Security Fence Inspection at Airports Using Object Detection

Nils Friederich[1] Andreas Specker[3,4] Jürgen Beyerer[3,2,4]
[1]Karlsruhe Institute of Technology, Institute for Automation and Applied Informatics
[2]Karlsruhe Institute of Technology, Institute for Anthropomatics and Robotics
[3]Fraunhofer IOSB [4]Fraunhofer Center for Machine Learning
nils.friederich@kit.edu {andreas.specker, juergen.beyerer}@iosb.fraunhofer.de

Abstract

To ensure the security of airports, it is essential to protect the airside from unauthorized access. For this purpose, security fences are commonly used, but they require regular inspection to detect damages. However, due to the growing shortage of human specialists and the large manual effort, there is the need for automated methods. The aim is to automatically inspect the fence for damage with the help of an autonomous robot. In this work, we explore object detection methods to address the fence inspection task and localize various types of damages. In addition to evaluating four State-of-the-Art (SOTA) object detection models, we analyze the impact of several design criteria, aiming at adapting to the task-specific challenges. This includes contrast adjustment, optimization of hyperparameters, and utilization of modern backbones. The experimental results indicate that our optimized You Only Look Once v5 (YOLOv5) model achieves the highest accuracy of the four methods with an increase of 6.9% points in Average Precision (AP) compared to the baseline. Moreover, we show the real-time capability of the model. The trained models are published on GitHub: https://github.com/N-Friederich/airport_fence_inspection.

1. Introduction

In contemporary times, airplanes have assumed a crucial role in global transportation. Ensuring the safety of passengers, cargo, and machinery is of great importance. This requires appropriate safety mechanisms, both onboard the aircraft and within the airport infrastructure. Protecting sensitive areas such as the airside is a major challenge for airport operators. In Germany, for instance, there are over 540 airfields, out of which 15 are classified as international airports according to § 27d Paragraph 1 Luftverkehrsgesetz (LuftVG) [1] [9]. To obtain this classification, airfields must secure their sensitive areas, including the airside, against unauthorized access by adhering to § 8 Luftsicherheitsgesetz (LuftSiG)[1]. Appropriate security fences are a common practice to protect these areas [10]. These fences must be regularly checked for damage in accordance with § 8 and § 9 LuftSiG[1]. Even minor damage to the fence potentially allows animals to enter the airfield and pose a danger to themselves, people, and machinery [10]. However, the availability of skilled human personnel to perform fence inspections is becoming increasingly limited [3]. Therefore, exploring automated methods to monitor this real-world surveillance application, such as utilizing mobile robots with cameras for detecting damages, is highly valuable.

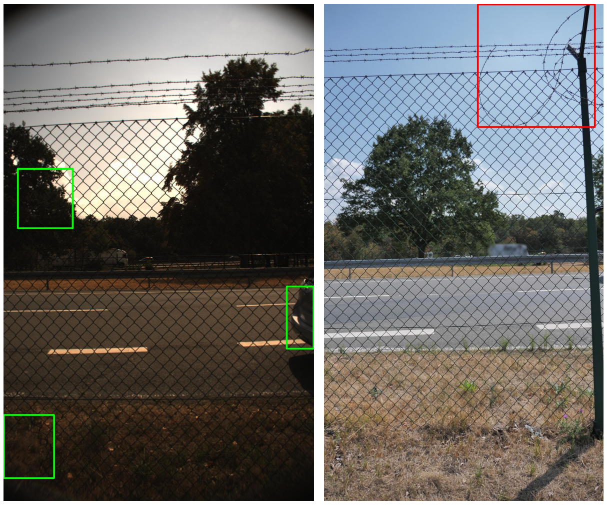


Figure 1. **Examples of damaged security fences** – The Bounding Box (BBox) colors symbolize different types of damage: Green marks a hole in the fence; Red marks damage to the climb-over-protection.

To implement such an automatic system, this work focuses on 2D object detection methods for three main rea-

[1] https://www.gesetze-im-internet.de/ (Gesetz im Netz - Federal Ministry of Justice), Date 01/09/2023

sons. First, the existing literature offers numerous robust methods to effectively tackle this task [5, 11, 17, 46]. Second, using cheap camera sensors is adequate for capturing the necessary imagery. Last, 2D image processing is computationally less heavy compared to, e.g., processing 3D data from a stereo camera.

In general, object detection methods aim at identifying and localizing specific objects or patterns within an input image. In the context of this work, our objective is to detect two commonly occurring types of damages within fence images captured at airports using a self-recorded dataset. Two examples of airport fences are presented in Fig. 1. There is a wire mesh structure in the lower part as a passage barrier and multiple rows of barbed wire in the upper part for climbing-over protection. Damage can occur in both sections. However, damage detection needs a clear differentiation between the fence and structures in the background. Moderate contrast in many areas, such as with the trees in the background, hardens the task. In addition, background clutter, e.g., leaves, further complicates the detection process, especially with the intricate wire mesh. To overcome these challenges, various techniques, including contrast adjustment, are examined throughout this work. For this purpose, SOTA deep learning methods, namely YOLOv5 [17], Task-aligned One-stage Object Detection (TOOD) [11], VarifocalNet (VFNet) [46], and Deformable DEtetction TRansformer (DETR) [51], are evaluated and compared for their potential in addressing the detection challenges associated with the security fence inspection task. Ideally, the resulting detection system should work autonomously on a mobile robot. However, this requires the most economical operation possible with reliable damage detection on affordable hardware. Therefore, we also investigate the tradeoff between speed and accuracy.

In summary, the main contributions of this paper are threefold:

- We conduct the first analysis of SOTA object detection methods for the security fence inspection use case.
- Our thorough evaluation of various design choices highlights key factors for strong damage detection results.
- The resulting real-time model demonstrates remarkable performance and generalization ability and, thus, provides a strong baseline for future research.

2. Related Work

The automated damage detection at airport fences requires Computer Vision (CV) algorithms [12]. In this use case, a simple image classification approach would be insufficient, resulting in a time-consuming search game for human operators. On the other hand, precise segmentation is not required for this task, as it does not demand detailed segmentation of each object instance wire. In addition, creating segmentation labels for intricate objects such as the wire mesh structure by human annotators would be both time-consuming and costly [21]. Therefore, object detection is utilized as a compromise between classification and segmentation. For the purpose of object detection, Deep Learning (DL) methods have gained prominence over classical CV methods due to hierarchical feature extraction, higher accuracy, and improved generalization capabilities [16,27,29,37]. For object detection methods, a differentiation can be made between anchor-based and anchor-free methods. Whereas anchor-based methods often converge faster, anchor-free methods require fewer hyperparameters and may have stronger generalization capabilities. Whether this is true in the context of this thesis is evaluated using the anchor-based method YOLOv5 and the anchor-free methods TOOD, VFNet and Deformable DETR.

Regardless of the model type, DL models often encounter issues with overfitting, particularly when dealing with small datasets. To mitigate this issue, pre-trained models are commonly employed. Since no pre-trained model tailored explicitly for the use case has been published, a default pre-trained model is utilized, such as those trained on the Common Objects in Contexts (COCO) dataset [22,40]. Furthermore, to the best of our knowledge, no appropriate datasets for security fence inspection have been published. Although there are related use cases, such as defencing [14, 18, 26], these datasets consist of images taken in closer proximity and different spatial contexts [14].

3. Methodology

This paper thoroughly examines the use of SOTA DL methods with different characteristics regarding their suitability for the damage detection task and derives best practices concerning design criteria. In detail, YOLOv5 [17], TOOD [11], VFNet [46], and Deformable DETR [51] are considered. After motivating these choices in Sec. 3.1, several adaptions are introduced to increase the detection performance for the task under real-world conditions. The overall goal is to identify the best design characteristics for DL methods from a quantitative perspective and further investigate this method concerning the influence of input image resolution to achieve a beneficial trade-off between detection results and computational complexity.

3.1. Deep Learning Methods

Recently, numerous new DL methods have been introduced [4, 6, 11, 17, 46, 51]. In terms of real-time object detection, several derivatives of the YOLO approach [20, 28, 34, 42] have proven suitable for various real-world applications [43, 48]. For instance, YOLOv5 achieves good detection results at lower operational expenses. However, YOLOv5 and its predecessors [1, 30, 31] are anchor-based, which may lead to limitations in generalization capabilities [23]. Therefore, two anchor-free DL methods are included in the analysis, namely TOOD [11] and VFNet [46].

All these three methods were developed as CNN-based methods [11, 17, 46]. Since transformer-based models promise improved generalization capabilities [7], the transformer-based Deformable DETR [51], a successor of the popular Vision Transformer (ViT)-based DETR [4], is investigated. However, Transformers, such as ViT, typically require more training data than Convolutional Neural Networks (CNNs) [44]. Since the available data for the fence inspection task is limited, further investigations need to be conducted.

3.2. Optimizations

In this work, we thoroughly study various design parameters to improve damage detection in security fences under real-world settings. In the following, the considered aspects are motivated and introduced.

Numerical stability: When implementing DL methods, numerical instabilities such as exploding gradients or zero divisions may occur. These numerical instabilities can lead to a degradation of the training results, which is why we eliminate them to improve the meaningfulness of the experiments. We contributed our code changes to the original code repositories.

Regularization: Regularization of DL models is crucial for preventing overfitting on small datasets with few Regions of Interest (RoIs) per image. For this, primarily three adaptations are investigated. First, the image weighting technique from YOLOv5 is used to over-represent difficult training examples. Due to the small training dataset, edge cases that occur rarely may otherwise be covered by the background noise of decent images. Second, optimizers with regularization abilities like Adam [19] or AdamW [25] are investigated. To prevent gradient oscillations but at the same time allow for a steep gradient descent, the impact of learning rate adjustments is explored.

Data augmentation: Data augmentation methods aim at increasing the diversity in small-scale datasets to prevent overfitting and improve robustness. Due to the small amount of data with few damages each, the impact of data augmentation methods like mosaic and affine transformations are investigated.

Contrast enhancement: Poor contrast, e.g., caused by low light during dusk or dawn, presents a significant challenge in detecting damages on airport fences. In such cases, the fine structures of the fences do not stand out clearly against the background. Pre-processing images with contrast enhancement methods prior to damage detection alleviates the problem. Contrast adjustment can generally be executed on the entire image or separately for multiple image regions. We compare both global and local contrast enhancement methods represented by Histogram equalization (HistEqu) [35, 36] and Contrast Limited Adaptive Histogram Equalization (CLAHE) [52], respectively.

Backbone: While YOLOv5 utilizes a modern CSPDarknet [38, 39] as backbone [17], TOOD and VFNet rely on variants of the Residual Network (ResNet) [15] and ResNeXt [41] architectures. However, more recent backbones such as Res2Net [13] or ConvNeXt [24] show better performance in various tasks [47, 49]. Therefore, these backbones are applied in conjunction with TOOD and VFNet. Analogous to the original backbones, we pre-train these backbones on the COCO dataset first.

Hyperparameter tuning: The choice of appropriate hyperparameters is essential to assure good performance, especially if few training data are available. In addition, the fence inspection task requires strong generalization capabilities. Due to the different conditions and demands, hyperparameters proposed by the original works might not be optimal in damage detection. As a result, detailed studies concerning the choice of hyperparameters are conducted.

Image resolution: When object detectors are deployed in real-world applications, fast computation is crucial. For instance, if the processing is performed on autonomous platforms, such as robots. The inference speed of object detectors is greatly affected by the resolution of the input images. Higher-resolution images provide a more detailed context, enabling improved detection of damages, while the computational complexity increases. Thus, achieving a suitable trade-off between detection accuracy and computational requirements is essential.

4. Experiments

For maximum reproducibility, the hardware and software stack was kept constant during all experiments. The official implementations of YOLOv5 (v6.2) [17] and MMDetection (MMDet) (v2.25.1) [5] were used as the basis for our adaptions and experiments. The methods were then executed using Nvidia's A6000 GPU and Intel's Xeon Silver 4210R CPU.

4.1. Dataset & Evaluation Metrics

Since there is no publicly available dataset for the task, a dataset of airport fence damages was created. Therefore, video sequences of different sections were recorded using two different camera models, namely a FLIR[2] camera model and Panasonic's GH5[3]. A total of 5 datasets were recorded, 3 with the FLIR and 2 with the GH5 camera. Then all images with damage were labeled, images without damage were sorted out and were not considered further. This results in 5 video sequences with an overall 475 video

[2] flir.eu, Date: 01/09/2023

[3] panasonic.com, Date: 01/09/2023

Case	Training	Validation	Test
1	FLIR	FLIR	FLIR
2	FLIR	FLIR	GH5
3	FLIR+GH5	FLIR+GH5	FLIR+GH5

Table 1. **Dataset splits** – Each investigation case specifies which the datasets used for training, validation, and testing. Training and evaluation are performed in each examination case according to the LOOCV.

frames and 725 annotated damages, divided into 104 climb-over defects and 621 holes. The images recorded with the FLIR camera have a resolution of 1920 × 1200 and those with the GH5 camera of 1920 × 1080, respectively.

This work considers three different cases, each reflecting another real-world scenario. The cases differ regarding the training, validation, and testing data, as shown in Tab. 1. Case 1 is the specialization case when training data from the exact camera used in the application is available. Case 2 evaluates the generalization performance since training and test data originate from different camera models with dissimilar characteristics. In the last Case 3, data from both camera models are used for all splits to evaluate the case when diverse data is available for training.

To ensure meaningful evaluation results, Leave-One-Out Cross-Validation (LOOCV) is performed in each of the three study cases to compensate for the small size of the dataset. In each split, another video sequence is leveraged for training, resulting in 12 splits.

The COCO AP [22] serves as the primary metric for both evaluation and validation. The results given represent the average across all three cases and will be abbreviated as Avg. AP in the following.

4.2. Baseline

Each method's baseline is evaluated on the 12 Leave-One-Out Cross-Validation (LOOCV) splits. For this purpose, the original implementations of the methods were slightly modified. For YOLOv5, only Pytorch's recommended measures for reproducibility[4] were added. This ensures better comparability of experiments. Unfortunately, this was impossible for the other three methods in MMDet 2.25.1. Nevertheless, to reduce the standard deviation between the training runs and to be able to make more meaningful comparisons, three runs were performed for each data split. For training, four changes were made to the original configurations. First, the batch size was reduced from 32 to 8 to allow a training with faster gradient descent. Second, to reduce the oscillation of the metrics validation curve during training, the learning rate was reduced to 5e-2. Third, the number of epochs had to be doubled for training convergence. Fourth and last, FP16 built-in training for faster training and lower memory consumption is used.

Method	Backbone	Avg. AP	Case 2 AP
YOLOv5 [17]	n6	53.52±21	25.86±8
	s6	55.33±21	27.42±7
	m6	59.53±17	37.44±2
	l6	61.37±15	41.84±4
	x6	**62.19**±14	**43.34**±0
TOOD [11]	ResNet50	66.14±11	50.42±2
	ResNet101	67.03±12	**51.95**±4
	ResNeXt101-64x4d	**67.10**±12	50.84±2
VFNet [46]	ResNet50	65.64±14	47.22±3
	ResNet101	65.78±13	47.86±2
	ResNeXt101-64x4d	**67.75**±12	**50.28**±3
Def. DETR [51]	ResNet50	**61.13**±14	**42.11**±5

Table 2. **Baseline results** – Different backbone configurations for each method are compared. For TOOD and VFNet, all configs use Deformable Convolutions (DConvs) [8, 50] and Multi-Scaling as additional data augmentation strategy. The best result for each configuration is highlighted in bold.

For all models, pre-trained COCO models are utilized. The models were then fine-tuned with the fence inspection dataset, whereby the resolution was adjusted to 768 pixels on the longest image side. Tab. 2 provides the baseline results of the four methods.

The results indicate that TOOD and VFNet provide the best results with 67.10% and AP 67.75% AP. YOLOv5 achieves worse outcomes with 62.19% AP, though still surpassing Deformable DETR by 2.06% points. One reason for the poor accuracy of Deformable DETR could be the limited training data, a general problem with transformers. Since the efficiency of Deformable DETR is significantly worse than YOLOv5 due to its transformer-based construction, the Deformable DETR method is not considered further in the remainder of this paper. One reason for the poorer results of YOLOv5 is the subpar generalization capability. Comparing the results for Case 2 in Tab. 2, it is apparent that the anchor-free TOOD and VFNet methods generalize remarkably stronger to unseen data. Whether this weakness of YOLOv5 remains despite the optimizations in the further chapters is investigated in Sec. 4.6.

4.3. Regularization

After training the baseline, optimizations are made for the three remaining methods. We have adjusted the YOLOv5 implementation to enable training with rectangular images training in conjunction with random shuffling and mosaic data augmentation [17]. Furthermore, different hyperparameter settings proved beneficial for the m6, l6, and x6 variants of YOLOv5 to achieve better convergence toward the global optimum and prevent overfitting. On the one hand, the OneCycle learning rate [33] is increased from 1e-4 to 1e-3 to enable faster convergence of

[4] pytorch.org, Date 01/09/2023

Backbone	Params (M)	FLOPs (B)	Avg. AP
n6	3.2	4.7	60.71±18
s6	12.6	17	62.36±15
m6	35.7	50.3	64.68±14
l6	76.8	111.8	**66.05**±14
x6	140.7	210.5	64.85±14

Table 3. **YOLOv5 baseline optimization results** – The best result is highlighted in bold.

Method	Experiment	Avg. AP	Case 1 AP	Case 2 AP	Case 3 AP
YOLOv5 [17]	Regularization	66.05±14	73.45±4	47.74±4	76.97±2
	CLAHE	66.22±14	73.87±3	47.28±1	**77.51**±2
	HistEqu	**67.16**±14	**75.46**±4	**48.88**±2	77.14±2
TOOD [11]	Baseline	67.10±12	73.24±4	50.84±2	**77.24**±2
	CLAHE	64.52±14	72.07±4	45.86±5	75.63±3
	HistEqu	**67.62**±11	**73.33**±4	**52.31**±1	77.22±2
VFNet [46]	Baseline	**67.75**±12	**74.14**±2	**50.87**±3	78.25±2
	CLAHE	65.14±15	72.97±3	44.54±3	77.91±2
	HistEqu	**67.49**±13	**73.73**±2	**50.40**±3	**78.36**±2

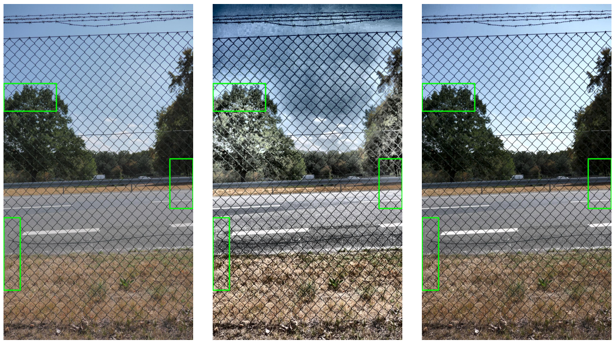
Table 4. **HistEqu and CLAHE results** – Results obtained with the best configuration of methods. The first line of each block indicates the best experiments so far on the original dataset. For comparison, the best results of YOLOv5 were taken from Sec. 4.3 and for TOOD and VFNet from Sec. 4.2. The best results for each DL method are highlighted in bold.

the deeper models m6, l6 and x6 and better exploit the hill climbing properties in gradient optimization. Second, more data augmentation is employed for enhanced regularization. For this, the percentage of image scaling is increased from $[-50\%, +50\%]$ to $[-90\%, +90\%]$. In addition, MixUp [45] is applied with a probability of 10%. Regarding TOOD and VFNet, no significant enhancements were observed.

The optimized YOLOv5 results are presented in Tab. 3. The results significantly surpass the baseline results. This is attributed to the increased diversity of data during training through Mosaic Data Augmentation and further regularization against overfitting introduced by shuffling. In total, these adjustments resulted in an improvement of 3.86% points in AP when comparing the best configurations. However, the best model is not the largest x6, but l6. The x6 model tends to overfit and performs notably worse with 64.85% AP. Even the increased data augmentation and additional regularization cannot compensate for this. Therefore, YOLOv5l6 is used as the best model in the following.

4.4. Contrast Adjustment

The two contrast adjustment methods CLAHE and HistEqu are compared in Tab. 4. The results indicate superior performance of the global method HistEqu regardless of the detection approach. One reason for this could be the over-adjustment of CLAHE in certain regions. Especially worse results concerning the generalization Case 2 support this hypothesis. Since GH5 images already show good contrast, an additional contrast adjustment leads to over-adjustment. Fig. 2 visualizes the differences between both methods for an image captured by the GH5 camera. The CLAHE method, as shown in Fig. 2b, clearly over-adjusts, compared to HistEqu, which is depicted in Fig. 2c. These overfits occur in areas with a high difference between light and dark pixels, such as trees and the sky. This leads to a very unnatural appearance of the image. As a result, parts of the fence structure are hardly recognizable.



(a) Original (b) CLAHE (c) HistEqu

Figure 2. **CLAHE vs. HistEqu** – CLAHE leads to over-adjustments compared to HistEqu. Due to the good contrast in the original image, the contrast is lowered by HistEqu. Nevertheless, the holes are clearly recognizable. In contrast, CLAHE results in too bright areas. Similar to dark areas, the fence structure is difficult to recognize.

4.5. Hyperparameter Optimization

The hyperparameters are optimized using HistEqu preprocessing. Analogous to regularization, the MMDet implementation methods TOOD and VFNet provide no significantly improved results. As a result, hyperparameter optimization focuses on YOLOv5. We found that choosing a learning rate of 5e-3 and applying image weighting turned out to be beneficial. This manual hyperparameter optimization increases the AP from 67.16% to 68.45%. Besides, numerous settings freezing different stages of the backbone, and the use of Adam [19] and AdamW [25] as the optimizer were evaluated to achieve stronger regularization and thereby a more stable training. We also evaluated several settings regarding the affine transformations to achieve a higher generalization. However, none of the mentioned adjustments led to significantly improved results.

Thereafter, an automatic hyperparameter tuning was performed. First, all previous internal evaluations of all 12 LOOCV splits were used, and the Pearson Correlation Coefficient between the average AP across the splits and the AP of each individual split was determined. Subsequently, the split is identified that correlates most with the average AP over all splits. This split is leveraged for automatic hyperparameter tuning.

We apply the Genetic Algorithm (GA) [32] implemented

Method	Backbone	DConv	Param (M)	FLOPs (B)	COCO AP	Avg. AP
TOOD	ResNet101	×	53.2	149.0	49.3	67.62±11
TOOD+	ConvNeXt-T	×	35.7	154.2	44.9	65.86±13
TOOD	ConvNeXt-T	×	35.7	154.2	48.6	**67.76**±12
TOOD	Res2Net101	×	51.7	220.1	45.2	62.05±9
TOOD	Res2Net101	✓	54.5	187.4	50.9	65.13±9
VFNet	ResNeXt101-32x4d	✓	55.1	208.1	49.7	67.49±13
VFNet+	ConvNeXt-T	×	36.5	161.1	44.5	66.63±13
VFNet	ConvNeXt-T	×	36.5	161.1	48.9	64.14±15
VFNet*	Res2Net101	×	52.4	227.0	49.2	65.77±14
VFNet*	Res2Net101	✓	54.9	187.5	51.1	**68.09**±12

Table 5. **TOOD and VFNet results with new backbones** – In each case, the first line of a block represents the best training so far of the methods from Tab. 4. Best Avg. AP (calculated on our fence dataset) is marked bold. *Pre-trained weights used. +Uses original configurations.

in YOLOv5 for automatic hyperparameter optimization in the predefined configuration, except for a few changes. Based on our previous findings, we reduce the defined search space and exclude the affine transformations rotation, shearing, perspective, and flipping since their use leads to significant degradation. Finally, automatic hyperparameter tuning is executed for 500 iterations with the remaining 21 hyperparameters. In each iteration, one or more hyperparameter adjustments are sampled according to the GA policy and then evaluated in a complete training run without early stopping. The most significant effects were observed in reducing the probability of Mosaic Data Augmentation from 100% to 91.5%, since the network requires original data to capture the inherent structure. Additionally, increasing the variation of the saturation in ColorJitter augmentation from $[-70\%, +70\%]$ to $[-89\%, +89\%]$ lead to notable improvement. In total, the optimized model achieves 69.09% in AP on average across all data splits.

4.6. Backbones

After hyperparameter tuning, modern SOTA backbones are evaluated in conjunction with TOOD and VFNet. Besides, the influence of using DConv within the Res2Net architecture is examined. The results of the so-far best models and the new pre-trained ones are given in Tab. 5. For each training session, the AP of the pre-trained network on the COCO dataset is presented in addition to the AP for our dataset. In the case of TOOD, for instance, the best pre-trained network on COCO is not necessarily the best network on our dataset. This is because the classes and the class semantics in the COCO dataset deviate considerably from those in this work. However, it provides a rough indication when further consideration of a backbone is not promising. The findings indicate that TOOD in conjunction with ConvNeXt achieves the highest accuracy. Regarding VFNet, Res2Net as the backbone performs best. Despite the significant improvement in accuracy with the new backbones, TOOD and VFNet do not surpass YOLOv5 in AP. Since YOLOv5 is also more resource efficient due to its design as a real-time object detector, YOLOv5 was selected as the best model and is utilized in the remainder of this paper.

Damage Type	Metric	YOLOv5		Improvement
		Baseline	Hyp. Opt.	
All	AP	62.19 ± 14	69.09 ± 12	+6.90
	AP^{small}	21.69 ± 14	26.80 ± 18	+5.11
	AP^{medium}	65.04 ± 11	70.75 ± 9	+5.71
	AP^{large}	68.52 ± 25	83.41 ± 10	+14.89
Climb over defect	AP	77.12 ± 12	86.53 ± 6	+9.41
	AP^{small}	–	–	–
	AP^{medium}	80.80 ± 10	89.50 ± 4	+8.70
	AP^{large}	77.81 ± 12	86.80 ± 6	+8.99
Hole	AP	47.26 ± 18	51.66 ± 18	+4.40
	AP^{small}	21.69 ± 14	26.80 ± 18	+5.11
	AP^{medium}	50.90 ± 17	54.30 ± 18	+3.40
	AP^{large}	45.82 ± 41	74.88 ± 16	+29.06

Table 6. **Defect results** – Comparison between the different types and sizes of damages. Results are given for baseline (see Sec. 4.2) and the Hyp. Opt. (see Sec. 4.5) as the best training of YOLOv5. The different ranges small, medium and large were defined as follows: $0 < AP^{small} \leq$ 24,000 pixels, 24,000 pixels $< AP^{medium} \leq$ 100,000 pixels and 100,000 pixels $< AP^{large}$.

4.7. In-depth Analysis

So far, all analyses have been performed with the AP across all types of failure. This showcased remarkable progress over the baseline with 6.9% points. This section thoroughly delves into the effects of the proposed optimizations to identify strengths and weaknesses of the system.

Types and area size of fence defects: Tab. 6 investigates the results for each defect type and different sizes of damages for YOLOv5. For this purpose, the damages are divided into three classes based on the covered area in pixels. Damage up to a size of 24,000 pixels is considered small. Correspondingly, damage ranging from 24,000 pixels to 100,000 pixels and over 100,000 pixels as medium or large, respectively. Thereby, 8% of all damages are small, 77% medium and 15% small. In general, the AP difference between the damage types decreases by the optimizations. However, the difference is still a considerable 24.87% points. The stronger detection of the climb-over-protection defects can be explained by their characteristic appearance and by the angle of view. Typically, the damage is in front of the bright sky and, therefore, discriminates well from the background, even under poor lighting conditions (see Fig. 1). In contrast, the wire mesh exhibits poor contrast. The next striking feature in the baseline is the very high standard deviation of 41 for large holes. This finding suggests unstable generalization capabilities and great dependence from the training and validation data. One reason for this is that in the AP^{large}, the holes are nearly normally distributed up to 500,000 pixels. Therefore, training splits with few large boxes may exceed the generalization capability of the baseline to evaluation splits with huge boxes. The

Metric	YOLOv5 Baseline	YOLOv5 Hyp. Opt.	Improvement
Class Error	0.44 ± 1	0.10 ± 0	-0.33
Localization Error	3.03 ± 3	0.80 ± 1	-2.23
As+Localization Error	0.03 ± 0	0 ± 0	-0.03
Duplicate Error	0.33 ± 0	0.26 ± 0	-0.07
Background Error	0.82 ± 1	1.55 ± 2	$+0.73$
Missing Error	5.78 ± 5	1.44 ± 1	-4.34
False Positive (FP) Rate	3.79 ± 3	4.06 ± 4	$+0.27$
False Negative (FN) Rate	7.83 ± 6	3.36 ± 3	-4.47

Table 7. **YOLOv5 analysis** – Comparison of YOLOv5 baseline and optimized results. Metrics are calculated with the **T**oolbox for **I**dentifying **O**bject **D**etection **E**rrors (TIDE) library.

Metric	YOLOv5 Baseline	YOLOv5 Hyp. Opt.	Improvement
AP^{50}	87.52 ± 10	91.73 ± 8	$+4.21$
AP^{55}	86.47 ± 11	90.17 ± 8	$+3.60$
AP^{60}	82.47 ± 11	87.23 ± 9	$+4.76$
AP^{65}	77.68 ± 15	82.79 ± 11	$+5.11$
AP^{70}	72.22 ± 18	76.94 ± 14	$+4.72$
AP^{75}	65.52 ± 19	71.27 ± 15	$+5.75$
AP^{80}	58.35 ± 18	65.77 ± 16	$+7.42$
AP^{85}	49.41 ± 19	59.39 ± 15	$+9.98$
AP^{90}	34.01 ± 16	45.66 ± 13	$+11.65$
AP^{95}	8.29 ± 7	20.15 ± 8	$+11.86$

Table 8. **Influence of IoU threshold** – Comparison of YOLOv5 APs for different IoU thresholds.

results for the optimized hyperparameters suggest greatly improved generalization capabilities. This improvement contributes to better results over all damages. The detection accuracy for the different damage sizes consistently shows the expected behavior that larger objects are detected more accurately than smaller objects. However, the difference in accuracy is very large in some cases. For instance, the difference for the best model between AP^{small} and AP^{medium} is 35.14% points. Even with a good contrast ratio, small holes caused by, e.g., minor cracks, are difficult to separate from sound parts of the mesh. Interestingly, medium-sized climb over defects are detected more robustly than large ones, regardless of the approach. This is due to a lack of training data depicting large climb over defects. In general, it can be concluded that climb over defects are easier to localize due to their position and larger size. In total, a 34.87% points difference in AP between such damages and holes is observed for the best model.

Error sources: So far, the analysis has been conducted quantitatively via the AP. In this section, the TIDE [2] library is utilized to break down the error sources. For this purpose, different error types are presented in the upper part of Tab. 7. The error types describe erroneous relationships of GT BBox and predicted BBox, such as a deviating position or even a missed prediction. The localization and the missing of damages have improved more than average. The improved ability to localize damages may lead to enhanced generalization to other fence types or transfer the learned features to new contexts. The significant reduction in localization error is due to increased AP all Intersection Over Unions (IoUs). The AP results with different IoUs, i.e., varying degrees of overlap with the ground truth BBoxes, are shown in Tab. 8. Thus, for IoU of 0.90 and 0.95 in each case over 11% improvement was obtained. However, in the context of this work, the improvement of the missing damages is more relevant. Exact recognition is not directly necessary, but can of course help with generalization. Although the false positive rate increased slightly by 0.27% points compared to the baseline, it is still at a low of 4.06%. This means there would not be too many false alarms in real-world use. In principle, it is better to detect a few too many holes, which can be rechecked digitally, than to completely forget holes. The latter would jeopardize the airport's approval. The significant improvement in missing damage is accompanied by a decrease in FN rate. This has improved by 4.47% points, implicating enhanced usefulness of the model for real-world fence inspection.

ID	Image Resolution $(pixels)$	Avg. AP	Case 1 AP	Case 2 AP	Case 3 AP
R1	288×384	57.2±19	68.00±4	31.72±7	71.88±2
R2	320×512	65.16±12	71.44±2	48.88±1	75.16±2
R3	416×640	67.26±11	72.61±2	52.34±2	76.83±2
R4	512×768	69.09±12	74.64±3	53.82±2	78.81±2
R5	624×960	70.78±9	74.96±2	**58.63**±2	78.75±1
R6	736×1152	70.86±10	75.49±1	58.09±2	78.99±1
R7	848×1344	**70.98**±11	**76.83**±1	56.21±1	79.91±1
R8	960×1536	70.64±11	76.44±3	55.62±2	**79.87**±1
R9	1136×1728	69.86±10	75.31±2	56.03±0	78.25±1
R10	1248×1920	69.86±12	75.62±3	54.58±4	79.39±1

Table 9. **Influence of image resolution** – Results of experiments image resolution with the HistEqu dataset and YOLOv5. R4 was used in the previous experiments.

4.8. Image Resolution

Previous experiments have been carried out with a fixed spatial resolution of input images. However, higher resolution imagery provides more details, which may be beneficial to the task. The results from various resolutions are presented in Tab. 9. One can observe that the AP increases the larger the images but drops again when the image is larger than 848×1344 pixels (R7). The drop is due to the pre-training with the COCO dataset in a resolution of 1280 × 1280, which expects objects to have a specific size.

4.9. Inference time

For the use of the model on, e.g., mobile robots, it is important to achieve a favorable tradeoff between accuracy and computation time. Fig. 3 compares the inference times of our best YOLOv5 model for different resolutions and

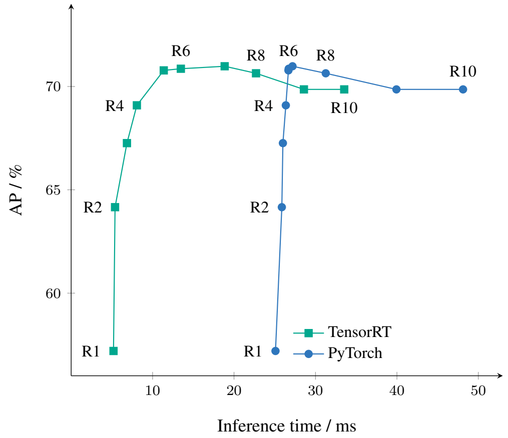
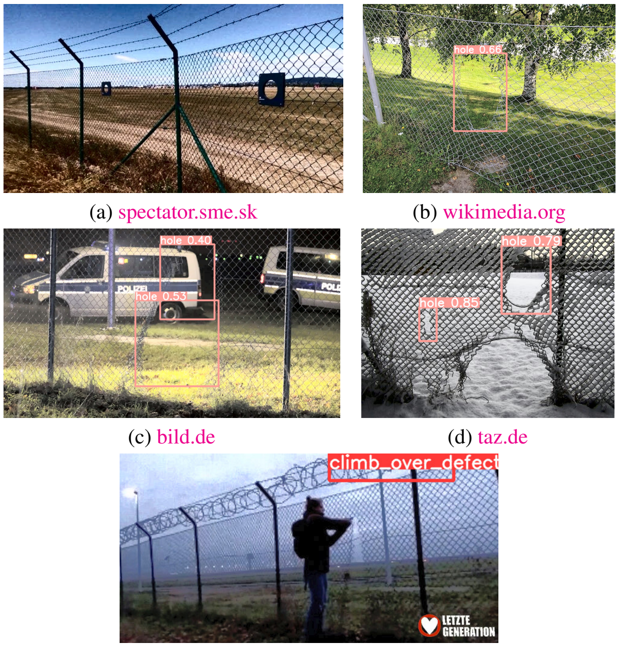


Figure 3. **Inference time** – Comparison between inference time and AP results for varying image resolutions. R1, R2, etc. refer to the ID in Tab. 9. By using TensorRT, all resolutions except R10 are real-time capable. Also a significant acceleration of up to 20ms could be achieved by TensorRT.

with and without the use of TensorRT [5] acceleration. The closer the result to the top left corner, the better. Obviously, TensorRT clearly outperforms PyTorch in inference time. The best tradeoff between speed and accuracy is found for resolution R5 with 624 × 960 pixels. Afterward, inference time increases and accuracy decreases. Noticeably, up to resolution R7, approximately the same inference time is needed with PyTorch. This suggests a computational bottleneck outside the GPU.

4.10. Generalization

As a last step, we evaluate the transferability of our model to further fences, camera models, and weather conditions to identify the strengths and also directions for future research. For this purpose, it is applied to external, freely available images of airport fences. Results are visualized in Fig. 4. As shown in the figure, not all fences have damage. For example, in Fig. 4a, new modules were added to the fences to facilitate photographing through the fences and avoid plane spotters from cutting holes in the fences. Our method does not detect these holes as damages, i.e., it works correctly. Large holes, which are bigger than those included in the dataset, are also correctly detected, as shown in Fig. 4c and Fig. 4b. Two holes are recognized instead of one in Fig. 4c. However, this is no issue in real-world applications, as only the occurrence of damage in a specific location is relevant. In contrast to the aforementioned examples, the hole depicted in Fig. 4e has a different shape and, thus, is not detected by our approach. Future works might consider more variation regarding the shapes of holes included in the training dataset. Furthermore, only two out of three damages are detected in the snowy environment visualized in Fig. 4d. All in all, it can be concluded that the model achieves strong generalization performance to novel image sources. However, training data with increased diversity concerning the shape of damages and weather conditions is required to address the existing weaknesses.



(a) spectator.sme.sk (b) wikimedia.org

(c) bild.de (d) taz.de

(e) geograph.org.uk

Figure 4. **Generalization** – YOLOv5 generalization results on external fence images.

5. Conclusion

Within the scope of the work, the four DL methods YOLOv5, TOOD, VFNet and Deformable DETR were compared to investigate, as a first publication ever, new design rules for airport fence inspection on a small dataset. In conclusion, Deformable DETR as a transformer-based model does not offer any value due to the too-low data volume and the significantly lower accuracy. TOOD and VFNet could achieve higher accuracy with modern SOTA backbones like ConvNeXt and Res2Net, but could not reach the accuracy and the efficiency of YOLOv5. Furthermore, we could show that YOLOv5 also provides good generalization capability on external data.

To improve the accuracy of fence analysis, it would be beneficial to separate the fence from the surrounding context. Although labeling such fine structures is time-consuming, recording with stereo or RGB-D cameras can provide additional information to separate the fence structure from the background. Additionally, a night vision camera can be used for nocturnal inspections, e.g., an infrared camera with higher contrast than its passive counterpart.

[5] https://developer.nvidia.com/tensorrt/, Date: 01/09/2023

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *ArXiv*, abs/2004.10934, 2020. 2

[2] Daniel Bolya, Sean Foley, James Hays, and Judy Hoffman. Tide: A general toolbox for identifying object detection errors. In *ECCV*, 2020. 7

[3] Alexander Burstedde and Filiz Koneberg. Fachkräftemangel im flugverkehr. IW-Kurzbericht 52/2022, Köln, 2022. 1

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 2, 3

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, and et al. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 2, 3

[6] Zehui Chen, Chenhongyi Yang, Qiaofei Li, Feng Zhao, Zheng-Jun Zha, and Feng Wu. Disentangle your dense object detector. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4939–4948, 2021. 2

[7] Stephane Cuenat and Raphael Couturier. Convolutional neural network (cnn) vs vision transformer (vit) for digital holography. In *2022 2nd International Conference on Computer, Control and Robotics (ICCCR)*, pages 235–240. IEEE, 2022. 3

[8] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 764–773, 2017. 4

[9] Deutsche Flugsicherung (DFS). Luftverkehr in deutschland – 2021, 01 2022.
https://www.dfs.de/. 1

[10] European Union Aviation Safety Agency (EASA). Certification specifications and guidance material for aerodrome design (cs-adr-dsn), 03 2022. https://easa.europa.eu. 1

[11] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R Scott, and Weilin Huang. Tood: Task-aligned one-stage object detection. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3490–3499. IEEE Computer Society, 2021. 2, 3, 4, 5

[12] Xin Feng, Youni Jiang, Xuejiao Yang, Ming Du, and Xin Li. Computer vision algorithms and hardware implementations: A survey. *Integration*, 69:309–320, 2019. 2

[13] Shang-Hua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip Torr. Res2net: A new multi-scale backbone architecture. *IEEE transactions on pattern analysis and machine intelligence*, 43(2):652–662, 2019. 3

[14] Divyanshu Gupta, Shorya Jain, Utkarsh Tripathi, Pratik Chattopadhyay, and Lipo Wang. A robust and efficient image de-fencing approach using conditional generative adversarial networks. *Signal, Image and Video Processing*, 15(2):297–305, 2021. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 06 2016. 3

[16] Xudong Jiang. Feature extraction for image recognition and computer vision. In *2009 2nd IEEE International Conference on Computer Science and Information Technology*, pages 1–15, 2009. 2

[17] Glenn Jocher, Ayush Chaurasia, Alex Stoken, and et al. ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference, Feb. 2022. 2, 3, 4, 5

[18] Sankaraganesh Jonna, Sukla Satapathy, and Rajiv R. Sahay. Stereo image de-fencing using smartphones. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1792–1796, 2017. 2

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3, 5

[20] Oğuzhan KIVRAK and Mustafa Zahid GÜRBÜZ. Performance comparison of yolov3, yolov4 and yolov5 algorithms: A case study for poultry recognition. *Avrupa Bilim ve Teknoloji Dergisi*, (38):392–397, 2022. 2

[21] Jonghyeok Lee, Talha Ilyas, Hyungjun Jin, Jonghoon Lee, Okjae Won, Hyongsuk Kim, and Sang Jun Lee. A pixel-level coarse-to-fine image segmentation labelling algorithm. *Scientific Reports*, 12(1):1–18, 2022. 2

[22] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 2, 4

[23] Shujian Liu, Haibo Zhou, Chenming Li, and Shuo Wang. Analysis of anchor-based and anchor-free object detection methods based on deep learning. In *2020 IEEE International Conference on Mechatronics and Automation (ICMA)*, pages 1058–1065. IEEE, 2020. 3

[24] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11976–11986, 06 2022. 3

[25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 3, 5

[26] Takuro Matsui and Masaaki Ikehara. Single-image fence removal using deep convolutional neural network. *IEEE Access*, 8:38846–38854, 2020. 2

[27] Will Nash, Tom Drummond, and Nick Birbilis. A review of deep learning in the study of materials degradation. *npj Materials Degradation*, 2(1):1–12, 2018. 2

[28] Upesh Nepal and Hossein Eslamiat. Comparing yolov3, yolov4 and yolov5 for autonomous landing spot detection in faulty uavs. *Sensors*, 22(2):464, 2022. 2

[29] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *Science and information conference*, pages 128–144. Springer, 2019. 2

[30] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[31] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018. 2

[32] Jonathan Shapiro. Genetic algorithms in machine learning. In *Advanced Course on Artificial Intelligence*, pages 146–168. Springer, 1999. 5

[33] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019. 4

[34] Marco Sozzi, Silvia Cantalamessa, Alessia Cogato, Ahmed Kayad, and Francesco Marinello. Automatic bunch detection in white grape varieties using yolov3, yolov4, and yolov5 deep learning algorithms. *Agronomy*, 12(2), 2022. 2

[35] P Suganya, S Gayathri, N Mohanapriya, et al. Survey on image enhancement techniques. *International Journal of Computer Applications Technology and Research*, 2(5):623–627, 2013. 3

[36] D Vijayalakshmi, Malaya Kumar Nath, and Om Prakash Acharya. A comprehensive survey on image contrast enhancement techniques in spatial domain. *Sensing and Imaging*, 21(1):1–40, 2020. 3

[37] Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018. 2

[38] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-yolov4: Scaling cross stage partial network. *CoRR*, abs/2011.08036, 2020. 3

[39] Chien-Yao Wang, Hong-Yuan Mark Liao, I-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, and Jun-Wei Hsieh. Cspnet: A new backbone that can enhance learning capability of cnn. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1571–1580, 2020. 3

[40] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016. 2

[41] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 3

[42] Guanhao Yang, Wei Feng, Jintao Jin, Qujiang Lei, Xiuhao Li, Guangchao Gui, and Weijun Wang. Face mask recognition system with yolov5 based on image recognition. In *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pages 1398–1404. IEEE, 2020. 2

[43] Guanhao Yang, Wei Feng, Jintao Jin, Qujiang Lei, Xiuhao Li, Guangchao Gui, and Weijun Wang. Face mask recognition system with yolov5 based on image recognition. In *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pages 1398–1404, 2020. 2

[44] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12104–12113, 2022. 3

[45] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 5

[46] Haoyang Zhang, Ying Wang, Feras Dayoub, and Niko Sunderhauf. Varifocalnet: An iou-aware dense object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8514–8523, 2021. 2, 3, 4, 5

[47] Hongbin Zhang, Xiang Zhong, Guangli Li, Wei Liu, Jiawei Liu, Donghong Ji, Xiong Li, and Jianguo Wu. Bcunet: Bridging convnext and u-net for medical image segmentation. *Computers in Biology and Medicine*, 159:106960, 2023. 3

[48] Fangbo Zhou, Huailin Zhao, and Zhen Nie. Safety helmet detection based on yolov5. In *2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA)*, pages 6–11, 2021. 2

[49] Jinjie Zhou, Baohui Zhang, Xilin Yuan, Cheng Lian, Li Ji, Qian Zhang, and Jiang Yue. Yolo-cir: The network based on yolo and convnext for infrared object detection. *Infrared Physics & Technology*, 131:104703, 2023. 3

[50] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9300–9308, 2019. 4

[51] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2, 3, 4

[52] Karel J. Zuiderveld. Contrast limited adaptive histogram equalization. In *Graphics Gems*, 1994. 3