Introducing a New Workflow for Pig Posture Classification Based on a Combination of YOLO and EfficientNet

Jan-Hendrik Witte
University of Oldenburg
jan-hendrik.witte@uni-oldenburg.de

Jorge Marx Goméz
University of Oldenburg
jorge.marx.gomez@uni-oldenburg.de

Abstract

This paper introduces a pipeline for image-based pig posture classification by applying YOLOv5 for pig detection and EfficientNet for subsequent pig posture classification into 'lying' and 'notLying'. A high-quality dataset consisting of 5311 heterogeneous images from different sources with 78215 bounding box annotations was created. The bounding box annotations were then used to create a separate dataset for image classification, consisting of 9209 and 7855 images for each 'lying' and 'notLying'. The YOLOv5 model achieves an $AP^{IoU=0.5}$ of 0.994 for pig detection, while EfficientNet achieves a precision of 0.93 for pig posture classification. Comparing the results of the proposed method with other approaches found in literature, it shows that significant improvements in terms of accuracy can be achieved by splitting the classification of pig posture into separate models. This research provides a foundation for the continued development of real-time monitoring and assistance systems in pig Precision Livestock Farming.

1. Introduction

Structures of modern pig livestock farming and pork production have been undergoing major changes in recent years. Data from the Federal Statistical Office in Germany shows the opposite trend of steadily decreasing numbers of farms [1] with simultaneously increasing numbers of animals per farm [2] while also maintaining volatile slaughter prices over the past years[1], which poses and will continue to pose great challenges for the farmer. At the same time, politics and society alike are calling for more sustainable and more animal-friendly husbandry [3], which puts additional pressure on the farmer and makes economically profitable pig livestock farming increasingly difficult. These challenges cannot be met with conventional methods, which is why new and innovative solutions are needed. As a result, research in the domain of Precision Livestock Farming (PLF) has increased in recent years. PLF describes systems that utilize modern camera and sensor technologies to enable automatic real-time monitoring in livestock production to supervise animal health, welfare and behavior [3, 4]. This involves the automated acquisition, processing, analysis and evaluation of sensor-based data like temperature, humidity or CO_2 concentration [5] as well as image and video data [6, 7]. These different types of information and data sources hold the potential to enable data-driven assistance systems that support farmers in their daily work and would help them adapt to the constantly changing conditions in pig livestock farming.

To create such systems, methods are first needed that allow the automated processing of these different types of data streams in the form of image, video and sensor data. Video data alone can be used for a variety of use cases in PLF, many of which can already be found in literature. Considering practical applications such as the counting of pigs [8], the tracking of pigs over specific time intervals [9], the detection of aggressive behavior among group housed pigs [6], or the automatic weight estimation [10], there are a number of use cases which are addressed by utilizing image data based on camera recordings. The image based detection of pigs poses a particular challenge with specific problems such as the grouping, overlapping and occlusion of pigs, their different postures, orientations and positions, as well as constantly changing factors such as different lighting conditions, soiling of animals or occlusion caused by objects in the pen. Due to their ability to generalize, the use of deep learning methods from the field of computer vision has been proven effective in addressing these challenges.

In the *DigiSchwein* project funded by the BMEL and BLE[2], various use cases and problems have been formulated that will be explored by applying deep

[1] https://bit.ly/3zikeuG, last visit: 06.05.2021

[2] https://bit.ly/35pid1X, last visit: 06.05.2021

URI: https://hdl.handle.net/10125/79472
978-0-9981331-5-7

learning methods. Some of the main topics are the early detection of diseases as well as the early detection of tail biting events in piglet rearing and fattening. With regard to these use cases, the activity level of pigs in the pen is a crucial factor that can be used as an indicator for the early recognition of abnormal behavior. In this context, the determination of the activity can be described as a collection of different sub-tasks. On the one hand, methods for the automated recognition and localization of pigs on image data are needed. On the other hand, methods are required for the classification of the individual pig posture, which could then be used in subsequent processes to derive an aggregated activity level inside the pen. For this reason, this paper presents a method for the detection and localization of pigs, as well as the classification of individual posture into *lying* and *notLying*, which differs from other methods found in the literature.

The term object detection describes the detection and localization of objects of a defined class by enclosing bounding boxes around the respective object in the image. With one exception, all papers found in literature address the classification of pig posture as a one-step object detection task. In this case, the defined bounding box classes correspond directly to the respective postures that should be detected by the object detection model. This includes examples like *lying*, *lying on side*, *lying on belly* in [11] or classes such as *standing* and *lying lateral* in [12]. However, this leads to the problem that classes are considered to be disjoint and self-sufficient from each other, which in principle is not the case for the classification of pig posture, since each defined class and object belongs to the higher-level class *pig*. A more intuitive approach would therefore be to detach the actual detection of the object *pig* from the classification of the state or condition. For this reason, this paper investigates whether the process of detection and localization of pigs as well as the classification of posture can be separated into different model architectures or sub-tasks.

Fig 1 shows a conceptual workflow of this process. An input image is given into an object detection model to detect and localize pigs. The output of the model in the form of bounding box predictions is then used to crop each detected object based on the bounding box coordinates. These cropped images are passed into an image classification model that assigns the input to one of the defined classes. It will be examined how these changes will affect performance and execution speed, or what advantages and disadvantages this would have. For this purpose, a dataset consisting of 5311 highly heterogeneous images with a total of 78215 individually labeled bounding boxes was created. The cropped bounding boxes were then used to create a training and test dataset for binary image classification of pigs into *lying* and *notLying*.

The paper is structured as follows: First, the current state of the art in the field of pig posture classification is presented. The primary focus lies on papers that apply deep learning models and architectures as well as their respective performance in terms of precision. Next, a description of the criteria used to select the applied deep learning architectures and models for pig detection and posture classification in this paper is presented. This is followed by a description of the created datasets for each task as well as the hardware utilized to train and evaluate the respective models. The results are presented based on quantitative evaluation metrics, applying specific evaluation metrics for bounding box prediction and posture classification. The method presented in this paper is then compared to other approaches identified in the literature by using a publicly available dataset as a benchmark. The interpretation of the results discusses the insights gained from the quantitative evaluation as well as the results of the comparison. The conclusion and outlook summarize the results and describe how they can be used in future research.

2. Related Work

Two approaches can be distinguished for the posture classification of pigs using deep learning: One-stage object detection methods like You Only Look Once (YOLO) [13] or Single Shot Multibox Detector (SSD) [14] and two-stage object detection like Faster-R-CNN [15]. Two-stage object detection can be understood as a combination of different model architectures and methods, in which results from one model instance are passed on to another. In the case of Faster R-CNN, object detection is achieved through the combination of a Region Proposal Network (RPN) for proposal generation of bounding boxes and the RoI Pooling Layer, which further processes the proposals of the RPN to generate the final output of the model [15]. One-stage object detection like YOLO or SSD do not apply a method to generate regional proposals, instead bounding box prediction are directly performed on the input image, which is why these kinds of models usually have a faster execution time than two-stage object detection approaches [15]. On the other hand, two-stage object detection models are considered to be more accurate, which can lead to a use case specific trade-off in the selection of the respective object detection method.

Both one-stage and two-stage object detection have been applied in the literature to classify pig posture. Alameer et al. applied YOLO with a ResNet50

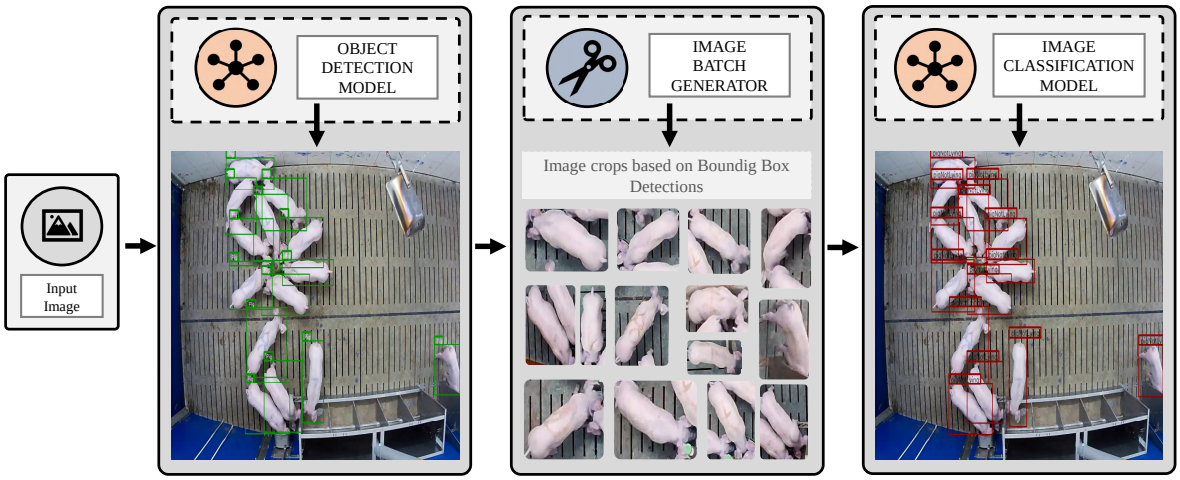

Figure 1. Example workflow of proposed method.

backbone to classify pig posture into *lying lateral*, *lying sternal*, *sitting* and *standing* [12]. The accuracy of the detected postures was evaluated based on the mAP and achieved an accuracy of 0.98. Manual inspection of the publicly available dataset used in the paper indicated that there is little variance in the training data in regard to locations, backgrounds, viewing angles, camera positions or number of animals, which could limit the model's ability to generalize.

Nasirahmadi et al. evaluated Faster R-CNN, SSD and Region-based Fully Convolutional Network (R-FCN) with different backbone combinations based on ResNet, InceptionV2 and Inception ResNet for pig posture classification [11]. Out of the tested combinations, the R-FCN with a ResNet101 achieved the best overall performance. An average precision of 0.93, 0.95 and 0.92 was achieved for each *standing*, *lying on side* and *lying on belly*, which results in an overall mAP of 0.93.

Riekert et al. use Faster R-CNN to classify pig posture into *lying* and *notLying* [16]. Unlike the other papers, Neural Architecture Search (NAS) is applied as backbone for feature extraction instead of a ResNet, which improved performance. The authors also made their dataset publicly available. The manual inspection of the dataset showed that it is more complex than the data of Alameer et al. due to varying camera positions, viewing angles, the high and varying number of animals per image as well as different pens and backgrounds, which is why the dataset is also applied in this paper to to compare the proposed method with other approaches found in literature.

The only paper in the literature that applies a similar approach to the one presented here was introduced by Shao et al. [17]. Yolov5 architecture is applied to first detect pigs on images using bounding boxes. The classification of pigs into *standing*, *lying*, *lying on side* and *exploring* is done by utilizing the DeepLabv3 semantic segmentation architecture and a ResNet101 as backbone. This method achieved an accuracy of 0.92. Since the dataset is composed of images from a single camera, the viewing angle, camera lens, as well as the background is identical for each image, which might limit the generalizability of the model. The presented method in this paper can be distinguished from the method of Shao et al. by:

a) Replacing the DeepLabv3 semantic segmentation architecture with an EfficientNet model for subsequent posture classification.

b) Using a more complex dataset for training and evaluation, which consists of different camera viewing angles, camera lenses, locations and backgrounds as well as a varying number of animals per image.

c) Evaluating the approach using a publicly available dataset to demonstrate the potential advantages and disadvantages of the proposed method.

3. Materials and methods

3.1. Model selection

The method presented in this paper consists of two elements: An object detection model for the detection and localization of *pig* objects and an image classification model for the subsequent classification of the cropped bounding box images into *lying* and *notLying*. The model selection for the respective task

was conducted by defined selection criteria. These criteria are based both on models and architectures that were already used in literature as well as on the requirements for PLF systems that have been mentioned in the PLF literature. The following criteria were defined:

1) **Prediction accuracy**: The prediction of the respective models should be as accurate as possible [18].

2) **Prediction speed**: Model inference should be in real-time [19].

3) **Cost efficiency**: The respective models should be as resource efficient as possible to allow a potential deployment to low cost hardware [20].

The website paperswithcode[3] provides an overview of all published real-time object detection architectures and their benchmark results on the COCO test-dev, a popular dataset on which model performance is evaluated and benchmarked. Since the COCO format is also used for training and evaluation in this paper, this overview served as a basis for selecting the object detection model based on the defined criteria. The same applies to the selection of the image classification model, since paperswithcode also offers an overview for image classification models[4]. The following models and architectures were selected:

YOLOv5: Since the YOLO architecture has already been used in the PLF literature for pig posture classification and also represents the state of the art in real-time object detection, it matched the specified criteria. YOLOv5 is the latest installment of the YOLO architecture, but there is currently no official paper for this version. The latest paper release is YOLOv4 by [13], which applies specific methods and concepts summarized under the terms *bag of freebies* and *bag of specials* to improve accuracy and execution speed compared to YOLOv3 and other architectures such as EfficientDet. The performance was further improved by introducing a network scaling approach by modifying model depth, width, resolution and structure [21]. The comparison of the two official implementations of YOLOv4 [22] and YOLOv5 [23] resulted in the selection of the YOLOv5 implementation, as it was more suitable for the context of this paper.

EfficientNet: EfficientNets are the current state of the art in image classification on benchmark datasets like ImageNet, while also being smaller and faster compared to other architectures like ResNet or Inception [24]. At the core, EfficientNets are based on a traditional convolutional neural network architecture. By applying the introduced compound scaling method to uniformly scale network depth, width and resolution as well as a neural architecture search, different EfficientNet variants were created depending on the selected compound coefficient [24]. In the context of this paper, EfficientNet-B0 was used since is the smallest of the EfficientNet variants and therefore fits the specified criteria.

[3]https://bit.ly/2SoCz91, last visit: 06.05.2021

[4]https://bit.ly/35lFP7T, last visit: 06.05.2021

Table 1. Posture classes found in literature.

Author	Classes
Alameer et al. [12]	*lying lateral, lying sternal, sitting, standing*
Nasirahmadi et al. [11]	*lying on side, lying on belly, standing*
Riekert et al. [16]	*lying, not lying*
Shao et al. [17]	*lying on stomach, lying on side, exploring*

3.2. Posture class selection

With respect to the defined posture classes, different approaches can be observed in the literature. Tab. 1 gives an overview of all posture classes that could be identified. It shows that, with respect to the defined posture classes, both differences and similarities can be observed. Each of the papers listed defines the class *lying*, whereby Alameer et al. [12], Nasirahmadi et al. [11] and Shao et al. [17] further distinguish the class into *sternal* and *lateral lying*, *lying on stomach* and *lying on side* as well as *lying on side* and *lying on belly*, respectively. Both Alameer et al. and Nasirahmadi et al. define the class *standing*. The classes *sitting* and *exploring* are solely found in the paper of Alameer el al. and Shao et al. respectively, with only Alameer explaining in more detail why this class was included. The authors state, that *sternal* and *lateral lying* as well as *sitting* "[...] may be indicative of a pig being cold or suffering from abdominal pain [...]" or "[...] may be indicative of locomotory problems" [12]. Riekert et al. stand out in some respects considering the selected classes for posture classification. Instead of specific classes such as *standing*, *sitting* or *exploring*, only the class *not lying* is defined in addition to the *lying* class, so that the trained model ultimately only distinguishes between lying and non-lying pigs.

The approach presented in this paper is primarily intended to provide a basis for determining and deriving pen based activity levels. The ratio of lying and non-lying pigs is to be used as a central indicator for determining the activity level, which is why the differentiation of pigs into *lying* and *notLying* is

considered sufficient. For this reason, the approach of Riekert et al. is adapted and the classes *lying* and *notLying* are adopted.

3.3. Dataset description

Object detection dataset: To evaluate the performance of the selected object detection model, a dataset consisting of 5311 images with a total of 78215 high quality bounding box annotations was created. The open source tool *Labelme* was used to annotate the images for model training and evaluation [25]. To ensure heterogeneity in the data, the dataset was compiled from a combination of several datasets. When assembling the dataset, efforts were made to consider as many different backgrounds, camera angles, shooting lenses and lighting conditions in the images as possible. Care was also taken to include PLF specific challenges such as piling, occlusion or overlapping of pigs. To reduce the overall annotation time, the dataset creation process was divided into different phases:

1) A sample of 1000 out of the total 5311 images were manually annotated. This sample was first randomly extracted from the overall data pool and then inspected to ensure data variety was sufficient in the sample.

2) A YOLOv5 model based on the YOLOv5x6 pretrained checkpoint was trained using the annotated dataset and was subsequently applied on the unlabeled data.

3) The predictions of the model were saved in JSON format and loaded into the Labelme annotation tool, where the predictions of the model were subsequently checked and adjusted manually to ultimately reduce the amount of manual label work. This process was repeated at certain intervals until finally all of the 5311 images were annotated and inspected.

The object detection data set is composed as follows: Psota et al. published a dataset with a total of 2000 keypoint annotated images from 17 different locations [26]. Each image of this set was extracted and annotated with bounding boxes. A sum of 720 images were provided by the KoVeSch[5] project of the Lower Saxony Chamber of Agriculture, which includes 360 pictures from piglet rearing and 360 pictures with fattening pigs. Another data source was provided by the InnoPig[6] project, which contains 1268 images for piglet rearing and 418 images for fattening pigs. A total of 600 images from the publicly available dataset of Alameer et al. were also used to build the dataset [12]. The remaining 305 images were extracted from the dataset published by Riekert et al. [16]. Compared to the other datasets found in literature, the presented dataset has a higher variation of data within the set, as different locations, camera angles, camera lenses, a wide variety of pigs per image as well as day and night images were considered. The data set was divided into a training and test set, with 80% being used for training and 20% being used for testing. Fig. 2 shows samples from the dataset that illustrate the data heterogeneity.

Image classification dataset: For the creation of the image classification dataset, each annotated bounding box was extracted from the respective images and stored as separate file. Out of the 78215 extracted files, a total of 9209 were labeled as *lying* and 7855 were labeled as *notLying*. Since the Keras function *image_dataset_from_directory*[7] was used to load the dataset before training, the files only had to be moved to a folder representing the respective class. The dataset was also divided into 80 % training and 20% testing.

3.4. Test environment and setup

Model training was performed on a desktop workstation with two Nvidia RTX 3090 with 24 GB VRAM each, a Threadripper 3960X and 64 GB RAM. For the object detection task of detecting and locating pigs in images, the YOLOv5 implementation of Jocher et al. was applied [23]. Standard parameters were used for training[8]. The model was trained for 300 epochs with a batch size of 128 and the images were scaled to 640×640. Based on the selection criteria, the smallest checkpoint, *YOLOv5s*, was used for initial training and to enable transfer learning.
For the image classification task of classifying pigs into *lying* and *notLying*, the official Keras implementation of EfficientNets[9] was applied. The smallest possible version, EfficientNetB0, was selected as baseline. Transfer learning was applied by first freezing all but the top layers when initializing the model and utilizing pre-trained *ImageNet* weights. Second, the last 15 layers were unfrozen to fit the model on the new data. In both steps, the model was trained for 30 epochs with a batch size of 64 and an image input size of 224×224. Data augmentation was applied in form of random horizontal flipping, random zoom, random rotation as well as random crops and random contrast changes. Since the classification is binary, sigmoid was

[5]https://bit.ly/3cBzNnp, last visit: 07.06.2021
[6]https://bit.ly/3woSzq6, last visit: 15.06.2021
[7]https://bit.ly/3w4osEk, last visit: 08.06.2021
[8]https://bit.ly/2TqHHtd, last visit: 08.06.21
[9]https://bit.ly/3iIlqBI, last visit: 07.06.2021

Figure 2. Example images from the dataset.

used as the activation in the last layer. Adam was used as optimizer and binary crossentropy was specified as loss function.

4. Results

4.1. Quantitative Results

For the object detection of the pigs as well as the classification of the posture, different metrics are applied for evaluation. The commonly used metric for evaluation and benchmarking object detection models is the mAP, which is the mean of the Average Precisions (AP) based on a set of different Intersection over Union (IoU) thresholds [27]. IoU is defined as the similarity between the ground truth annotation and the predicted annotation present in the image and is determined by dividing the intersection with the area of union [28]. Common thresholds to calculate the mAP are values in the range 0.5 to 0.95 with a threshold step size of 0.05, represented as AP@[.5:.05:.95][29]. To evaluate the overall image classification performance over defined classes, the standard precision metric is used. The Keras framework provides an evaluation method for this purpose. For both object detection and image classification, precision and recall are also provided, describing what portion of the positive predictions were actually correct and what portion of the actual positive predictions were detected correctly. In addition, the inference time, number of frames per second (FPS) that can be processed and the parameter size of the respective model are specified as well. The number of parameters describes the model size and can affect the required hardware to train and operationalize the respective model. A comparison of the performance of the object detection models examined in this paper with other architectures found in literature was not conducted since the sizes of the applied models do not match the defined criteria in Sec. 3.1. For instance, the ResNet101 backbone used by [11] alone has a parameter size of 58.16 million [30], which could make real-time executability on low-cost hardware problematic. Tab. 2 summarizes the results.

Out of the box, the applied YOLOv5 model architecture achieves very good results in the task of pig detection. With an $AP^{IoU=0.5}$ of 0.994 on the test set, there is only little room left to improve the AP for this threshold. Looking at the mAP, it shows that the models accuracy becomes progressively weaker when the IoU threshold increases. The higher the IoU threshold, the less margin is allowed in the deviation of the ground truth bounding box and predicted bounding box, so the AP is usually lower at higher thresholds. Therefore it is not surprising that the mAP with a value of 0.899 is lower than the $AP^{IoU=0.5}$. By changing to a larger baseline of the YOLOv5 model, the mAP could potentially be improved. However, this would happen at the expense of inference time and parameter size, since larger models also have larger hardware requirements. Precision and recall with values of 0.982 and 0.975 are very close to each other, meaning that the model predicts almost all

Table 2. Results for pig object detection and posture image classification.

Task	Model	Class	Metric	Performance
Object Detection	YOLOv5	pig	$AP^{IoU=0.5}$	0.994
			mAP	0.899
			Precision	0.982
			Recall	0.975
			Inference	0.01 sec
			FPS	100
			Parameter	7,056,607
Image Classification	EfficientNet-B0	lying	Precision	0.94
			Recall	0.928
		notLying	Precision	0.92
			Recall	0.928
		overall	Precision	0.93
		Inference		0.03 sec
		FPS		33
		Parameter		4,057,253

positive samples correctly and also identifies nearly all actual positive samples correctly as well. The inference time for the object detection model demonstrates the capability for real-time applications. With the ability to process 100 FPS, even video recordings with 60+ FPS could theoretically be processed with the corresponding hardware.

In the task of classifying pigs into *lying* and *notLying*, an overall precision of 0.93 was achieved on the test set using the EfficientNet-B0 architecture. When comparing the precision for the individual classes *lying* and *notLying*, it is noticeable that the precision for *lying* is slightly higher than for *notLying*. This could be related to the fact that the *notLying* class indirectly includes subordinate classes such as *sitting* or *standing*, which could make the classification more complex. Here, balancing the data set by adding additional *notLying* images could address these differences. Similar to object detection, the precision and recall values for both classes are close to each other. Although the number of parameters for the EfficientNet-B0 model is lower than for the YOLOv5 model, the inference time is significantly higher. With an inference time of 0.03 seconds, approximately 33 FPS can be processed.

4.2. Comparison with other methods

To illustrate the advantages of the method presented in this paper, the one-step classification of pig posture using an object detection model is compared with the two-step classification process using a combination of an object detection and an image classification model. The following experiments were formulated:

1) Classification of pig posture into *lying* and *notLying* using YOLOv5 object detection model.

2) Classification of pig posture into *lying* and *notLying* using YOLOv5 object detection model to first detect pigs on images and then classify pig posture using an EfficientNet-B0 image classification model.

The publicly available data set of Riekert et al. provides the basis for conducting the formulated experiments. The dataset consists of a total of 305 manually annotated images with of a total of 7277 bounding box annotations, in which 5077 images were assigned to the classes *lying* and 2200 to the classes *notLying*. For the experiments, a subset of the dataset was used, consisting of 265 images with 4526 *lying* and 1392 *notLying* samples. In order to use the dataset for YOLOv5 model training, the dataset was first transferred into the COCO data format and then converted into YOLO format.

To prepare the image classification dataset, the same steps as described in Sec. 3.3 were applied, resulting in 4526 *lying* and 1392 *notlying* samples.Likewise, the same specifications as in in Sec. 3.4 were used for model training. To address class imbalance in the image classification dataset, the *class_weight* function provided by the sklearn[10] package was applied to take the different weights of the classes into account during training. The results for both experiments can be seen in Tab. 3. In the first experiment, an overall precision of 0.793 can be achieved, with an overall $AP^{IoU=0.5}$ of 0.744. Looking at the individual classes *lying* and *notLying*, significant differences in precision, recall and

[10]https://bit.ly/2WvKgeV, last visit: 08.06.2021

Table 3. Experiment results.

Experiment	Model	Class	Metric	Performance
1)	YOLOv5	lying	$AP^{IoU=0.5}$	0.815
			Precision	0.869
			Recall	0.648
		notLying	$AP^{IoU=0.5}$	0.673
			Precision	0.718
			Recall	0.648
		overall	$AP^{IoU=0.5}$	0.744
			Precision	0.793
			Recall	0.648
2)	YOLOv5	pig	$AP^{IoU=0.5}$	0.86
			Precision	0.903
			Recall	0.763
	EfficientNet-B0	lying	Precision	0.96
			Recall	0.938
		notLying	Precision	0.86
			Recall	0.91
		overall	Precision	0.91

$AP^{IoU=0.5}$ can be observed. The $AP^{IoU=0.5}$ and precision with 0.672 and 0.718 for *notLying* are noticeably lower than the results for *lying*. These differences may be explained by the imbalance in the data set as well as by the general higher complexity of the *notLying* class. It can also be observed that the recall for both classes is considerably lower than the precision. Conversely, this means that the model has difficulties to identify the actual positive samples in the test set. When comparing the results of the two experiments with respect to the classification precision for the classes *lying* and *notLying*, it can be observed that by splitting the use case into separate object detection and image classification tasks, a significant increase in performance can be achieved. With regards to object detection, by merging the two classes into the higher-level class *pig*, a 11.4% jump in performance comparing the overall $AP^{IoU=0.5}$ in Exp. 1 and the $AP^{IoU=0.5}$ in Exp. 2 can be achieved. The difference is even more noticeable for the overall precision, which is 13.87% higher with a precision value of 0.903 compared to the precision of 0.793. It shows, that the model becomes more capable in the actual detection of *pig* objects.

A similar observation can be made when looking at the results of the image classification task. With a overall precision of 0.91 on the test set for pig posture classification into *lying* and *notLying*, the performance is significantly higher compared to the 0.793 precision of the first experiment. In general, it can be observed that the difference between precision and recall both in the object detection of pigs as well as in the classification of the posture is reduced. The exception is the precision and recall values of the EfficientNet-B0 for the class *notLying*, where the recall is higher than the precision, meaning that for this class the model yields many false positive predictions. Again, this could be due to the imbalance of classes in the data set.

In total, the result show that by combining both architectures for pig posture classification, the margin for error can be reduced. Not only is the general detection of pigs more reliable, the actual classification of pig posture can also be enhanced as well. Following example illustrates this in more detail:

Based on an exemplary number of 20 pigs per pen, using the $AP^{IoU=0.5}$ of Tab. 3 for the first experiment, it can be stated that on average, 74% of all bounding boxes classes are predicted correctly, which results in 14,8 correct detections per image. Compared to the results of $AP^{IoU=0.5}$ of the second experiment, 86% of all pigs in the image can be detected correctly, which corresponds to 17.2 pigs per image. Based on the precision of 91% for the classification task based on the EfficientNet, out of the 17.2 detected pigs, subsequently 15.65 postures per image can be predicted correctly. Compared to the 14.8 correct predictions when only using the YOLOv5 architecture, this makes a difference of 5.74% or 1,15 pigs per image. Processing video data with higher FPS values, this factor can be significant. The improved recall values when splitting the posture classification is not even taken into account in this example. However, the increase in performance comes at the expense of inference time. Only using YOLOv5, 100 FPS can

be achieved. By adding the EfficientNet-B0 model for classification, the FPS drops to 25, since the inference time of the model has to be included as well. Nevertheless, an execution of this pipeline would still be possible with the proper hardware.

4.3. Interpretation of the results

With the proposed method, the accuracy in predicting pig posture on images can be improved using light weight deep learning models with low parameter size and real-time capability. One potential advantage that has not yet been discussed is the ability to filter out detected pigs or pigs parts that cannot be assigned to a distinct posture. When preparing the training data, care is taken to consistently annotate each object that should be detected. Looking at Fig. 3, annotating the image for object detection of *lying* and *notLying*, challenges and limitations might occur. The problem is that the pigs in the red boxes cannot immediately be assigned to a specific or distinct class. It is not apparent whether the animal is lying or not. Regarding the annotation process, the following options are possible:

1) Uncertain objects are omitted and not annotated.

2) An additional class is created, e.g. *unsafe*.

Both options are problematic. Ignoring these cases could lead to inconsistency within the model, since the respective entities still belong to the superordinate class *pig* and are also similar to the other annotated objects in the image. Adding another class could lead to problems as well, since the results in Sec. 4.2 show that adding additional classes could decrease overall model performance in terms of accuracy and precision. Using the approach described in Exp. 2, an alternative method can be identified that could be further investigated in future research. By using subsequent image classification, such cases could be filtered out by an image classification model by adding an additional class *unsure* in this step instead of in the object detection. Thus, in object detection, each object could still be annotated as *pig* and processed accordingly in the subsequent image classification.

4.4. Conclusion and outlook

In this paper, a alternative approach for pig posture classification in *lying* and *notLying* has been introduced. The approach is able to distinguish itself from other methods found in literature by considering pig posture classification as a two-step classification process consisting of an object detection step and an image classification step. For both tasks, specific

Figure 3. Difficult postures.

criteria were defined on which the respective models were selected. These criteria were based on previously used model architectures as well as on the mentioned requirements for PLF systems found in literature. The result presents a model combination that has not been implemented for pig posture classification before. First, state of the art YOLOv5 real-time object detection architecture is used to detect and localize pigs on images with an $AP^{IoU=0.5}$ of 0.994. Second, the EfficientNet-B0 is applied to classify pig posture based on the cropped bounding box images from the object detection model with an precision of 0.93. For each model, a custom dataset with high image heterogeneity has been created. The object detection model was trained on a dataset consisting of a total of 5311 images with 78215 high quality bounding box annotations. The annotated bounding boxes were then used to create an additional dataset for image classification of pig posture, of which 9209 were labeled as *lying* and 7855 were labeled as *notLying*. The comparison of this method with one-step methods in the literature using a publicly available dataset shows, that splitting the posture classification into sub-tasks leads to an increase in performance in terms of $AP^{IoU=0.5}$, precision and recall. Since only default parameters were used for training, optimization of the parameters can be pursued in future research to further improve model performance or inference time. Although the YOLOv5 architecture and the EfficientNet-B0 variant are already relatively small in terms of parameter size, future research may also investigate the type of low-cost hardware on which they can actually be applied. Edge devices like a Raspberry Pi or the NVIDIA Jetson Nano could be conceivable here. Regarding the development of early warning systems for farmers, this work can serve as a basis for future research to develop assistance systems based on resource-efficient and real-time capable deep learning methods.

References

[1] Statisitsches Bundesamt, "Betriebe: Deutschland, Jahre, Tierarten," 2021.

[2] Statisitsches Bundesamt, "Gehaltene Tiere: Deutschland, Jahre, Tierarten," 2020.

[3] D. Berckmans, "Precision livestock farming technologies for welfare management in intensive livestock systems," *Revue scientifique et technique (International Office of Epizootics)*, vol. 33, no. 1, pp. 189–196, 2014.

[4] R. B. D'Eath, M. Jack, A. Futro, D. Talbot, Q. Zhu, D. Barclay, and E. M. Baxter, "Automatic early warning of tail biting in pigs: 3D cameras can detect lowered tail posture before an outbreak," *PloS one*, vol. 13, no. 4, p. e0194524, 2018.

[5] J. Cowton, I. Kyriazakis, T. Plötz, and J. Bacardit, "A Combined Deep Learning GRU-Autoencoder for the Early Detection of Respiratory Disease in Pigs Using Multiple Environmental Sensors," *Sensors (Basel, Switzerland)*, vol. 18, no. 8, p. 2521, 2018.

[6] C. Chen, W. Zhu, J. Steibel, J. Siegford, K. Wurtz, J. Han, and T. Norton, "Recognition of aggressive episodes of pigs based on convolutional neural network and long short-term memory," *Computers and Electronics in Agriculture*, vol. 169, p. 105166, 2020.

[7] C. Chijioke Ojukwu, Y. Feng, G. Jia, H. Zhao, and H. Ta, "Development of a computer vision system to detect inactivity in group-housed pigs," *International Journal of Agricultural and Biological Engineering*, vol. 13, no. 1, pp. 42–46, 2020.

[8] G. Chen, S. Shen, L. Wen, S. Luo, and L. Bo, "Efficient Pig Counting in Crowds with Keypoints Tracking and Spatial-aware Temporal Response Filtering," 2020.

[9] J. Cowton, I. Kyriazakis, and J. Bacardit, "Automated Individual Pig Localisation, Tracking and Behaviour Metric Extraction Using Deep Learning," *IEEE Access*, vol. 7, pp. 108049–108060, 2019.

[10] Y. Cang, H. He, and Y. Qiao, "An Intelligent Pig Weights Estimate Method Based on Deep Learning in Sow Stall Environments," *IEEE Access*, vol. 7, no. 99, pp. 164867–164875, 2019.

[11] A. Nasirahmadi, B. Sturm, S. Edwards, K.-H. Jeppsson, A.-C. Olsson, S. Müller, and O. Hensel, "Deep Learning and Machine Vision Approaches for Posture Detection of Individual Pigs," *Sensors (Basel, Switzerland)*, vol. 19, no. 17, 2019.

[12] A. Alameer, I. Kyriazakis, and J. Bacardit, "Automated recognition of postures and drinking behaviour for the detection of compromised health in pigs," *Scientific reports*, vol. 10, no. 1, p. 13665, 2020.

[13] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020.

[14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," 2016.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," 2015.

[16] M. Riekert, A. Klein, F. Adrion, C. Hoffmann, and E. Gallmann, "Automatically detecting pig position and posture by 2D camera imaging and deep learning," *Computers and Electronics in Agriculture*, vol. 174, p. 105391, 2020.

[17] H. Shao, J. Pu, and J. Mu, "Pig-Posture Recognition Based on Computer Vision: Dataset and Exploration," *Animals : an open access journal from MDPI*, vol. 11, no. 5, 2021.

[18] T. Norton, C. Chen, M. L. V. Larsen, and D. Berckmans, "Review: Precision livestock farming: building 'digital representations' to bring the animals closer to the farmer," *animal*, vol. 13, no. 12, pp. 3009–3017, 2019.

[19] S. Lee, H. Ahn, J. Seo, Y. Chung, D. Park, and S. Pan, "Practical Monitoring of Undergrown Pigs for IoT-Based Large-Scale Smart Farm," *IEEE Access*, vol. 7, pp. 173796–173810, 2019.

[20] Banhazi, H. Lehr, J. L. Black, H. Crabtree, and D. Berckmans, "Precision Livestock Farming: An international review of scientific and commercial aspects," *International Journal of Agricultural and Biological Engineering*, vol. 5, no. 3, pp. 1–9, 2012.

[21] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-YOLOv4: Scaling Cross Stage Partial Network," 2020.

[22] Alexey et al., "AlexeyAB/darknet: YOLOv4 pre-release," 2020.

[23] G. e. a. Jocher, "ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations," 2021.

[24] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 6105–6114, PMLR, 2019.

[25] K. Wada, "labelme: Image Polygonal Annotation with Python," 2016.

[26] E. T. Psota, M. Mittek, L. C. Pérez, T. Schmidt, and B. Mote, "Multi-Pig Part Detection and Association with a Fully-Convolutional Network," *Sensors (Basel, Switzerland)*, vol. 19, no. 4, p. 852, 2019.

[27] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," 2017.

[28] M. A. Rahman and Y. Wang, "Optimizing Intersection-Over-Union in Deep Neural Networks for Image Segmentation," in *ISVC*, 2016.

[29] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, "A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit," *Electronics*, vol. 10, no. 3, p. 279, 2021.

[30] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.