor. The features from both HOG and Color-Hist are first computed separately. The optimal parameters used for HOG in the combined feature are different from the HOG descriptor alone, because they gave slightly better results in the preliminary experiments. The HOG parameters used in this technique use 32×32 pixels per cell, for which we used 9 cells in total from 100×100 pixel images with a single block. The number of orientation bins is set to 4 and the final feature dimensionality is 36. We used the hue channel from the color-histogram with 32 bins. These features are normalized and concatenated to obtain the final feature vector with 68 elements.

Several experiments were conducted to determine the best choice of parameters for the used classifiers with the different classical feature descriptors. For the K parameter in K-nearest neighbor (KNN) we tried $K = \{1, 2, 3, 4, 5, 10\}$. The C parameter of the linear SVM is set to $C = 2^{q-1}$, with the explored values $q \in \{1, 2, ..., 19\}$. For the SVM with the RBF kernel, we tried $C = \{1, 2, 3, 5\}$ with $\gamma = 10^{p-1}$, where $p \in \{1, 2, ..., 4\}$. The optimal parameters used for each of the classifiers are reported in Table V. All the algorithms used for the classical techniques were developed in Python.

TABLE V
BEST FOUND PARAMETERS USED FOR THE VARIOUS CLASSIFIERS WITH THE CLASSICAL FEATURE DESCRIPTORS

Classical Techniques	RBF-SVM	Linear SVM	K-NN
HOG	$C = 3, \gamma = 1000$	$C = 8$	$K = 1$
Color-Hist	$C = 1, \gamma = 100$	$C = 8192$	$K = 3$
HOG-Color-Hist	$C = 1, \gamma = 100$	$C = 256$	$K = 3$

IV. EXPERIMENTAL RESULTS

To compute the average results of the different subsets, we compute the weighted average accuracy, which is computed by summing over the relative dataset sizes multiplied with the average accuracies on the different datasets.

A. Evaluation of the CNN Architecture

In our preliminary studies, we carried out experiments on the data-augmentation (OFL-DA) version of our dataset to determine the optimal crop size. We used models generated from the train-validation experiments for evaluating our test sets. We initially employed the scratch CNN with the cross-entropy classification loss, which is combined with or without flipping and with different crop sizes: 125×125, 224×224, and 250×250. The results of these experiments are shown in Figure 4a, and suggest that the optimal method uses a crop size of 224×224 pixels with flipping. This yields an accuracy of 98.18% that occurred at the 5^{th} epoch. We observed in general, that there exist marginal differences between the various settings.

Based on the outcome of this, we used the best crop size with flip settings to carry out the experiments using the scratch and fine-tuned versions of the CNN. For this, we used both the data-augmented dataset (OFL-DA) and the original (ORIG) images. The validation results from Figure 4b show that the scratch and the fine-tuned CNN applied on the two kinds of images converge to a near maximum level of performance. The reason for this lies in the fact that most of the validation images contain similar objects as in the training set. The validation results at the 5^{th} epoch are reported in Table VI. From the table, we can see that the use of the original dataset leads to more overfitting. The results of the different CNNs with the cross-entropy loss function are shown in Figure 4c. From this figure we can observe that the best obtained test accuracy

is obtained by the fine-tuned CNN applied on the OFL-DA images in the 2^{nd} epoch. We further investigated the CNN with the L_1 Hinge Loss, using the earlier mentioned CNN settings (scratch and fine-tuned versions) applied on the two sets of images (OFL-DA and ORIG). The results obtained are shown in Figure 4d.

Based on the performances recorded during this preliminary investigation, we only compared results obtained at the 5^{th} epoch as reported in Table VI. The results show that the fine-tuned CNN trained on the data-augmented images yields higher test classification accuracies when compared to the fine-tuned CNN trained on the original images of the dataset. We compared the different approaches using the binomial distribution of correctly classifying test images. The results show that the fine-tuned CNN trained on the data-augmented images yields significantly higher classification accuracies ($p = 0.01$) when compared to the fine-tuned CNN trained on the original images of the dataset. Overall, the fine-tuned CNNs obtain the best results and combined with the data-augmented images, the results are very good (99.65%). Finally, the results show that overall the use of the cross-entropy loss function leads to better results than the use of the Hinge loss function.

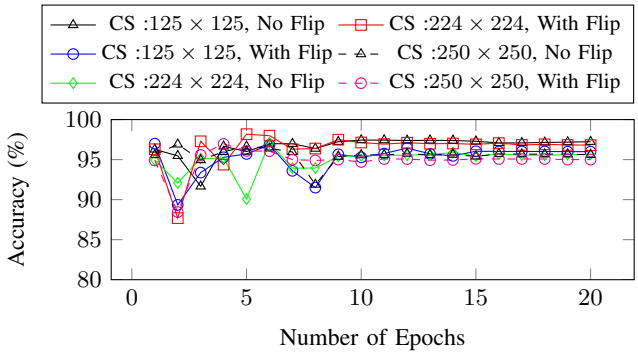
TABLE VI
WEIGHTED MEAN OF THE TEST AND VALIDATION CLASSIFICATION ACCURACIES OF THE CNN APPLIED ON THE AERIAL IMAGERY DATASET AFTER 5 EPOCHS

Evaluation	Method	Cross Entropy Loss	Hinge Loss
Test	Fine-tuned CNN, OFL-DA	**99.65**	**99.65**
	Fine-tuned CNN, ORIG	98.67	98.19
	Scratch CNN, OFL-DA	98.18	96.16
	Scratch CNN, ORIG	97.87	97.51
Validation	Fine-tuned CNN, OFL-DA	**99.94**	**99.94**
	Fine-tuned CNN, ORIG	**100.00**	**100.00**
	Scratch CNN, OFL-DA	**99.68**	**99.81**
	Scratch CNN, ORIG	**99.94**	**99.94**

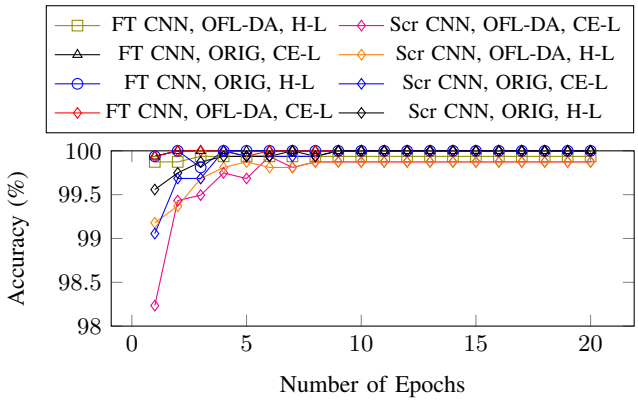
B. Evaluation of Classical Descriptors

The weighted mean test accuracies of the classical techniques on the aerial imagery dataset are reported in Table VII. We observe that the RBF-SVM outperforms the other two classifiers (K-NN and linear SVM) when combined with each of the feature descriptors. Another observation is that the classifiers with the Color-Hist or HOG-Color-Hist features yield better performances than using the HOG descriptor alone. This shows the importance of using color information for this classification problem. Still, the results are significantly worse than the results using the CNN methods.

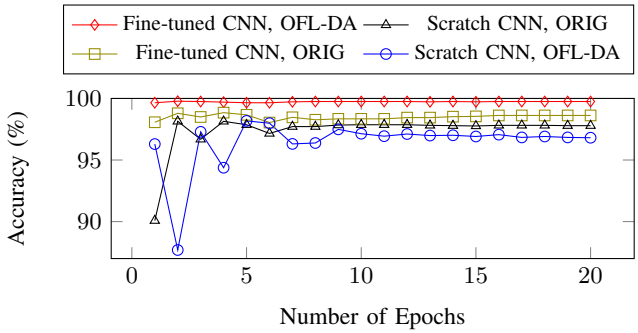
Table VII also shows the results of using the RBF-SVM with different datasets and different feature descriptors using larger images (250×250 pixels). The results show that here data-augmentation does not lead to significantly better results. This can be explained by the fact that the best feature descriptor, the color histogram, is not affected by this data-augmentation method. Finally, we note that the original image with the smaller 100×100 resolution works better for the HOG feature descriptor and therefore also for HOG combined with the color histogram. This can be explained by the fact that we optimized the HOG parameters using the smaller images.



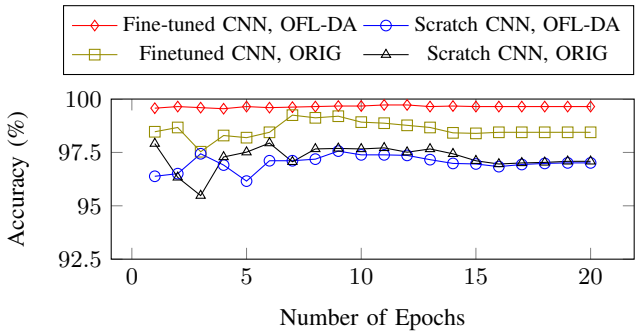
(a) Scratch CNN with cross-entropy loss (softmax classifier) applied on the multi-orientated data-augmentation dataset (offline data augmentation, OFL-DA) using different crop sizes (CS), with and without flips.



(b) Validation set evaluation of the CNN with cross entropy loss (CE-L) and Hinge Loss (H-L) using a crop size of 224×224 and flip. The OFL-DA means the augmented dataset and ORIG means the originally up-scaled images. The FT means Fine-tuned and Scr means Scratch.



(c) Test evaluation of the CNN with cross entropy loss (softmax classifier) using a crop size of 224×224 and flip.



(d) Test evaluation of the CNN with L1-Norm Hinge Loss using a crop size of 224×224 and flip.

Fig. 4. Weighted mean classification accuracy while training for 10000 iterations (20 Epochs).

Although the performances of the CNN techniques are much better, the classical techniques have a lower training computing time: $t \leq 1$ min. This is because of the low dimensionality of the extracted features and the low number of trainable parameters.

TABLE VII
SUMMARY OF THE WEIGHTED MEAN TEST PERFORMANCES FOR ALL CNNs AND THE CLASSICAL METHODS ON OUR DATASET

METHODS	Sub 1	Sub 2	Sub 3	Weighted Mean
Fine-tuned-CNN, OFL-DA, Cross Entropy Loss	100.00	99.73	99.39	99.65
Fine-tuned-CNN, OFL-DA, Hinge Loss	99.62	99.77	99.39	99.65
Fine-tuned-CNN, ORIG, Cross Entropy Loss	99.62	98.29	99.30	98.67
Fine-tuned-CNN, ORIG, Hinge Loss	99.62	97.55	99.30	98.19
Scratch-CNN, OFL-DA, Cross Entropy Loss	98.23	98.72	96.96	98.18
Scratch-CNN, OFL-DA, Hinge Loss	98.08	96.19	95.65	96.16
Scratch-CNN, ORIG, Cross Entropy Loss	98.85	99.34	94.35	97.87
Scratch-CNN, ORIG, Hinge Loss	97.69	98.83	94.52	97.51
RBF-SVM-HOG, ORIG-100×100	96.56	86.99	95.30	90.02
RBF-SVM-Color-Hist, ORIG-100×100	96.56	96.07	96.87	96.33
RBF-SVM-HOG-Color-Hist, ORIG-100×100	96.56	96.11	96.69	96.31
Linear-SVM-HOG, ORIG-100×100	85.88	81.51	95.65	85.88
Linear-SVM-Color-Hist, ORIG-100×100	96.95	93.77	95.83	94.57
Linear-SVM-HOG-Color-Hist, ORIG-100×100	95.80	94.08	93.74	94.09
KNN-HOG, ORIG-100×100	88.17	84.35	96.78	88.19
KNN-Color-Hist, ORIG-100×100	96.56	96.50	94.86	96.03
KNN-HOG-Color-Hist, ORIG-100×100	96.95	96.46	94.78	96.01
RBF-SVM-HOG, ORIG-250×250	85.88	81.51	95.65	85.88
RBF-SVM-Color-Hist, ORIG-250×250	96.57	95.37	96.52	95.78
RBF-SVM-HOG-Color-Hist, ORIG-250×250	85.88	81.51	95.65	85.88
RBF-SVM-HOG, OFL-DA-250×250	85.88	81.51	95.65	85.88
RBF-SVM-Color-Hist, OFL-DA-250×250	96.18	95.25	96.70	95.73
RBF-SVM-HOG-Color-Hist, OFL-DA-250×250	95.04	93.97	96.08	94.65
RBF-SVM-HOG-OFL-DA-NR-250×250	94.66	81.51	86.61	83.84
RBF-SVM-Color-Hist-OFL-DA-NR-250×250	96.56	95.13	96.43	95.60
RBF-SVM-HOG-Color-Hist-OFL-DA-NR-250×250	95.04	91.98	96.43	93.47

V. Conclusion

We developed a novel data-augmentation method that transforms an image into a new image containing multiple random transformations of the image. The new augmentation method does not lead to an increase in the number of training images compared to previously used data-augmentation techniques. We evaluated this method with deep neural networks and feature descriptors combined with supervised learning algorithms on a new dataset of aerial images of cows.

Our study shows that the use of the data-augmented images leads to the best performances when combined with fine-tuned CNNs. Furthermore, the results show that all CNN approaches significantly outperform the classical approaches with or without the use of data augmentation. The performances of the scratch CNNs are worse than the accuracies of the fine-tuned CNNs with data-augmented images which obtain an accuracy of 99.65%. Furthermore, the RBF-SVM yields better classification performances than the K-NN and a linear SVM when combined with the used feature descriptors. It should be noted that our DA algorithm is useful for the CNNs, because although our CNNs are more or less translational invariant, they are not rotational invariant.

The idea of our data-augmentation method can be extended by including different techniques to create new images such as color casting with different illumination effects. Furthermore, the proposed data-augmentation technique can also be combined with other data-augmentation methods to create more training images, which may be useful when dealing with small datasets.

References

[1] C. Zhang and J. M. Kovacs, "The application of small unmanned aerial systems for precision agriculture: a review," *Precision agriculture*, vol. 13, no. 6, pp. 693–712, 2012.

[2] P. Katsigiannis, L. Misopolinos, V. Liakopoulos, T. K. Alexandridis, and G. Zalidis, "An autonomous multi-sensor UAV system for reduced-input precision agriculture applications," in *Control and Automation (MED), 24th Mediterranean Conference on*. IEEE, 2016, pp. 60–64.

[3] V. Lukas, J. Novák, L. Neudert, I. Svobodova, F. Rodriguez-Moreno, M. Edrees, and J. Kren, "The combination of UAV survey and landsat imagery for monitoring of crop vigor in precision agriculture," *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 953–957, 2016.

[4] F. López-Granados, J. Torres-Sánchez, A. Serrano-Pérez, A. I. de Castro, F.-J. Mesas-Carrascosa, and J.-M. Peña, "Early season weed mapping in sunflower using UAV technology: variability of herbicide treatment maps against weed thresholds," *Precision Agriculture*, vol. 17, no. 2, pp. 183–199, 2016.

[5] T.-Y. Lin, Y. Cui, S. Belongie, and J. Hays, "Learning deep representations for ground-to-aerial geolocalization," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[6] J. C. Caicedo and S. Lazebnik, "Active object localization with deep reinforcement learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2488–2496.

[7] Y. Fang, S. Du, R. Abdoola, K. Djouani, and C. Richards, "Motion based animal detection in aerial videos," *Procedia Computer Science*, vol. 92, pp. 13–17, 2016.

[8] L. F. Gonzalez, G. A. Montes, E. Puig, S. Johnson, K. Mengersen, and K. J. Gaston, "Unmanned aerial vehicles (UAVs) and artificial intelligence revolutionizing wildlife monitoring and conservation," *Sensors*, vol. 16, no. 1, p. 97, 2016.

[9] Y. Imamura, S. Okamoto, and J. H. Lee, "Human tracking by a multirotor drone using HOG features and linear SVM on images captured by a monocular camera," in *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1, 2016.

[10] S. Jeon, J.-W. Shin, Y.-J. Lee, W.-H. Kim, Y. Kwon, and H.-Y. Yang, "Empirical study of drone sound detection in real-life environment with deep neural networks," *arXiv preprint arXiv:1701.05779*, 2017.

[11] C. C. Charalambous and A. A. Bharath, "A data augmentation methodology for training machine/deep learning gait recognition algorithms," *arXiv preprint arXiv:1610.07570*, 2016.

[12] G. Rogez and C. Schmid, "Mocap-guided data augmentation for 3d pose estimation in the wild," in *Advances in Neural Information Processing Systems*, 2016, pp. 3108–3116.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[14] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the importance of initialization and momentum in deep learning." *International Conference on Machine Learning ICML (3)*, vol. 28, pp. 1139–1147, 2013.

[15] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[16] E. Okafor, P. Pawara, F. Karaaba, O. Surinta, V. Codreanu, L. Schomaker, and M. Wiering, "Comparative study between deep learning and bag of visual words for wild-animal recognition," in *IEEE Symposium Series for Computational Intelligence (SSCI)*, 2016.

[17] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition (CVPR). IEEE Computer Society Conference on*, vol. 1, 2005, pp. 886–893.

[18] O. L. Junior, D. Delgado, V. Gonçalves, and U. Nunes, "Trainable classifier-fusion schemes: an application to pedestrian detection," in *Intelligent Transportation Systems*, vol. 2, 2009.

[19] K. Takahashi, S. Takahashi, Y. Cui, and M. Hashimoto, "Remarks on computational facial expression recognition from HOG features using quaternion multi-layer neural network," in *Engineering Applications of Neural Networks*. Springer, 2014, pp. 15–24.