Fakultät für Bauingenieurwesen und Umweltwissenschaften
Universität der Bundeswehr München

Automated Image-Based Damage Recognition for Bridge Inspections

Johannes Flotzinger

Vollständiger Abdruck der von der Fakultät für Bauingenieurwesen und Umweltwissenschaften der Universität der Bundeswehr München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

angenommenen Dissertation.

Vorsitzender:
Prof. Dr.-Ing. Max Spannaus

Prüfende der Dissertation:
1. Prof. Dr.-Ing. Thomas Braml
2. Prof. Dr. Heiko Neumann,
   Universität Ulm
3. Prof. Dr.-Ing. Mathias Obergrießer,
   OTH Regensburg

Die Dissertation wurde am 18.06.2025 bei der Universität der Bundeswehr München eingereicht und durch die Fakultät für Bauingenieurwesen und Umweltwissenschaften am 15.10.2025 angenommen. Die mündliche Prüfung fand am 3.12.2025 statt.

Zusammenfassung

Automatisierte bildbasierte Schadenserkennung, im Englischen: “Automated Image-based Damage Recognition” (*AIDR*), ermöglicht Klassifizierung, Vermessung und Verortung von visuell unterscheidbaren Schäden. Die fehleranfällige und ineffiziente manuelle Erfassung dieser Attribute, wie bei der derzeit praktizierten Prüfung, kann durch diese Automatisierung ersetzt werden. Während einer automatisierten Brückeninspektion werden Bilder in der Regel mit Geräten wie Drohnen oder Smartphones erfasst und anschließend von einem Modul zur Schadenserkennung verarbeitet. Das Herzstück dieses Moduls ist die *AIDR*. Die jüngsten Erfolge von *AIDR* sind auf zwei Entwicklungen zurückzuführen: erhebliche Fortschritte bei der Entwicklung von Bilderkennungsmodellen aus dem Bereich Deep Learning, und das Veröffentlichen von Datensätzen für die automatisierte Schadenserkennung.
Modelle, die auf derartigen Datensätzen trainiert wurden, zeigen häufig begrenzte Generalisierungsfähigkeit gegenüber realweltlichen Anwendungsfällen. Hierbei sind sie mit stark variierender Bildqualität sowie einer größeren Vielfalt an Schadensklassen konfrontiert, als sie in bestehenden Datensätzen abgedeckt wird. In dieser Dissertation werden datengestützte Lösungsansätze für zuverlässige *AIDR* analysiert. Dabei werden Open-Source-Daten genutzt und neue Datensätze basierend auf realen und synthetischen Daten entwickelt. Die wesentlichen Beiträge sind:

- Nutzung von Open-Source-Daten: Durchführung einer umfassenden Optimierung der Trainingspipeline sowie Fusion offener Datensätze, wobei eine unzureichende Generalisierungsfähigkeit gegenüber realen Einsatzbedingungen festgestellt wurde.
- Neue Benchmark: Vorstellung von *dacl10k*, der ersten pixelweise annotierten Benchmark für multi-label Segmentierung bei Brückeninspektionen.
- Bewertung der Genauigkeit und Robustheit von Modellen, die auf *dacl10k* sowie ergänzenden synthetischen Datensätzen trainiert wurden.

In einem breiteren Kontext legt diese Arbeit durch die Nutzung vorhandener Daten, neben Echtwelt-, und synthetischen Datensätzen einen Grundstein für die Entwicklung automatischer, bildbasierter Schadenserkennung.

Abstract

Automated Image-based Damage Recognition (*AIDR*) provides the attributes class, size, and location of visually recognizable defects during bridge inspections. The error-prone and inefficient manual collecting of these attributes, as currently practiced in assessments, can be replaced through this automation. In the context of automated bridge inspection, visual data is typically acquired using devices such as drones or smartphones and subsequently processed by an automated damage recognition module. At the core of this module lies *AIDR*. Recent progress of *AIDR* is primarily driven by two key developments: significant advances in deep learning-based computer vision models, and the emergence of publicly available datasets for automated damage recognition.
However, models trained on such data often struggle to generalize to real-world scenarios, where they must cope with varying image quality and a broader range of defect classes than existing datasets offer. This thesis approaches the challenge of reliable *AIDR* by emphasizing data-driven solutions, incorporating open-source data, and introducing novel real-world datasets and synthetic extensions. The main contributions are:

- Leveraging open-source data: Conducting a sophisticated training pipeline optimization for defect classification using compilations of open-source datasets, revealing limitations in generalization to real-world conditions.
- Novel benchmark: *dacl10k* is presented, the first large-scale, pixel-annotated benchmark for multi-label semantic segmentation in bridge inspection.
- Synthetic data: Evaluating the accuracy and robustness of models trained on *dacl10k* when complemented by synthetic datasets.

In a broader context, this work lays the foundation for the development of application-oriented and reliable automated image-based damage detection by using existing data in combination with real-world and synthetic datasets.

Contents

Zusammenfassung **iii**

Abstract **iv**

Contents **v**

1 Introduction **1**

2 State of the Art **5**
2.1 The role of AIDR in Automated Bridge Inspections 5
2.2 Datasets for Damage Recognition on Built Structures 8
2.3 Deep learning Models for Automated Image-Based Damage Recognition . . . 11
2.3.1 Image Encoders 11
2.3.2 Semantic Segmentation Architectures 12
2.4 Training Pipeline Optimization 15
2.5 Metrics 18

3 Thesis Synopsis **20**
3.1 Leveraging Open-Source Data for Image-Based Concrete Defect Recognition 21
3.2 Developing a Damage Segmentation Benchmark for Automated Bridge Inspection 23
3.3 Investigating Synthetic Data Extensions for Bridge Damage Segmentation . . 25

4 Leveraging Open-Source Data for Image-Based Concrete Defect Recognition **27**
4.1 Building Inspection Toolkit: Unified Evaluation and Strong Baselines for Bridge Damage Recognition 28
4.2 Automated Damage Classification on Concrete Bridges Using Convolutional Neural Networks 33
4.3 dacl1k: Real-World Bridge Damage Dataset Putting Open-Source Data to the Test 42

5 Developing a Damage Segmentation Benchmark for Automated Bridge Inspection **52**
5.1 dacl10k: Benchmark for Semantic Bridge Damage Segmentation 53
5.2 dacl-challenge: Semantic Segmentation during Visual Bridge Inspections . . 76

6 Investigating Synthetic Data Extensions for Bridge Damage Segmentation 86
6.1 synth-dacl: Does Synthetic Defect Data Enhance Segmentation Accuracy and Robustness for Real-World Bridge Inspections? 87

7 Conclusion 105
7.1 Summary 105
7.2 Outlook 106

Bibliography 109

Acknowledgements 123

1 Introduction

Infrastructure systems such as bridges are fundamental to the functioning of modern societies. Yet a significant stock of bridges is old and deteriorated, particularly in countries that experienced substantial economic growth and infrastructure development between the 1950s and 1980s [1, 2]. Many of these buildings are now reaching or exceeding their design life, leading to increased occurrences of defects such as crack, spalling, efflorescence, or corrosion. Several factors contribute to the degradation of infrastructure:

- High traffic loads: Increased usage beyond original design expectations accelerates fatiguing [3, 4, 5].
- Extreme weather events: Climate change has led to more frequent and severe weather events, such as heatwaves and floods [6, 7], which can compromise structural integrity of bridges [8].
- Environmental exposure: Infrastructure in coastal areas is subject to corrosion from sea salt [9], while regions that use deicing chemicals face similar corrosive challenges [10].
- Earthquakes: The energy of the ground motion imposes dynamic loads that can exceed the structure's capacity, causing damage [11].

Due to economic and bureaucratic constraints and a shortage of skilled personnel, governments must maintain these systems efficiently and operate them as long as possible. At the same time, safety is the highest priority, while the burden on the public caused by rehabilitation or inspection measurements must be minimized. Bridge inspections play a critical role in this context, guiding key decisions such as restricting heavy traffic or closing a bridge.

Currently, bridge inspections are typically performed manually by certified civil engineers specialized in structural assessments. Most national bridge inspection guidelines, *e.g.*, from Austria [12], Germany [13, 14] or the UK [15], differ between general and principal inspections, where the principal takes place every six years and the general from once a year to every three years. [16]. The general inspection comprises a visual observation of all parts of the structure that can be assessed without additional access equipment, using ground-level viewing positions around the structure. The principal inspection is a close examination, within touching distance, of all parts of a building [15]. Therefore, depending on the bridge size, additional access equipment such as swing stages, lifts, or rope access is often required. These are examined in a close-up setting because certain critical defects, such as hollow concrete areas, insufficient concrete cover, or generally subtle issues, can only be detected acoustically, with additional sensors, or from a short distance. The following outlines the currently practiced methodology of principal bridge inspections. The description is exemplified based on the German standard; however, it is essential to note that similar methods and documentation practices are used internationally [16].

Once a defect is detected, the inspector takes a photograph and documents key attributes such as the type, size, and location of the damage. This information is manually recorded in a damage sketch. An example is shown in Figure 1.1, where defects are numbered sequentially. For instance, entry #32 refers to a crack ("Riss") with a width of 0.2mm, located approximately 7 meters from the cross girder ("Querträger"), between the left side of the web and flange on the leftmost T-beam (cross-section 4–5). Based on the according damage parameters, standardized defect IDs – as defined by German regulations – are assigned to every instance. For each ID, national guidelines provide recommended ratings for structural safety, traffic safety, and durability. These are adjusted by the inspector using their professional judgment. Finally, the overall condition of the bridge is computed taking all defect ratings into account. This assessment is expressed as a condition grade (analogous to a school grade). Consequently, the inspection report is generated. It includes a comprehensive list of defects, their respective attributes, grades, and recommendations for maintenance actions such as repair, load restriction, or replacement. The defect sketch is appended to the report to enable a visual defect localization [13, 14].

Several steps in the analogue inspection process – such as defect classification, measurement, localization, and evaluation – are prone to inconsistency, subjectivity, and error. These limitations are largely attributable to the manual and fragmented nature of documentation. A study by Phares *et al.* [17] illustrates the variability of inspection outcomes. The researchers analyzed assessments from 49 inspectors across 25 state departments of transportation (DOTs) in the United States, who each assessed one of six bridges. Their findings revealed that approximately 56% of the reported condition ratings deviated significantly from the established reference ratings which were considered as ground-truth. A major source of this inconsistency was the lack of uniformity in notes and supporting imagery. In many cases, critical details or photographs of key defects were missing.

The consequences of deteriorating infrastructure, together with the traditional inspection method, pose significant safety, economic, and environmental risks. Failures can lead to increased maintenance costs, service disruptions, and – in the worst case – abrupt failures. The building inspector is responsible for recommending rehabilitation measurements and closing buildings before tragedies can happen, while being confronted with an analog, error-prone, and inaccurate inspection procedure.

The key factor in improving the existing procedure, independently of the used device and the inspector-machine, or human-machine, interaction, is *Automated Image-Based Damage Recognition* (*AIDR*). *AIDR* enables the efficient extraction of key defect attributes on image level, namely class, size, and location. These are further converted to physical units through depth measurement and localized via pose estimation techniques. This approach not only enhances the accuracy and repeatability of damage assessment but also relieves inspectors from the burden of manual damage capturing and documentation, allowing them to focus on interpreting and evaluating defects. Models from the field of deep learning (deep neural networks) are ideally suited for robust *AIDR* because they can learn to identify complex visual patterns associated with bridge defects by training on labeled image data. They build up understanding in layers, from detecting simple textures and edges to recognizing specific damage types. This enables accurate and efficient defect detection even under varying environmental conditions and concrete textures. To detect concrete spalling, for example, a deep learning model might first recognize edges and textures, then notice rough, roundish areas

with exposed aggregates, and finally conclude that these features indicate spalling. Each layer adds more meaning, helping the model understand the defect like a bridge inspector.

One possible approach to advancing *AIDR* is model-centric, *i.e.*, by developing novel and more powerful network architectures. However, observing leaderboards [19, 20, 21], this path appears unpromising due to stagnating progress on image recognition benchmarks, where improvements are often limited to marginal gains measured in fractions of a percentage point. In contrast, datasets in the field of *AIDR* remain scarce and limited – especially compared to other domains [22, 23]. Concerning practical usage, the existing data lacks diversity of image quality, environmental conditions, and the variety of defect classes that must be recognized in accordance with inspection standards [24, 25, 26, 27]. Therefore, it is essential to approach *AIDR data-centric*, rather than focusing on the already well-explored field of model optimization. In other words:

> *Initially, data will solve the problem.*

Guided by this central hypothesis, the underlying work contributes to addressing the challenge of robust *AIDR* for automated bridge inspections through three main pillars: *(i)* leveraging open-source data, *(ii)* creating an own damage segmentation benchmark, and, *(iii)* augmenting this benchmark with synthetic data to investigate its impact on accuracy and robustness.

The two subsequent chapters provide the relevant state of the art and a detailed synopsis of the research, followed by the three major chapters presenting the main contributions. This cumulative thesis includes six peer-reviewed publications[1]. The final chapter concludes the dissertation and offers perspectives for future research in the field of *AIDR* for automated bridge inspection.

[1] The manuscript "synth-dacl: Does Synthetic Defect Data Enhance Segmentation Accuracy and Robustness for Real-World Bridge Inspections?" is cited as a preprint; however, it has been accepted for publication at the DAGM German Conference on Pattern Recognition 2025.

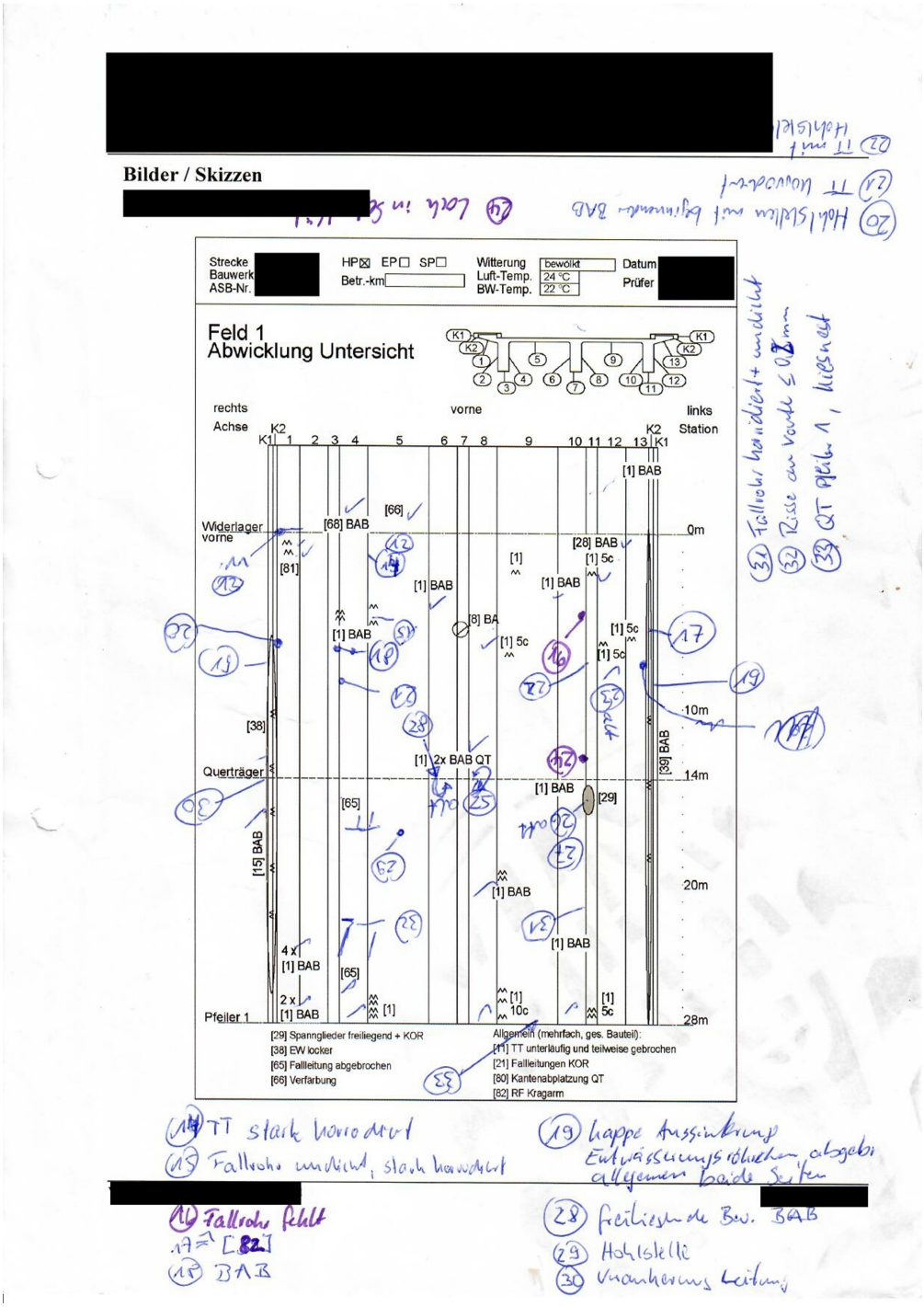


Figure 1.1: Damage sketch from a real bridge inspection with three T-beams. The sketch represents the bridge flipped open along its longitudinal axis. Thus, the "vertical lines" represent the edges of the bridge cross-section. The defects are numbered in chronological order. Exemplary damage information separated by a comma (damage number, damage class, longitudinal, and cross-section dependent localization): #28, *Spalling* ("BAB") with *Exposed Rebars* ("freiliegende Bewehrung"), at the cross girder ("Querträger"), on the left side of the web (6); #29, *Hollowarea* ("Hohlstelle"), approximately 3m from the cross girder ("Querträger"), on the flange (5); #31, *Drainage* ("Fallrohr") with *Rust* ("korrodiert"), which is leaking ("undicht"), approximately 7m from the cross girder ("Querträger"), on the bottom of the web (11) [18].

2 State of the Art

This section provides an overview of related work and key terminologies essential for understanding the concepts discussed in the main chapters.

2.1 The role of AIDR in Automated Bridge Inspections

Bridge inspection automation refers to integrating sensors, vehicles, and digital tools into the inspection workflow to reduce or eliminate manual effort. Depending on the level of human-machine interaction, three stages of increasing system intelligence and autonomy can be distinguished [28, 29]. The first two stages align with the inspection types noted in parentheses, as defined in relevant standards [13, 14, 15]:

- *Automatic Inspection (principal+general)* involves systems that can carry out inspection steps in sequence but require human input or human trigger at certain stages (human in the loop). This can be exemplified by an inspector using a smartphone with an app enabling automated damage recognition: the inspector captures an image of a defect, which the app then analyzes and documents, including the damage type, size, and location.
- *Automated Inspection (general)* systems execute the inspection task independently, based on predefined instructions (human on the loop). For instance, a drone can follow a programmed flight path to capture inspection data and feed it directly into an automated damage recognition module, requiring no real-time human control.
- *Autonomous Inspection (-)* denotes the most advanced level, in which the system interprets its environment and makes decisions independently. For instance, a drone could launch at scheduled intervals, survey the entire bridge surface, and identify all defects – without any human intervention or predefined flight paths.

Various devices such as drones [30, 31, 32, 33], robots [34, 35, 36, 37, 38], and smartphones [39, 40] have been introduced to automate building inspections. The core task remains the same: classifying, measuring, and localizing defects based on image data (*AIDR*).

The full inspection workflow[1] involving *AIDR*, regardless of the human-machine interaction and utilized device, is displayed in Figure 2.1. First, a model based on Building Information Modeling (BIM model) [42] is created, representing the foundation of the assessment. During inspection, images or video frames, acquired on site, are processed by a

[1] In earlier works – namely *dacl1k: Real-world bridge damage dataset putting open-source data to the test* [41] and *dacl10k: Benchmark for Semantic Bridge Damage Segmentation* [18] – this workflow was called "digital inspection" or "digitized inspection".

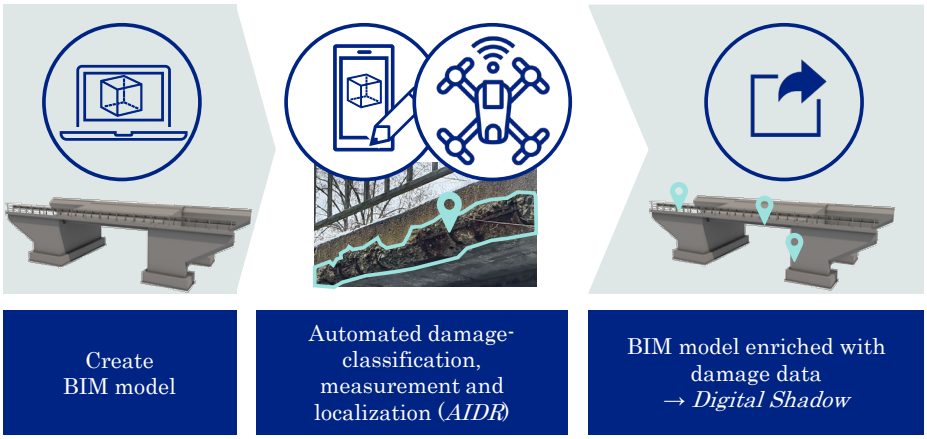


Figure 2.1: Inspection process of a bridge utilizing pixel-level damage recognition (*AIDR*) [40].

module for the automated damage recognition, which performs classification, localization, and measurement of visually detectable defects. The fundament of the automated damage recognition module is the *AIDR*, incorporating multi-label defect segmentation, assigning each pixel to one or more damage types. When combined with two additional elements, the depth measurement [43, 44, 45] and pose estimation (*e.g.*, SLAM [46, 47, 48]), the module enables the determination of defect size and location within the BIM model. Based on these damage parameters, the captured defect is automatically mapped to a corresponding entry in the country-specific defect catalog defining the guideline-conform classification and grading recommendations [49]. Finally, the damage properties are stored and located in the BIM model. This model constitutes a digital representation of an existing or planned physical asset – referred to as digital model. When this model is continuously updated through an automated, one-way data flow from the bridge to its digital counterpart, it qualifies as a *Digital Shadow* [50].

The goal of this assessment is to support inspectors throughout and after the general and principal inspection, while staying compliant with existing standards. Compared to analog traditional workflows, it raises consistency, accuracy, and reliability of inspections by automating defect detection and eliminating documentation in a manual sketch. Furthermore, it allows the inspector to easily track changes in the bridge's condition over time, visualized in a 3D model; instead of checking paper sketches and reports from past years against each other. In an *Automatic Inspection* context, where the inspector is the central decision-maker, the *AIDR* provides real-time defect suggestions that the inspector may review, adjust, and approve on-site. This human supervision is particularly important for defects that require broader contextual interpretation. *Automated Inspection* and *Autonomous Inspection* can significantly reduce or even eliminate human workload, enabling more frequent building observations—either by filtering out non-damaged regions for targeted follow-up inspections by humans, or by performing fully autonomous assessments of the asset. In summary, compared to the currently practiced analog workflow, utilizing automated damage recognition in combination with a continuously updated BIM model – or rather, a *Digital Shadow* – significantly improves efficiency and accuracy. Moreover, it facilitates the interpretation of interactions and the evolution of defects over time. In the following, the terms "automated

inspection" and "automated bridge inspection" will be used to refer to any bridge inspection type that employs *AIDR*.

2.2 Datasets for Damage Recognition on Built Structures

A deep learning model's ability to generalize to real-world tasks is fundamentally constrained by the quality and diversity of its training data. As the machine learning pipeline's first and most critical component, the dataset determines the model's learned representations' scope, transferability, and robustness. Without representative and high-quality data, even the most sophisticated network architectures are likely to fail under practical conditions. *AIDR* of concrete defects has made significant advances through the development and release of publicly available datasets. These datasets vary in scope, task, label setting, number of labels, and size (see Table 2.1). The following paragraphs review the current state of datasets in this domain, organized by the primary tasks they support: classification, object detection, and semantic segmentation. Finally, details on synthetic crack segmentation datasets are provided.

Table 2.1: Overview of publicly-available datasets for automated damage recognition and the according task, label setting, amount of classes, number of images, and publishing year. The setting can be binary (B), single-label (SL), or multi-label (ML). The task can be classification (C), object detection (OD), or segmentation (S).

Dataset	Task	Setting	Classes	Size	Year
SDNET [51]	C	B	2	13,620	2018
CDS [52]	C	SL	4	1,028	2018
BCD [53]	C	B	2	5,390	2019
MCDS [54]	C	SL	8	3,607	2019
CODEBRIM [27]	C/OD	ML	6	7,261	2019
S2DS [24]	S	SL	6	743	2022
CrackSeg9k* [55]	S	B	2	9255	2022
Bochum Crack Data Set [56]	S	B	2	370	2022
dacl1k [57]	C	ML	6	1,474	2023
Supervisely [58] (synthetic)	S	B	1	1558	2023
dacl10k [59]	S	ML	19	9,920	2023
He et al. [60]	S	SL	4	1203	2024
Cracktal [61] (synthetic)	S	B	1	∞**	2024
OMNICRACK30K* [25]	S	B	1	$\approx$30,000	2024

*Dataset collection, **plus 420 samples in real-world test set

Classification Until 2022, classification was the most established task in the field of concrete defect recognition. Datasets have been proposed for binary, multi-class (single-label), and multi-label classification – each reflecting varying degrees of realism, regarding how well they replicate real-world conditions. This distinction is important, as many datasets were created under controlled laboratory settings, often failing to capture the complexity and variability in actual inspection environments.

Binary classification datasets typically distinguish between cracked and non-cracked concrete surfaces. Two of the most prominent original datasets in this category are: BCD [53],

targeting bridge cracks under standardized imaging conditions, and SDNET [51], containing cracks on walls, decks, and pavements. Due to its noisy labels, a cleaned version is commonly used [62]. These datasets are limited in diversity, especially in terms of surface materials, defect types, and real-world noise factors. Nevertheless, they remain helpful in benchmarking low-complexity classification approaches.

Single-label or rather multi-class datasets label images with a single dominant defect. In contrast, multi-label datasets account for multiple simultaneous defects per image, which is often the case for bridge inspection applications. Two cornerstone datasets are MCDS [54, 62] and CODEBRIM [27]. MCDS is a multi-class dataset originally structured for a three-stage pipeline and later consolidated into an eight-class single-stage dataset [63, 62]. Classes include crack, spalling, efflorescence, exposed rebar, corrosion, scaling, and other minor defects. CODEBRIM is designed for multi-label classification and was collected under varied real-world conditions. It includes six damage types and background, based on 7,729 image patches from 30 bridges. Although realistic in appearance, it suffers from limitations due to its patch-based structure: (*i*) unusually shaped patches (e.g., very small or elongated), and (*ii*) lack of contextual information due to cropping.

Object Detection Datasets for object detection are rare in the *AIDR* domain. CODEBRIM remains the only primary open-source dataset to include bounding box annotations for multiple damage types. It provides 5,354 instances across 1,590 images. The labeling follows a three-step process consisting of manual selection of defect bounding boxes, assigning classes to each box, and adding patches showing undamaged areas. Patches vary in scale and were acquired using multiple camera systems, including UAVs for inaccessible areas. Bounding boxes provide only an approximate localization of defects, often leading to inaccuracies – particularly when defects are irregularly shaped or oriented diagonally relative to the image's main axis.

Semantic Segmentation In contrast to object detection, semantic segmentation enables pixel-level measurement. For instance, this is mandatory for determining the crack width and to exactly calculate the needed repair material for rehabilitation, *e.g.*, the volume of repair mortar to fill spallings. Several recent datasets address this challenge, but each has trade-offs in terms of realism, diversity, and coverage. S2DS [24] is the first single-label segmentation dataset specifically for bridge inspections. It includes 743 image patches (1024×1024 px) extracted from 8,435 photographs taken using DSLR cameras, mobile phones, and UAVs. Annotations for six classes (crack, spalling, corrosion, efflorescence, vegetation, and control points) were created with high precision. The best-performing model (hierarchical multi-scale attention) achieves 92% mIoU (see Chapter 2.5). CrackSeg9k [55] comprises 9,255 images focused exclusively on crack segmentation. Compiled from ten sub-datasets, it applies image processing to address labeling inconsistencies. Although extensive, it lacks annotations for non-crack defects. He et al. [60] introduced a dataset of 1,203 images with pixel-wise labels for the damage types, crack, spalling, rebar exposure, and pothole. However, only 13 images show more than one damage type, limiting its utility for multi-label segmentation.

Synthetic Crack Segmentation Synthesized data has emerged as a powerful tool to supplement real datasets, particularly where pixel-accurate labels are expensive or impractical

to obtain [64, 65, 66]. The first openly accessible crack segmentation dataset [58] comprises 1,558 synthetic images, primarily derived from web-scraped photographs of concrete and asphalt surfaces. The synthetic cracks were generated using a combination of random walk algorithms [67], rapidly-exploring random trees (RRT) [68], and L-systems [69]. To enhance realism, style transfer techniques were subsequently applied to the images.

Jaziri et al. [61] presents "Cracktal", a physics-based simulator designed to generate high-resolution images of cracked concrete surfaces, along with corresponding semantic annotations, depth, and surface normal maps. The simulation pipeline consists of two main stages: physically-based scene generation using Blender's rendering engine [70] and real concrete textures, and fractal-based crack generation. The scenes replicate concrete surfaces photographed from a short distance, not showing any other backgrounds (*e.g.*, bridge building parts, sky *etc.*). To achieve realistic rendering, concrete textures from the CODEBRIM training dataset are manually decomposed into physics-based texture maps (albedo, normal, roughness, and height). The main steps of the following physics-based rendering comprise: (*i*) the application of texture maps to a plane mesh grid, (*ii*) adding natural lighting conditions with Blender's black body radiator, and, optionally, (*iii*) mapping occlusion details like graffiti or weathering. Cracks are synthesized via iterative fractal transformations, using randomized displacement magnitudes and angles to mimic natural irregularity, inspired by a stochastic variant of the Koch snowflake model presented in [71]. These cracks are then composited into the scene material, blurred for realism, and rendered. The simulator provides rich, fully annotated data suitable for training models for real-world concrete crack segmentation.

Despite progress, existing datasets cover 1–7 key concrete defect types, whereas engineering standards [12, 13, 14, 15] require capturing at least 9 visually distinguishable damage forms[2]. Especially the segmentation datasets lack in diversity questioning their value for practical use. This highlights the overarching hypothesis that the problem of *AIDR*, firstly, must be tackled data-centric as existing data is insufficient for real-world inspections.

[2]Crack, Efflorescence, Honeycombing (Rockpocket), Hollowarea, Spalling, Restformwork, Wetspot, Corrosion, Weathering

2.3 Deep learning Models for Automated Image-Based Damage Recognition

In the following, material deep learning models for damage classification and semantic segmentation – utilized in subsequent chapters – are presented in the context of the evolution of image recognition. This also includes an overview of key benchmark datasets that have shaped model development and enable performance comparisons.

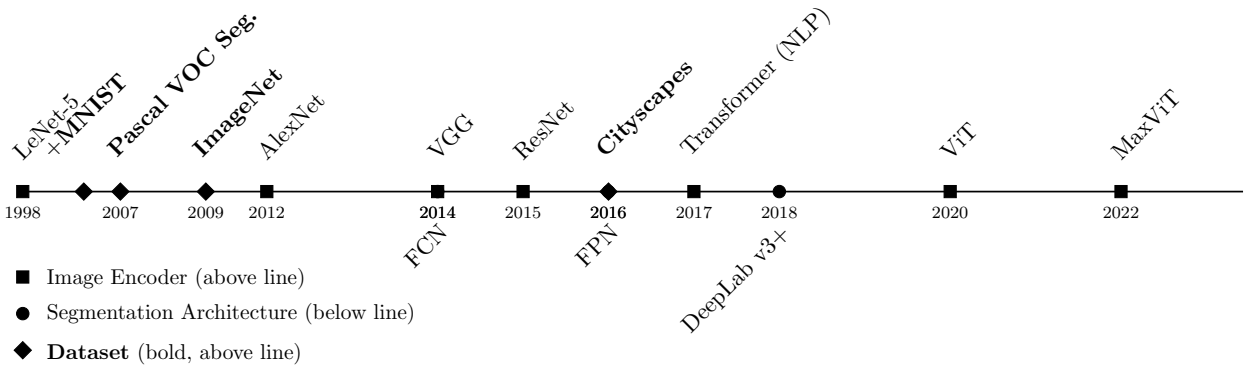


Figure 2.2: Timeline of key deep learning models and datasets for image recognition.

2.3.1 Image Encoders

The timeline in Figure 2.2 begins with one of the most influential groundworks in the evolution of DL-based Computer Vision and Convolutional Neural Networks (CNNs) itself: LeNet-5 and the accompanying dataset MNIST by LeCun *et al.* [72]. LeNet is the first end-to-end trained CNN using backpropagation. Introduced in 2009, ImageNet [73] quickly became the most widely used benchmark for image classification tasks. However, it was the advent of deep CNNs, particularly AlexNet [74], that catalyzed a paradigm shift. AlexNet's victory in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 [75] marked the beginning of the deep learning era in computer vision, achieving unprecedented accuracy with the use of GPUs for large-scale training. Building on this foundation, architectures such as VGG[76] and ResNet[77] further advanced the field by significantly increasing network depth, enabling greater representational power and performance.

ResNet While VGG is a simple, sequential architecture stacking convolutional layers with increasing depth, ResNet introduced the residual block. This block includes skip connections that route the input activations around one or more convolutional layers and directly add them back to later layers. Such connections ensure that features extracted in earlier layers are not lost in the depth of the network, also called the vanishing gradient phenomenon [78, 79]. In the current leaderboard for the standard benchmark ImageNet-1k and the constraint of "Models not using extra training data" [19], ResNet-152 achieves 82.4% top-1-accuracy, reported by Wightman *et al.* [80]. While keeping the architecture unchanged, they entirely relied on "novel optimization and data-augmentation" [19], or rather *training pipeline optimization*. For comparison, the currently best model performance is 88.3% by a

vision transformer from 2021 [81]. This indicates that in over six years, the top performance has improved by only around 6%, with no new state-of-the-art surpassing it since 2021. Consequently, ResNet is still a widely used baseline in image recognition [82, 83, 84]. It also achieved best metrics on open-source data for multi-label classification of concrete defects in *Building Inspection Toolkit: Unified Evaluation And Strong Baselines For Bridge Damage Recognition* [63], which is part of Chapter 4.

MaxViT After the era of deep CNNs, the next major architectural innovation came with the adaptation of the transformer, initially developed for natural language processing [85], to vision tasks. The Vision Transformer (ViT) [86] demonstrated that pure attention-based models could match or exceed CNN performance on large-scale image datasets. Unlike CNNs, which operate on local receptive fields, transformers process image patches globally using a self-attention mechanism that computes relationships between all image patches to enable better modeling of long-range dependencies. This shift gave rise to a new family of hybrid vision models, which combine convolutional and attention mechanisms like the multi-axis vision transformer (MaxViT) [87]. One of MaxViT's central features is the use of multi-axis attention, which couples window-based attention for local context with grid-based attention for capturing global interactions. This allows the model to maintain computational efficiency while modeling both short- and long-range dependencies. Its hierarchical architecture—progressively reducing spatial resolution while increasing channel depth—mirrors the structure of traditional CNNs, leveraging convolutional inductive biases such as locality and translation equivariance while incorporating the flexibility of self-attention. Furthermore, the hierarchical block structure makes it well-suited to be integrated as a backbone in downstream tasks such as semantic segmentation. MaxViT combined with Feature Pyramid Network [88] was used by the best non-ensemble approach in the *dacl-challenge* [89] (see Chapter 5), and consequently used in the subsequent work *synth-dacl: Does Synthetic Defect Data Enhance Segmentation Accuracy and Robustness for Real-World Bridge Inspections?* [90] (see Chapter 6).

Today, both CNNs [91, 92], pure transformer-based [93], and hybrid architectures [94] coexist in the image recognition landscape, with CNNs remaining highly efficient and robust. At the same time, transformers offer superior scalability and global context modeling, especially when trained on large datasets.

2.3.2 Semantic Segmentation Architectures

Before 2014, most semantic segmentation methods relied on sliding-window techniques [95, 96]. In these approaches, a fixed-size window (e.g., 32×32 or 64×64 pixels) is moved across the image, and each extracted patch is fed into a model consisting of fully connected layers to predict the class of its center pixel. This process continues until every pixel in the image has been labeled. However, sliding-window inference is computationally intensive, and each CNN prediction is limited to the contents of a single patch, restricting the available contextual information. In 2014, Long et al. introduced Fully Convolutional Networks (FCN) [97], replacing fully connected layers with convolutional ones so that an entire image can be processed in a single forward pass. FCN converts the final classification layers into 1×1 convolutions (*i.e.*, spatially local filters operating across the whole channel depth), producing a coarse feature map that is then upsampled via learnable deconvolution layers

to yield a full-resolution segmentation. By eliminating patch-wise inference as in the patch-wise approaches, this architecture drastically reduces redundant computation and enables end-to-end learning of pixel-wise labels. Moreover, FCN incorporates skip connections from intermediate layers, combining high-resolution, low-level features with deep, semantic ones to recover spatial detail lost during downsampling.

The first widely adopted benchmark dataset featuring pixel-level annotations was introduced in 2007 as part of the PASCAL Visual Object Classes (VOC) Challenge's Segmentation task. Today, the updated PASCAL VOC 2012 dataset [98] remains in use for evaluating semantic segmentation models. In 2016, Cityscapes [22] was released for urban scene understanding in the context of autonomous driving. Together with other benchmarks like COCO [99] and ADE20k [100], these datasets continue to serve as reference points for evaluating and comparing semantic segmentation methods.

Feature Pyramid Network In 2016, the Feature Pyramid Network (FPN) [88] was proposed as a key advancement for multi-scale feature representation. The core idea of FPN is to leverage the hierarchical structure of convolutional encoders (such as ResNet or MaxViT) by extracting features at multiple spatial resolutions along the encoder's bottom-up pathway. Early stages capture low-level features like edges and textures, while deeper layers encode high-level semantic concepts such as object parts or complex structures. This is particularly well-suited to the task of *AIDR*, where some defects (e.g., cracks, cavities) depend on fine-grained low-level features, while others (e.g., spalling with exposed reinforcement and corrosion) require broader contextual understanding. FPN addresses this by creating a top-down pathway that progressively upsamples high-level feature maps and combines them with corresponding lower-level features via lateral connections. This fusion process results in a feature pyramid in which each level has strong semantic information and fine spatial detail. The ability of FPN to integrate features across scales makes it highly effective for tasks involving local and global patterns. FPN showed promising results in the underlying works: *dacl10k* and *dacl-challenge* (see Chapter 5), as well as *synth-dacl* (see Chapter 6).

DeepLab The DeepLab family of models represents a progression of increasingly effective semantic segmentation architectures. DeepLab v1 [101] introduced the idea of using atrous (or dilated) convolutions to increase the receptive field of convolutional layers without reducing the spatial resolution of feature maps. This allowed the model to retain fine spatial details while incorporating broader contextual information. With the second version of DeepLab [102], Atrous Spatial Pyramid Pooling (ASPP) was added, which aggregates multi-scale contextual information via parallel atrous convolutions with varying dilation rates. The third generation refined the ASPP by including batch normalization and image-level features, enhancing its ability to capture global context. Finally, DeepLab v3+ [103, 104] extends the DeepLab family by combining the Atrous Spatial Pyramid Pooling (ASPP) module with an explicit encoder–decoder structure to enhance boundary refinement. DeepLab v3+ is the current leader on the PASCAL VOC 2012 test set [20] and was therefore investigated in Chapter 5.

While both DeepLab v3+ and architectures like FPN are effective for multi-scale feature learning, they differ fundamentally in approach. FPN constructs a multi-scale feature pyramid through top-down fusion of features at different resolutions, whereas DeepLab v3+

emphasizes semantic richness by leveraging parallel dilated convolutions and context aggregation, prioritizing scale-aware semantics over resolution hierarchy.

2.4 Training Pipeline Optimization

In supervised tasks, training deep learning models aims to find the parameters (model weights) that minimize the loss function on the training data. Such model optimization is typically performed using gradient-based methods such as stochastic gradient descent or Adaptive Moment Estimation (Adam) [105, 106], where the gradients of the loss with respect to the weights are computed using partial derivatives and the chain rule of calculus. Training pipeline optimization aims to find the perfect setup of the factors that influence model optimization while indirectly ensuring that the model generalizes to unseen (validation) data, which is also referred to as Regularization [107]. The key factors relevant to the present study are discussed in the following sections.

Hyperparameter Tuning In the context of deep learning, hyperparameters refer to the higher-level configuration settings of the training pipeline. They are set by the practitioner, or another learning algorithm [108], rather than learned from data (model weights). The setting of hyperparameters must be defined before the training [107]. They play a crucial role in shaping the model optimization process and its regularization. Hyperparameter tuning is the process of systematically searching for the combination of these parameters that efficiently maximizes a model's performance, as measured by validation loss or a task-specific metric (*e.g.*, accuracy, F1 score). Key hyperparameters that have a significant impact on model training include the learning rate, batch size, and number of epochs [109]:

- Learning Rate: Determines the intensity with which the weights are changed for each iteration of gradient descent, while moving toward a minimum of the loss function during training [107]. A learning rate that is too high may prevent the model from reaching the global minimum, while a rate that is too low can result in slow convergence and getting stuck in local minima.
- Batch Size: Regulates the number of training examples considered to compute the gradients for one weight update step [110]. Smaller batch sizes offer more granular updates and can improve generalization, but may lead to noisier gradients and longer training times. Larger batch sizes can make training faster and more stable, but require more memory and may reduce generalization performance.
- Number of Epochs: Indicates how many complete passes over the training dataset are performed [110]. Too few epochs can lead to underfitting [3], while too many can cause overfitting [4], where the model memorizes the training data instead of regularizing.

Transfer Learning In many real-world cases, including *AIDR* for bridge inspections, gathering data for training deep learning models is costly and challenging. As a result, it becomes essential to develop high-performing models using readily available data or pretrained models from the same, familiar, or different domain. The central point in transfer learning is adjusting the model's initial feature space (patterns learned from the source dataset) to the

[3] Underfitting occurs when the model fails to learn patterns from the training data.

[4] Overfitting occurs when the model learns the training data – often too precisely, including noise – and performs poorly on unseen validation data.

data distribution characteristics of the target dataset. The "adjusting" is necessary owing to the usually present domain shift resulting from differences in feature distribution and marginal distribution, exemplified in the following paragraph [111].

The de facto standard source dataset for image classification is ImageNet [73]. In the context of anomaly detection across various domains, including healthcare [112], agriculture [113], and automotive [114], using ImageNet has been shown to significantly enhance convergence speed and model accuracy. In the context of *AIDR* for bridge inspection, the data distribution of ImageNet has also proven beneficial [115]. When leveraging an ImageNet-pretrained model for a specialized task such as multi-label damage classification in concrete bridges, *e.g.*, the CODEBRIM dataset, the question arises as to which layers of the model require adaptation and to what extent. This can also be referred to as the search for the transfer learning or fine-tuning strategy. In the underlying case, the marginal and feature distributions between source and target tasks differ significantly: ImageNet involves 1000 classes in a single-label setting, while CODEBRIM comprises six classes in a multi-label setup. Additionally, the semantic gap in features – e.g., from goldfish to exposed reinforcement – is substantial. Owing to the difference in class count between the source and target tasks, the entire classifier head of the pretrained model is typically replaced with a new set of fully connected layers, designed to match the class structure of the CODEBRIM dataset in the output layer. This head is typically initialized using the Kaiming uniform distribution [116] and must be trained intensely. Moreover, performance often improves when adapting the model base, also called image encoder. The encoder contains weights derived from the source dataset that are valuable for the target task. The early layers, which extract low-level, domain-invariant features such as edges, corners, and textures, often generalize well and may require little to no adaptation. In contrast, deeper layers encode higher-level, task-specific representations [117] and generally benefit from more extensive fine-tuning. Several established strategies have emerged for fine-tuning:

- Feature extraction: The most computationally efficient approach involves freezing the entire pretrained base and training only the classifier head. This strategy significantly reduces the number of trainable parameters and has performed remarkably well, especially with limited target-domain data [118, 110].

- Sequential fine-tuning: A more adaptive strategy starts by training the classifier head as above, followed by unfreezing and jointly fine-tuning a subset of deeper layers in the base. This approach is particularly effective when the target dataset is moderately sized or exhibits considerable distributional shift from the source domain [110].

- Discriminative fine-tuning: This technique uses layer-wise learning rates—typically assigning lower rates to early layers and higher rates to later or newly added layers. Unlike sequential fine-tuning, discriminative fine-tuning is performed in a single training phase, offering a balance between performance and efficiency [119, 120].

These strategies offer varying trade-offs between computational cost and model expressiveness. The optimal choice depends on factors such as the size of the target dataset, domain similarity, complexity of the target task, and model architecture.

Data Augmentation Data augmentation is a widely used technique in image recognition to artificially increase the diversity and size of a training dataset by applying various

label-preserving transformations to the input images. This process helps improve model generalization and robustness, particularly when training data is limited or prone to overfitting. Common augmentation techniques include geometric transformations (e.g., horizontal flipping, rotation, cropping, scaling, translation), color-based modifications (e.g., brightness, contrast, saturation, hue adjustments), and more advanced methods such as Cutout, Mixup, and CutMix. Furthermore, several automatic augmentation methods dynamically modify training samples during the learning process. These methods can apply transformations either randomly – such as in RandAugment [121] and TrivialAugment [122] – or adaptively based on performance feedback. The latter approach involves learning augmentation policies using reinforcement learning or optimization strategies, as seen in AutoAugment [121] and Fast AutoAugment [123], where augmentation decisions are guided by validation metrics to maximize model performance.

Balancing Datasets Dealing with imbalanced data is a common challenge in real-world tasks such as *AIDR*, where certain defect types like honeycombing (*Rockpocket*) or concrete dissolving (*Washouts/Concrete corrosion*) appear far less frequently than others [18]. Two primary strategies are widely used to address this issue. The first is oversampling, which involves presenting underrepresented classes more frequently during training. The second is weighted loss, where the training objective is adjusted by assigning higher loss weights to minority classes, encouraging the model to pay greater attention to rare defects. Both approaches aim to mitigate bias toward dominant classes and improve the model's ability to detect and classify infrequent but important damage patterns. An additional strategy involves synthetically generating samples of underrepresented defect classes, thereby enhancing data diversity while reducing the reliance on labor-intensive manual labeling. This can be achieved through semi-synthetic approaches, like in the work of Dwibedi *et al.* [124], where object instances are cropped and composited onto new backgrounds, or through fully synthetic methods using Generative Adversarial Networks (GANs) [125].

2.5 Metrics

The evaluation of models in the underlying work relies on standard metrics used in classification and semantic segmentation tasks, which are introduced below.

To quantify how many defects were missed, the Recall metric is computed:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.1}$$

where TP (true positives) represents correctly detected defects, and FN (false negatives) denotes missed defects. High recall is critical in safety-relevant applications like structural inspections, where overlooking damage may have serious consequences.

In contrast, false positives – incorrectly predicted defects – are a common challenge, too. These can undermine trust in automated inspection systems due to additional work for the customer of *AIDR* applications who has to sort out these false positives, which is why Precision is also reported:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.2}$$

Precision reflects the proportion of predicted defects that are actually correct. It is reduced as the number of false positives increases, setting the true positives in relation to prediction quality. This metric is crucial for decision-makers and public authorities responsible for commissioning and evaluating automated inspection tools.

To balance both perspectives (Recall and Precision), the F1 score, which is the harmonic mean of precision and recall, is used:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.3}$$

This combined metric provides a single measure of a model's accuracy in detecting actual damage and avoiding false alarms.

The Exact Match Ratio (EMR) is commonly used for multi-label classification, which was utilized for damage detection in the work on CODEBRIM [27]. It is defined as:

$$EMR = \frac{1}{n} \sum_{i=1}^{n} I(y_i = \hat{y}_i), \tag{2.4}$$

where n is the number of samples and I is the indicator function, which returns 1 if the predicted set exactly matches the true set and 0 otherwise. y_i is the ground-truth and $\hat{y}_i$ is the predicted label. EMR rewards only completely correct predictions, making it a stringent metric. Therefore, it was the major metric in the works of Chapter 4.

The most widely used metric for semantic segmentation is Intersection-over-Union (IoU), also referred to as the Jaccard Index. It quantifies the overlap between the predicted segmentation and the ground-truth mask for a given class:

$$\text{IoU} = \frac{TP}{TP + FP + FN} \tag{2.5}$$

where TP, FP, and FN represent the number of true positive, false positive, and false negative pixels, respectively. IoU measures the ratio between the intersection and the union

of the predicted and actual regions. In this thesis, IoU is adopted as the primary evaluation metric for semantic segmentation, as used in the experiments presented in Chapters 5 and 6.

In general, the mean of a metric is abbreviated with the prefix "m". In the case of the mean IoU, the abbreviation is mIoU.

3 Thesis Synopsis

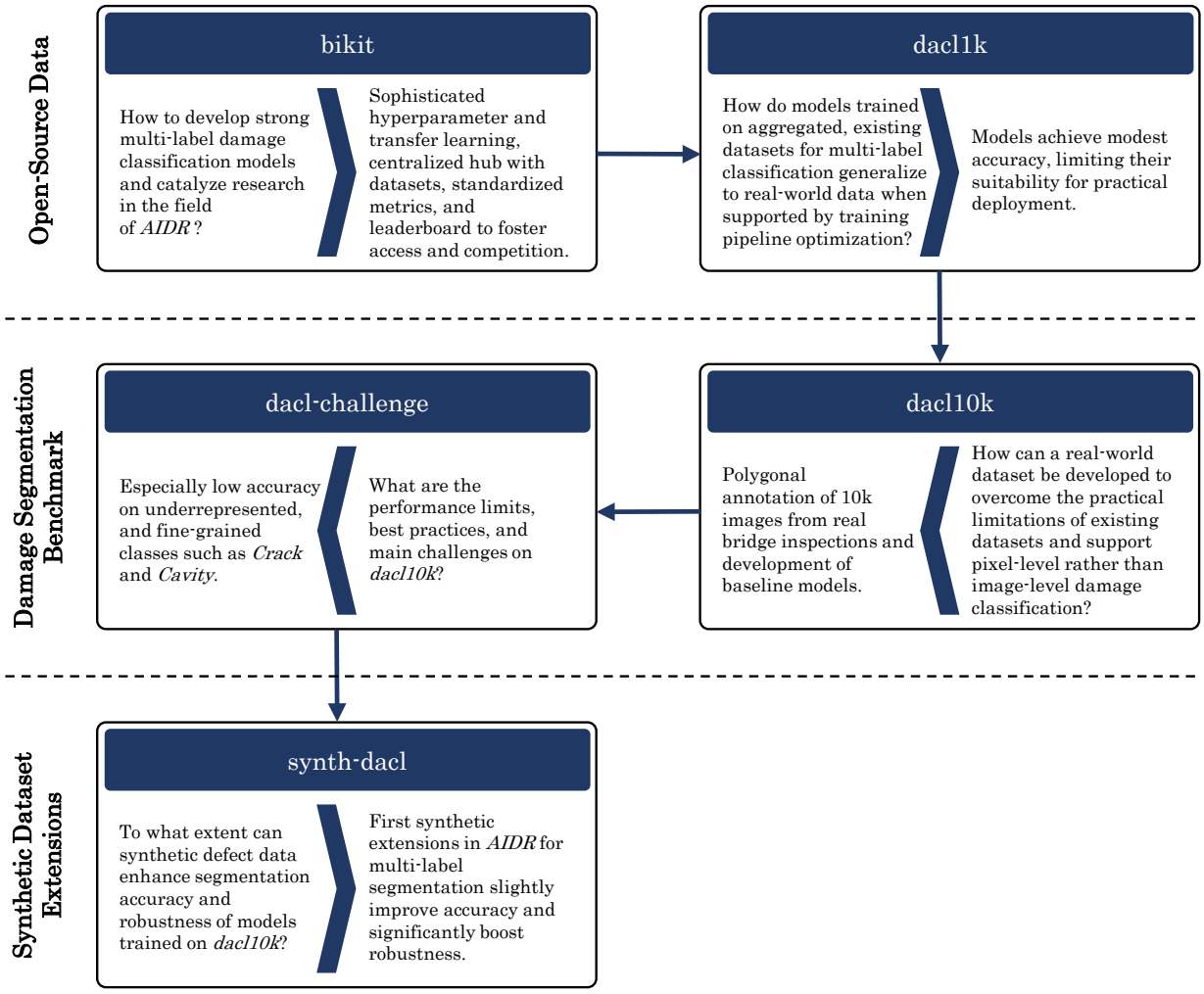


Figure 3.1: Structural outline of the core manuscripts and their interrelations presented in this dissertation. Each box summarizes a key publication (short title in white) with its main research question and contribution. The manuscripts are grouped by dashed lines highlighting affiliation to the main chapter (left, bold text). Thin arrows indicate methodological progression across manuscripts.

In alignment with the structural outline presented in Figure 3.1 the main chapters of this thesis are titled as follows: *Chapter 4: Leveraging Open-Source Data for Image-Based Concrete Defect Recognition*, *Chapter 5: Developing a Damage Segmentation Benchmark for Automated Bridge Inspection*, *Chapter 6: Investigating Synthetic Data Extensions for*

Bridge Damage Segmentation. Each chapter is introduced by a detailed scientific synopsis that outlines the respective manuscripts, their core hypotheses, and key contributions.

3.1 Leveraging Open-Source Data for Image-Based Concrete Defect Recognition

Chapter 4 investigates how existing *AIDR* datasets can be leveraged and evaluated concerning practical usage. The chapter also addresses how accessibility, comparability, and reproducibility of datasets and models within the *AIDR* research community are enhanced.

bikit The initial question was how to develop strong baseline models and foster research in the *AIDR* area. Research in *AIDR* for bridges was hampered by several issues: datasets were difficult to obtain, baseline results were incomparable due to inconsistent data splits and evaluation metrics, no pretrained models were readily available, and there was no public leaderboard to track progress. Furthermore, limited research on transfer learning strategies for the bridge inspection domain existed. The only work investigating transfer learning concepts for *AIDR* models stems from Bukhsh *et al*. [115]. They analyzed model performance on all possible source and target dataset combinations from six concrete defect datasets. Also, models initialized with and without ImageNet pretraining were examined, where ImageNet-pretrained models consistently outperformed others across all datasets. Notably, for CODEBRIM [27], the best results were achieved using only ImageNet weights without additional in-domain pretraining. In contrast, models trained on MCDS [54] benefited from additional pretraining on CODEBRIM. Similarly, binary crack datasets such as SDNET [51] and BCD [53] saw improved performance when models were pretrained on related crack datasets, highlighting the value of task-specific transfer on top of ImageNet features. However, Bukhsh *et al*. use standard hyperparameters, while results are difficult to reproduce and compare. Four of the examined open-source datasets were released initially without predefined splits, thus, they performed their own train/valid/test partitioning but did not publish details of these splits. Moreover, their trained model checkpoints were not made publicly available. Furthermore, as the primary metric, they report the mean of the Area Under the Receiver Operating Characteristic Curve, which wasn't reported on any of the analyzed datasets before. Inspired by the work of Bukhsh *et al*., the "datasets" library [126] providing datasets for Natural Language Processing and *OpenML* [127] offering additionally performance metrics and leaderboards, *bikit* was developed. This toolkit contributes:

- A centralized data hub aggregating publicly available *AIDR* datasets and harmonizing them with standardized training/validation/test splits.
- Task-specific performance metrics, particularly EMR and per-class recall.
- A model hub providing strong pretrained baselines for two realistic multi-label datasets, based on modern CNNs and multiple transfer learning strategies.
- An online leaderboard for transparent benchmarking and community participation.

Additionally, a sophisticated analysis on transfer learning strategies and hyperparameters for three state-of-the-art models was conducted to equip *bikit* with strong baselines. The three different strategies were: (*i*) updating the model's classification head only, (*ii*) sequentially training the head and then the whole model, and (*iii*) training the head and base in parallel but with different learning rates (see Section 2.4).

The toolkit enhances comparability, supports transfer learning research with reproducible baselines, and promotes collaboration in the domain of *AIDR*. Furthermore, with 73.73% EMR, the sophisticated investigation on transfer learning and hyperparameters delivered strong baselines outperforming the previous baseline by 1.5% [27].

The work *Automated Damage Classification on Concrete Bridges Using Convolutional Neural Networks* [128] introduced the concept of bridge inspections incorporating *AIDR* to the German civil engineering community. It presents an analysis of predictions from the best-performing model from *bikit*, trained on the CODEBRIM dataset, and offers an early preview of the "*dacl10k*" dataset, then referred to as "*dacl5k*". The key findings of the prediction analysis focus on Non Exact Matches, according to the definition of the EMR (see Section 2.5). It was observed that many misclassifications stemmed from incorrect labeling, often induced by significant image perturbations. For instance, samples labeled as efflorescence frequently exhibited overexposure, which likely contributed to annotation errors.

dacl1k Following prior work on improving accessibility and comparability of open-source datasets in *bikit* [63], the next central question was: how do these models perform on real-world data? We hypothesize that existing datasets are limited in size and class diversity, and that their images are typically captured under constrained laboratory conditions. These conditions include a narrow range of camera types, fixed acquisition setups regarding camera-object pose, and controlled lighting environments. In contrast, real-world inspection scenarios involve significant variability in environmental conditions, camera hardware, and operator behavior, which are all factors that substantially affect image quality and model performance. The work *dacl1k: Real-world bridge damage dataset putting open-source data to the test* [41] investigates the applicability of compiled open-source datasets to real-world scenarios by training models on these datasets and evaluating them on a novel dataset derived from actual bridge inspections. A refined training pipeline is introduced, improving the setup presented in [63].

In the field of few-shot classification, heterogeneous datasets – such as ImageNet [73] and COCO [129] – have been aggregated into "meta datasets" with subsets held out for validation [130]. The goal was to develop models that generalize from minimal supervision. Recent work extends this by fusing cross-domain and in-domain data for target datasets with weak, sparse, or no labels [131, 132]. In the *AIDR* domain, CrackSeg9k [26] represents the first work on aggregating existing datasets. Its samples were collected from ten different crack datasets across materials like masonry, asphalt, and concrete. Another work in this domain, by Bukhsh *et al.* [115], investigated the transfer between single datasets from the field of concrete damage classification. However, CrackSeg9k primarily concentrates on image preprocessing and model development. Its quantitative evaluation is based on stratified splitting of the entire dataset, which limits its ability to assess cross-dataset generalization. Furthermore, for practical applications in structural inspections, detecting cracks alone is

insufficient; other critical defects, including spalling, efflorescence, or exposed reinforcement, must also be reliably identified. Bukhsh *et al.*, by contrast, did not investigate the transfer of knowledge across multiple defect datasets simultaneously (meta dataset) and relied on default training pipeline settings provided by the chosen Python API, without further optimization.

The work on *dacl1k*[1] builds on the concept of building meta datasets from available sets by harmonizing key datasets relevant to bridge damage detection within a multi-label classification setting. Three meta datasets are created, combining two multi-label and two binary crack datasets. The datasets are incrementally merged, starting with the one closest in domain to *dacl1k*, to study cumulative transfer effects. Building on transfer learning insights from *bikit* [63], the training pipeline is further optimized through the evaluation of five automated augmentation strategies, increased training crop sizes, and multi-label oversampling. Performance is evaluated using both quantitative (extrinsic) and intrinsic methods. Quantitatively, EMR and class-wise recall are reported on *dacl1k*. The model achieving the highest scores on these metrics is selected for the intrinsic evaluation. For each dataset under analysis, 330 images are randomly drawn and passed through the model to extract bottleneck features. These features are then visualized using dimensionality reduction techniques [133] to assess whether resulting clusters correspond more closely to defect classes or the original dataset sources. The refined pipeline yields a 5% improvement in EMR, reaching a maximum of 32%. Yet, intrinsic feature analysis shows that knowledge transfer from existing datasets remains limited for real-world deployment. This shortfall highlights the need for a larger and more diverse real-world dataset.

3.2 Developing a Damage Segmentation Benchmark for Automated Bridge Inspection

Given the modest performance achieved on *dacl1k* through usage of existing *AIDR*-data (see Chapter 4), and since practical *AIDR* applications require the classification, measurement, and localization of defects on pixel level, Chapter 5 introduces the largest and most diverse multi-label semantic segmentation dataset for *AIDR* during concrete bridge inspections *dacl10k*. Additionally, to sensitize the computer vision community to this benchmark, a challenge and a workshop are organized.

dacl10k The first work of this chapter, *dacl10k: Benchmark for Semantic Bridge Damage Segmentation* [18, 134], tackles the question of how to generate a benchmark allowing for pixel-level classification, measurement and localization while targeting all visually distinguishable defects and components that must be recognized according to inspection guidelines [12, 13, 14, 15]. While domains such as autonomous driving [22, 135, 136, 137, 138] and medical imaging [23, 139] have benefited from huge benchmarks for over a decade, the field of automated bridge inspection has lacked such a foundation. There were two relevant datasets in the domain of automated bridge inspections: CrackSeg9k [26] and S2DS [24]. CrackSeg9k comprises 9,255 images from ten open-source datasets showing cracked and uncracked surfaces made of materials such as concrete, brick walls, and asphalt. Most of the subsets of

[1]The word *dacl* is the acronym for damage classification.

CrackSeg9k are acquired under constant laboratory conditions, two of them were part of the analysis in the work on *dacl1k* [41], questioning their suitability during practical usage. S2DS is the first semantic segmentation dataset within the realm of automated bridge inspections, including 743 patches extracted from 8,435 images taken during structural inspections. It contains the five most common concrete defect types (crack, spalling, corrosion, efflorescence, vegetation) and control point used for georeferencing. However, S2DS employs a single-label semantic segmentation task, meaning overlapping defects cannot be represented. Additionally, the relatively small number of samples raises concerns regarding generalizable model training. *dacl10k* consists of 9,920 images that stem from real-world bridge inspections. The second version of this dataset "*dacl10k-v2* [2]" provides 13 defects and 6 bridge components. This positions *dacl10k* as the largest and most diverse dataset for multi-label semantic segmentation focused on bridge inspection. In the following, this version and its associated statistics are referred to as *dacl10k*. The accompanying statistics reveal a significant class-imbalance on pixel level. For instance, only 0.14% of the pixels are annotated as *Rockpocket*, 0.30% as *Crack*, and 0.39% as *Cavity*. For comparison, 9% of pixels are marked as *Weathering*. Two semantic segmentation architectures, DeepLabV3+ and FPN, each combined with three image encoders as well as a transformer-based model, SegFormer, provide initial baseline results on the benchmark (see Section 2.3). The best result can be obtained for FPN with an EfficientNet-B4 [140] backbone, which achieves a mIoU of 41%. A qualitative evaluation is enabled by an "Engineer vs. Machine" comparison confronting annotations by five civil engineers and predictions of the best segmentation model on four test samples. Three images show better segmentation by the model than the annotations by the engineers. One sample showing seven different – mostly overlapping – defect classes is less accurately predicted by the model. Still, the predictions are, when it comes to practical use, sufficiently accurate for most classes and could support the inspector decisively. The only notable limitation is the *Crack* class: due to coarse annotations in the *dacl10k* dataset, model predictions do not align precisely with crack edges, impeding accurate crack width estimation.

dacl-challenge Civil engineering has recently increased its visibility through workshops at computer vision conferences. For example, at ECCV'22 the *Computer Vision for Civil and Infrastructure Engineering Workshop* [141] brought together a diverse set of papers and posters. *AIDR*-related contributions were CrackSeg9k for concrete crack segmentation, active learning on imbalanced inspection data [142], or model-assisted labeling using explainability techniques for visual inspection [143]. Separately, the "1st workshop on Vision-based InduStrial InspectiON (VISION)" [144], at CVPR'23, focused on research and emerging practical challenges in industrial inspection using computer vision. It hosted a two-track challenge that aimed to develop methods that leverage limited polygonal-labeled samples alongside abundant unlabeled images drawn from diverse industrial settings (e.g., metal cylinders, rings), and produce algorithmic solutions capable of generating realistic synthetic defects to boost model performance under label-scarce conditions. In general, all impactful benchmarks in image recognition are accompanied by a challenge to raise awareness [75, 99, 98, 145]. In contrast, the domain of automated visual inspection for civil infrastruc-

[2] The updated statistics for *dacl10k-v2* were released in the paper *dacl-challenge: Semantic Segmentation during Visual Bridge Inspections* [89].

ture has, until recently, lacked both a focused workshop and a corresponding benchmark challenge. To bridge this gap, the "*dacl-challenge: Semantic Segmentation during Visual Bridge Inspections*" [89] [3] and the corresponding "*1st Workshop on Vision-Based Structural Inspections in Civil Engineering*" were organized. Most of the challenge submissions incorporated ensemble learning and found pretrained weights from ADE20k or ImageNet useful. In general, the training pipeline optimization, including augmentation schemes, optimizer, and hyperparameters, was the major determinant of performance, rather than architectural considerations. The best mIoU was 51%, 14 percentage points higher than the original baseline. Regarding model architectures, the best performance over all non-ensemble approaches was achieved by pairing the FPN segmentation architecture coupled with a MaxViT [87] image encoder (see Section 2.3). The leaderboard unfolds important challenges, which are motivation for the subsequent chapter. The lowest class-specific IoU for *Washouts/Concrete corrosion*[4] (20%), followed by *Cavity* (26%), *Rockpocket* (37%), and *Crack* (41%). The concrete defect class *Cavity* refers to small air voids on the concrete surface, whereas *Rockpocket*, also known as honeycombing, denotes areas where coarse aggregate is exposed due to inadequate deaeration, thus, the absence of cement paste. A comprehensive annotation guideline for these defects is provided in the supplementary material of the *dacl10k* dataset publication.

3.3 Investigating Synthetic Data Extensions for Bridge Damage Segmentation

In general, models deployed for *AIDR* in real-world bridge inspections must exhibit strong robustness to a broad spectrum of challenging conditions. These include variations in image quality, background textures, as defects often appear on concrete surfaces with differing textures and levels of weathering. Such robustness is even harder to achieve for underrepresented classes revealed through statistics on *dacl10k* and results of the *dacl-challenge*. In the competition, contestants implementing sophisticated training pipeline optimization, model searches, and ensembles of models could not eradicate class imbalance of concrete defects such as *Rockpocket* and the challenging nature of the defect classes *Crack* and *Cavity*. Building on these data-specific limitations, the study *synth-dacl: Does Synthetic Defect Data Enhance Segmentation Accuracy and Robustness for Real-World Bridge Inspections?* [90, 146] explores the potential of synthetic data to mitigate class imbalance and enhance dataset diversity, which are key factors hindering segmentation performance. This is particularly relevant for fine-grained defect classes such as *Crack* and *Cavity*. The overarching goal is to enhance model accuracy and robustness by incorporating synthesized concrete defect data.

To this end, the work introduces *synth-dacl*, a set of three novel synthetic dataset extensions, each consisting of 5,000 samples rendered with realistically simulated concrete textures and synthesized *Weathering*. The rendered concrete surfaces are an extended version of the physics-based rendering (PBR) introduced in Jaziri *et al.* [61] (see Section 2.2), which are described as follows:

[3] `https://eval.ai/web/challenges/challenge-page/2130/overview`

[4] Further investigation of the *Washouts/Concrete Corrosion* class would require an extensive relabeling process. Therefore, this class is excluded from subsequent analysis.

- The first extension, *daclonsynth*, is a semi-synthetic dataset where defect instances from the *dacl10k* training set are cropped and overlaid onto synthesized concrete background. This is inspired by the work of Dwibedi *et al.* [124] who paste object instances on real-world background scenes. The class distribution of generated samples is dynamically computed to balance the original *dacl10k* set, considering both shape and pixel counts. This strategy introduces both balance and diversity.

- The second extension, *synthcrack*, is purely synthetic. It features fully synthetic cracks mapped on the synthesized concrete backgrounds. This approach is based on the work of Jaziri *et al.* [61].

- The third extension, *synthcavity*, follows the same concept but targets a novel synthetic generation of *Cavities*.

Furthermore, a revised and cleaned version of *dacl10k* is provided, addressing previously confused classes (*e.g.*, *Spalling* vs. *Rockpocket*). The test set now additionally includes pixel-accurate masks for *Cracks*, which are necessary to properly evaluate models trained on the *synthcrack* extension, whose masks exhibit crack-edge-exact precision. This enables a fair assessment on real-world data of potentially more accurate predictions compared to original coarse annotations.

The results show that adding purely synthetic extensions individually or in combination with the semi-synthetic extension, *daclonsynth*, the best class-level performance is obtained. Moreover, it can be stated that models trained solely on *synthcrack* outperform those trained on real crack annotations from *dacl10k* when evaluated on finecrack masks. This suggests that the feature distributions learned from well-designed synthetic samples can surpass those derived from manually annotated real data. An additional evaluation across 15 perturbed test sets, following the work of [147], shows that incorporating the *synth-dacl* extensions during training significantly improves model robustness. Notably, models trained on the combined *dacl10k* and synthetic datasets achieve consistent 2% gains in mean IoU, F1 score, Recall, and Precision, compared to models trained on *dacl10k* alone. Nonetheless, there is still room for improvement. In particular, *synthcavity* lacks representations of large *Cavities* with sharp edges, which could be addressed through more advanced synthetic data generation techniques. Overall accuracy must be raised as a mIoU of 43% is not enough compared to other semantic segmentation benchmarks.

4 Leveraging Open-Source Data for Image-Based Concrete Defect Recognition

Publications in this chapter:

- J. Flotzinger, P. J. Rösch, N. Oswald, and T. Braml. Building inspection toolkit: Unified evaluation and strong baselines for bridge damage recognition. In 2022 IEEE International Conference on Image Processing (ICIP), pages 1221–1225, 2022. doi: 10.1109/ICIP46576.2022.9897743
- J. Flotzinger and T. Braml. Automatisierte Klassifizierung von Schäden an Massivbrücken mittels Neuronaler Netze [Automated Damage Classification on Concrete Bridges Using Convolutional Neural Networks]. Beton Stahlbetonbau, 117(10):786–794, October 2022. doi: 10.1002/best.202200068
- J. Flotzinger, P. J. Rösch, N. Oswald, and T. Braml. dacl1k: Real-world bridge damage dataset putting open-source data to the test. Engineering Applications of Artificial Intelligence, 137:109106, 2024. doi: 10.1016/j.engappai.2024.109106

BUILDING INSPECTION TOOLKIT: UNIFIED EVALUATION AND STRONG BASELINES FOR BRIDGE DAMAGE RECOGNITION

Johannes Flotzinger, Philipp J. Rösch, Norbert Oswald, Thomas Braml

University of the Bundeswehr Munich

{johannes.flotzinger,philipp.roesch,norbert.oswald,thomas.braml}@unibw.de

ABSTRACT

In recent years, several companies and researchers have started to tackle the problem of damage recognition within the scope of automated inspection of built structures. While companies are neither willing to publish associated data nor models, researchers are facing the problem of data shortage on one hand and inconsistent dataset splitting with the absence of consistent metrics on the other hand. This leads to incomparable results. Therefore, we introduce the building inspection toolkit – bikit – which acts as a simple to use data hub containing relevant open-source datasets in the field of damage recognition. The datasets are enriched with evaluation splits and predefined metrics, suiting the specific task and their data distribution. For the sake of compatibility and to motivate researchers in this domain, we also provide a leaderboard and the possibility to share model weights with the community. As a starting point we provide strong baselines utilizing extensive hyperparameter search using three transfer learning approaches for state-of-the-art algorithms. The toolkit[1] and the leaderboard[2] are available online.

Index Terms— deep learning, transfer learning, building damage recognition

1. INTRODUCTION

Bridges represent a very important part of the infrastructure worldwide. Therefore, the availability of functioning and safe bridges is crucial. Due to the fact that many bridges are currently reaching ages at which defects occur more often [1] the assessment intervals must be set more frequently, which increases the pressure on individual inspectors.

We want to accelerate the development in the context of damage recognition and therefore introduce *bikit*. It is designed to be an easy to use platform and consists of three components: (*i*) datasets with fixed splits, (*ii*) a public leaderboard with predefined metrics, and (*iii*) the availability to upload trained models. Although being a niche area, there are some datasets for reinforced concrete damage (RCD) publicly available. Most of them are binary classification datasets that examine the presence of cracks or defects in general. Besides that, there are also more detailed and realistic multi-target datasets available (see Table 1). In the current *bikit* version we provide all open-source datasets. Apart from accessibility, comparability is another crucial feature. Hence, fixed test sets and useful metrics are required. We supply predefined training, validation, and testing splits if missing. On the basis of data distribution, we create individual splits. Moreover, useful metrics will be defined for each dataset to evaluate performance. The multi-target datasets are provided with a strong baseline model using state-of-the-art algorithms. For this purpose, we run an expensive hyperparameter search using three different transfer learning strategies on famous (pre-trained) models and make these available online.

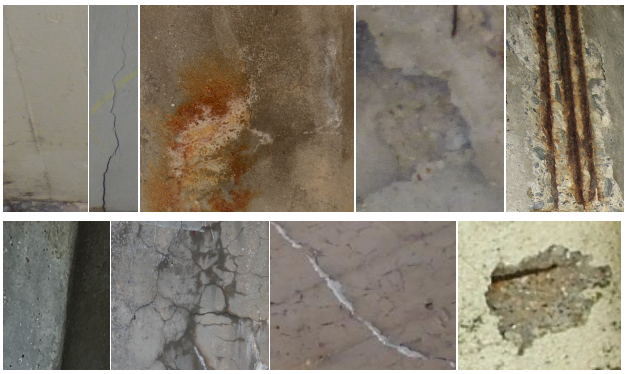


Fig. 1: Example defects from CODEBRIM (top line) and MCDS (bottom line).

2. SIMILAR APPROACHES

Data and Model Hubs. Machine learning research is based on the availability of data. In recent years, the number of datasets that are open to the public has continued to grow and the usage has been simplified. For example, *datasets* [2] is a platform for a large number of NLP datasets from a wide variety of fields. Besides renowned datasets – often cited in research – niche datasets and low-resource languages can also be accessed. *OpenML* [3] provides metrics and leaderboards

[1] https://github.com/phiyodr/building-inspection-toolkit
[2] https://dacl.ai

Dataset	Classes	Types	Size	Splits
CDS [5]	2	STC	1,028	no
BCD [6]	2	STC	5,390	✓
SDNET2018 [7]	2	STC	13,620	no
ICCD[3] [8]	2	STC	60,010	✓
MCDS [9]	8	MTC	3,607	no
CODEBRIM [10]	6	MTC	7,261	✓

Table 1: Open-source datasets for damage recognition on reinforced concrete bridges. Single-target (STC) and multi-target classification (MTC). Splits refer to the presence of fix train, valid and test splits in the original publication.

Classes	MCDS	CODEBRIM
No Damage	452	2,506
Crack	787	2,507
Efflorescence	304	833
Spalling	422	1,898
Exposed Bars	221	1,507
Rust	350	1,559
Scaling	163	-
Other Defects	264	-
Number of images	2,597	7,261
Avg. number of classes/image	1.14	1.13

Table 2: Counts of classes in datasets with additional statistics.

in addition to data and helps not only with readily available data but also with results on specific hyperparameter settings. The above mentioned developments accelerate research and make results more comparable.

Transfer Learning. The most comprehensive work in the area of RCD recognition was executed by Bukhsh *et al.* [4]. They investigated several concepts of transfer learning for six classification datasets. Transfer learning strategies consist of varying combinations of upstream and downstream datasets using cross-domain and in-domain data. In their analysis, common CNNs up to the year 2015 were taken into account. We decided to use more recent architectures for our analysis. Moreover, they only trained the networks' head. Besides that, we also investigate the gain of updating the whole model.

3. BUILDING INSPECTION TOOLKIT

The toolkit consists of three parts. The first building block are the datasets conceived by other researchers and now harmonized into one module. We also add training, validation and testing data splits if they do not exist. The second component is the introduction of performance metrics, which fit task type and data distribution. Results are made available via an online leaderboard. As a third part we launch a model hub to make models accessible.

Datasets. Several RCD datasets are publicly available for single-target and multi-target classification (see Table 1). To our knowledge no open-source object detection and semantic segmentation datasets exists currently. In this publication we deal with two multi-target classification datasets MCDS [9] and CODEBRIM [10]. They are more challenging than single-target tasks and are also more realistic, since several damage can be present on one image at the same time. Moreover, images are taken in a less standardized setting with different camera angles and image resolutions.

In total there are seven specific damage types that are currently available in RCD datasets, i.e. *cracks, efflorescence, spalling/spallation, exposed bars/reinforcement, rust staining/corrosion strain, scaling*, and *other/general defects*. Other types exist, but are aggregated in meta classes. MCDS subordinates *graffiti* and *vegetation* to "Other Defects". So far MCDS is the most diverse dataset available, whereas the *German guideline for structural inspections* [11] currently distinguishes approximately 50 RCD in total. There is still a lot of work to be done, until machine learning-based systems can reliably support civil engineers.

MCDS. This dataset [9] was created for a three-stage approach where, depending on the results of the first model, a second model would be applied and, depending on that, a third. The second and third stage determine whether *exposed bars* and *rust staining* is visible. We cleaned the dataset and transferred it to an eight-class dataset. Due to the fact that no data splits were provided we introduced fix splits for training (2057 samples), validation (270), and testing (270).

CODEBRIM. This dataset is the second richest in terms of damage categories. Compared to MCDS, two classes are missing, but the number of unique images is much higher. The authors also made splits available so that all former research is comparable and no splits need to be created. The numbers for each split are 6013 (training), 616 (validation), and 632 (testing). We use the balanced version for our experiments.

Metrics. On the one hand, authors of MCDS decided to provide a huge number of metrics. On the other hand CODEBRIM authors chose to report Exact Match Ratio (EMR) only. We go a middle course and report *EMR* and *recall by class*. We make both metrics accessible in an online leaderboard.

EMR. EMR is a challenging task since all predictions must match. While EMR is not a metric created directly for unbalanced data, it is complex enough so that the distribution problem is adequately addressed. Moreover, a side effect is that current values are still not close to perfect fit, which leaves room for improvement within this research domain.

Recall by class. The previous metric was primarily selected from a machine learning perspective. For civil engi-

[3]Data is only available upon request.

Data	Approach	Model	EMR	Recall by class							
				NoDam.	Crack	Spall.	Effl.	Expos.	Rust	Scal.	Other
CODEBRIM	HO	RN	63.77	89.33	77.33	76.67	62.42	78.67	80.00	-	-
	HO	EN	59.97	84.00	72.67	85.33	53.69	92.00	80.67	-	-
	HO	MN	63.92	88.00	74.00	80.67	68.46	82.00	70.67	-	-
	HTA	**RN**	**73.73**	94.67	**88.00**	84.00	**75.84**	88.67	79.33	-	-
	HTA	EN	65.35	90.67	76.67	87.33	65.10	**92.67**	81.33	-	-
	HTA	MN	69.46	94.00	78.67	83.33	69.80	88.67	76.67	-	-
	DHB	RN	70.57	**95.33**	87.33	86.67	75.17	90.00	82.00	-	-
	DHB	EN	68.67	92.00	83.33	**88.67**	74.50	**92.67**	**85.33**	-	-
	DHB	MN	69.15	90.00	79.33	81.33	69.13	86.00	78.67	-	-
MCDS	HO	RN	44.44	66.67	63.33	73.33	63.33	36.67	75.00	23.33	23.33
	HO	EN	30.37	33.33	53.33	43.33	56.67	10.00	56.67	0.00	3.33
	HO	MN	47.04	56.67	70.00	56.67	70.00	25.00	**76.67**	50.00	36.67
	HTA	RN	43.70	73.33	53.33	63.33	60.00	25.00	65.00	**76.67**	10.00
	HTA	EN	46.30	46.67	76.67	55.56	80.00	26.67	68.33	30.00	26.67
	HTA	**MN**	**54.44**	70.00	76.67	58.89	**90.00**	21.67	68.33	43.33	**46.67**
	DHB	RN	48.15	66.67	73.33	44.44	86.67	23.33	65.00	36.67	43.33
	DHB	EN	51.85	46.67	73.33	**61.11**	80.00	**38.33**	75.00	43.33	**46.67**
	DHB	MN	51.85	**80.00**	**83.33**	57.78	73.33	30.00	66.67	33.33	36.67

Table 3: EMR and recall by class for all models and transfer learning approcheas. Bold font indicates best results and underlined best results per transfer learning approach.

neers it is mainly relevant to see how many damage are overlooked. In current approaches, machine learning-based software is made to support and not to replace engineers. Consequently, it is more vital to detect all defects that exist than missing risky damage. Due to that fact, we decided to provide *recall* measures for all classes, in order to be able to see where current models still fail often.

Model Hub. We also create a model hub to accelerate research. Having pre-trained models available does not only help to reproduce results but also to obviate the need to fine-tune models again and again. This is especially interesting if researchers want to adapt pre-trained classification models for later use in object detectors or for semantic segmentation tasks.

4. STRONG BASELINES FOR DAMAGE RECOGNITION

To fill the model hubs with strong baselines we use three modern CNN architectures and conduct an expensive hyperparameter search with three different transfer learning strategies.

Architectures. The used CNNs are ResNet50 (RN) [12], MobileNetV3-Large (MN) [13], and EfficientNetV1-B0 (EN) [14], which all belong to a set of state-of-the-art models. We initialize the models with weights from ImageNet [15] training. The original heads are removed and replaced by two fully connected layers that have dropout applied before each layer.

Transfer Learning. In total we want to train and compare three different transfer learning strategies. We use one standard approach and two more advanced strategies. (*i*) The first updates the head's weights only (*HO*). The learned fully-connected layers act as a classifier on the bottleneck features [16] derived from training on ImageNet objects. (*ii*) The second initially trains the head and then all weights (*HTA*). Kornblith *et al.* [17] show that this can leverage performance significantly. (*iii*) The last applies different learning rates for model head and base (*DHB*). This strategy was introduced by Howard and Ruder [18] as discriminative fine-tuning.

Hyperparameter Search. We defined a rich set of parameters, which are evaluated for the *HO* approach. All variable settings are listed in Table 4. The models are trained using Adam with weight decay and data augmentation (center-cropping, rotating, flipping). The number of epochs is always set to 100 and the warmup duration is fixed to 10 epochs. To conduct the hyperparameter tuning we use *Bayesian Search* [19, 20] and *Hyperband-Early-Stopping* [21, 20]. Results of the best settings are provided in Table 5.

We use these results to reduce the parameter space for the *HTA* and *DHB* approach and only try different learning rates as indicated in Table 6. For *HTA* we use the best checkpoint from *HO* for the first stage and then train the model for another 100 epochs but without frozen weights. *DHB* is trained from scratch for 100 epochs, using the best hyperparameter settings from *HO* as well.

Hyperparam.	Parameter space
Hidden layer	16, 32, 64, 128, 256, 512, 1024
Batch size	16, 32 (MCDS); 64, 128, 256, 512 (both)
Learning rate	1e-4, 5e-4, 1e-3, 5e-3, 1e-2
Scheduler	Constant (CtW), Cosine (CeW) w/ warmup
Dropout	0, 0.1, 0.2, 0.3, 0.4
Weight decay	1e-7, 1e-6, 1e-5

Table 4: Values for the initial parameter search (*HO*) for CODEBRIM and MCDS. The best hyperparameter settings are displayed in Table 5.

Data	Hyperparameters	RN	EN	MN
CODEBRIM	Hidden layer	128	256	1024
	Batch size	256	256	64
	Learning rate	1e-5	1e-7	1e-6
	Scheduler	CtW	CeW	CeW
	Dropout	0.2	0.4	0.3
	Weight decay	1e-5	1e-7	1e-6
MCDS	Hidden layer	32	32	64
	Batch size	64	32	64
	Learning rate	5e-3	5e-3	5e-3
	Scheduler	CeW	CtW	CtW
	Dropout	0.0	0.4	0.2
	Weight decay	1e-5	1e-7	1e-6

Table 5: Best hyperparameter results for *HO* from Bayesian Search for ResNet (RN), EfficientNet (EN), and MobileNet (MN). Possible settings and abbreviations are listed in Table 4.

Hyperparameters	Learning rate space
HTA/DHB head	1e-4, 5e-4, 1e-3, 5e-3, 1e-2
DHB base	1e-7, 1e-6, 1e-5

Table 6: Values for *HTA* and *DHB* approach which are examined for all architectures and datasets. For the remaining hyperparameter see Table 5.

Data	Learning rate	RN	EN	MN
CODEBRIM	HTA	1e-5	1e-5	1e-5
	DHB head	1e-4	1e-3	1e-3
	DHB base	1e-5	1e-5	1e-5
MCDS	HTA	1e-7	1e-5	1e-5
	DHB head	1e-4	1e-4	5e-3
	DHB base	1e-5	1e-5	1e-5

Table 7: Best learning rate results for *HTA* and *DHB* for ResNet (RN), EfficientNet (EN), and MobileNet (MN). Possible learning rate settings are listed in Table 6.

5. RESULTS OF HYPERPARAMETER-TUNING AND TRANSFER LEARNING ANALYSIS

We want to share strong baseline models for CODEBRIM as well as MCDS and therefore, went through a computational expensive evaluation process. Our results are reported in Table 3. The more flexible approaches outperform *HO* clearly. While the performance of *HTA* and *DHB* depends on the model and dataset.

CODEBRIM. The overall best result shows ResNet with *HTA*, which scores 73.73 percent on EMR. This model outperformes the initially published model slightly by 1.5 percent points [10]. We cannot compare to Bukhsh *et al.*, since they do not provide EMR. The damage recall is the lowest for efflorescence (75.84 percent) and the highest for recognizing exposed bars (88.67 percent). The current best CODEBRIM model still missed 1/4 of efflorescence defects, which is probably related to the small amount of images. On the other hand, results for exposed bars are already in a good range (over 86.00 percent for all flexible approaches).

MCDS. The best model is MobileNet with *HTA* transfer learning. With a top EMR of 54.44 percent, performance on this dataset is substantially worse than for CODEBRIM, which is due to the dataset size. Recall is best for efflorescence, where 9 out of 10 defects are detected. The lowest recall is obtained for exposed bars, which have the second smallest number of images in MCDS. Since the original [9] and modified setting [4] are structured differently and no fix splits are provided, we cannot compare the results.

Transfer Learning. Our results show that not only training the head, but also the base part of the architecture significantly improves transfer learning outcomes. We have included these strategies because RCD images differ a lot from ImageNet objects which were used for the pre-training of our models. This leads to the conclusion that, after the classical *HO* approach has been conducted, it is reasonable to train all weights. *DHB* has the advantage of requiring only one pass.

6. CONCLUSIONS

The building inspection toolkit simplifies access to datasets and metrics and introduces splits in case they are missing. We also provide strong baseline models. An online leaderboard finally facilitates comparability between models and accelerates the progress within this application oriented research area. Experiments on the CODEBRIM dataset showed that ResNet works best, whereas MobileNet with *HTA* is the best model for MCDS. During experiments we found out that it is of great importance also to adapt the filter weights.

All parts of the building inspection toolkit will help to speed up research in damage recognition n built structures and give it a higher priority and greater visibility.

7. REFERENCES

[1] American Road and Transportation Builders Association (ARTBA), "2021 bridge report," 2021-11-09.

[2] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander M. Rush, and Thomas Wolf, "Datasets: A community library for natural language processing," 2021.

[3] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo, "OpenML: Networked Science in Machine Learning," *SIGKDD Explorations*, vol. 15, no. 2, pp. 49–60, 2013.

[4] Zaharah A. Bukhsh, Nils Jansen, and Aaqib Saeed, "Damage detection using in-domain and cross-domain transfer learning," *Neural Computing and Applications*, vol. 33, 12 2021.

[5] Philipp Hüthwohl and Ioannis Brilakis, "Detecting healthy concrete surfaces," *Advanced Engineering Informatics*, vol. 37, no. April, pp. 150–162, 2018.

[6] Hongyan Xu, Xiu Su, Huaiyuan Xu, and Haotian Li, "Autonomous bridge crack detection using deep convolutional neural networks," in *3rd Int. Conf. on Computer Engineering, Information Science & Application Technology*. Atlantis Press, 2019, pp. 274–284.

[7] Sattar Dorafshan, Robert J. Thomas, and Marc Maguire, "SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks," *Data in Brief*, vol. 21, no. November, pp. 1664–1668, 2018.

[8] Shengyuan Li and Xuefeng Zhao, "Image-Based Concrete Crack Detection Using Convolutional Neural Network and Exhaustive Search Technique," *Advances in Civil Engineering*, 2019.

[9] Philipp Hüthwohl, Ruodan Lu, and Ioannis Brilakis, "Multi-classifier for reinforced concrete bridge defects," *Automation in Construction*, vol. 105, Sep 2019.

[10] Martin Mundt, Sagnik Majumder, Sreenivas Murali, Panagiotis Panetsos, and Visvanathan Ramesh, "Meta-learning convolutional neural architectures for multi-target concrete defect classification with the codebrim dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[11] Bundesanstalt für Straßenwesen, "Richtlinie zur einheitlichen Erfassung, Bewertung, Aufzeichnung und Auswertung von Ergebnissen der Bauwerksprüfungen," 2017-02-22.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," 2015.

[13] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam, "Searching for mobilenetv3," *CoRR*, vol. abs/1905.02244, 2019.

[14] Mingxing Tan and Quoc V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 10691–10700, 2019.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[16] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *Proceedings of the British Machine Vision Conference*, pp. 1–11, 2014.

[17] Simon Kornblith, Jonathon Shlens, and Quoc V. Le, "Do better imagenet models transfer better?," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, pp. 2656–2666, 2019.

[18] Jeremy Howard and Sebastian Ruder, "Universal language model fine-tuning for text classification," *56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, vol. 1, pp. 328–339, 2018.

[19] Christopher K Williams and Carl Edward Rasmussen, *Gaussian processes for machine learning*, MIT press Cambridge, MA, 2006.

[20] Lukas Biewald, "Experiment tracking with weights and biases," 2020, Software available from wandb.com.

[21] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *Journal of Machine Learning Research*, vol. 18, pp. 1–52, 2018.

Johannes Flotzinger, Thomas Braml

AUFSATZ

Automatisierte Klassifizierung von Schäden an Massivbrücken mittels Neuronaler Netze

Herrn Prof. Dr.-Ing. Manfred Keuser zur Vollendung seines 70. Lebensjahrs gewidmet

Vor dem Hintergrund eines alternden Bauwerksbestands sowie des stetigen Anstiegs des Schwerverkehrs ist eine regelmäßige und qualitativ hochwertige Bauwerksprüfung unabdingbar. Bei der Bewältigung dieser Aufgabe birgt die Zuhilfenahme digitaler Methoden im Rahmen der digitalisierten Inspektion (DI) großes Verbesserungspotenzial in Hinblick auf Wirtschaftlichkeit und Qualität. Ein wesentlicher Bestandteil der DI ist das automatisierte Erkennen von Schäden mit Künstlichen Neuronalen Netzen. Im Rahmen des Forschungsprojekts „Modellbasierte digitale Bauwerksprüfung – MoBaP“ werden an der Universität der Bundeswehr München Neuronale Netze für die Klassifizierung von Schäden an Massivbrücken trainiert. Auf dem derzeit größten Open-Source-Datensatz (CODEBRIM) dieser Domäne erzielt das im Folgenden dargestellte Netz eine Exact Match Ratio von 74 % und definiert damit das aktuell beste Modell zur Multi-Target-Klassifizierung. Um auch Neuronale Netze für die Objektdetektion und semantische Segmentierung dieser Domäne zu trainieren, wird ein eigener Datensatz erstellt. Dadurch wird neben dem Klassifizieren auch das Lokalisieren der Schäden auf Bildern ermöglicht. In diesem Aufsatz erörtern die Autoren das Vorgehen zum Trainieren Neuronaler Netze für die Klassifizierung von Schäden an Massivbrücken und eine detaillierte Analyse von Testergebnissen. Außerdem werden die Entwicklung und der aktuelle Stand eines eigenen Datensatzes vorgestellt.

Stichworte automatisierte Schadensklassifikation; digitale Bauwerksprüfung; Deep Learning; Künstliche Neuronale Netze

Automated Damage Classification on Concrete Bridges Using Convolutional Neural Networks

Against the background of an ageing structure stock and the constant increase in heavy traffic, frequent structural inspections of high quality are indispensable. In accomplishing this task, the use of digital methods within the framework of digitized inspections (DIs) offers great potential for improvement in terms of cost-effectiveness and quality. An essential component of DIs is the automated detection of damage with Convolutional Neural Networks (CNNs). As part of the research project “Model-Based Digital Structural Inspection – MoBaP”, CNNs are being trained at the University of the Bundeswehr Munich for the classification of defects occurring on concrete bridges. On this domain’s currently largest open-source dataset (CODEBRIM), the best CNN achieves an exact match ratio of 74 % and thus currently defines a strong baseline. In order to also train neural networks for object detection and semantic segmentation in this domain, a separate dataset is created. This enables not only the classification but also the localisation of damage on images. In this paper, the authors discuss the procedure of training neural networks for the classification of defects on concrete bridges and show a detailed analysis of test results. In addition, the development and current status of their own dataset is presented.

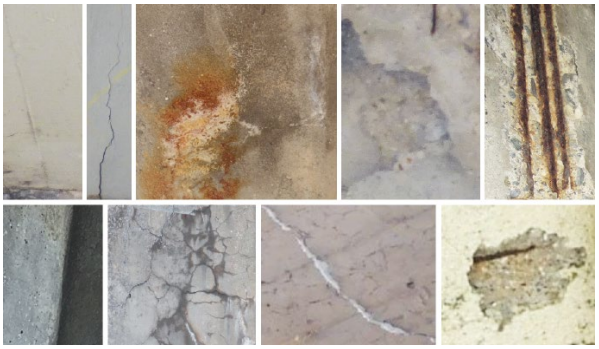
Keywords damage classification; digital inspection; deep learning; convolutional neural networks

1 Einleitung

Die Hälfte der ca. 40.000 Brücken im deutschen Bundesfernstraßennetz wurde zwischen den Jahren 1965 und 1985 errichtet. Hinzu kommen mehr als 26.000 Brücken, welche in der Baulast der Bundesländer liegen, sowie Brücken der Kommunen [1, 2]. Aufgrund der Altersstruktur der Bauwerke kommt es gehäuft zu Schäden (Bild 1), bspw. infolge fortschreitender Carbonatisierung, Frost-Tau-Wechsel oder des Eintrags von Chloriden. Zudem wird laut BMVI [1] die Güterverkehrsleistung respektive Schwerverkehr bis 2025 weiter steigen, was zu einer erhöhten Beanspruchung der gebauten Struktur führt. Durch regelmäßige Prüfungen können Instandhaltungsmaßnahmen an den Bauwerken ideal geplant und die Belastung der Wirtschaft und Bevölkerung infolge Verkehrsbehinderungen, speziell wenn diese abrupt erforderlich sind, gering gehalten werden [3].

In Deutschland erfolgt die Bauwerksprüfung nach DIN 1076 [6] in Kombination mit der Richtlinie RI-EBW-PRÜF [7] und dem Prüfungsmanagement durch SIB-Bauwerke [8], im Weiteren analoge Bauwerksprüfung genannt. Die Digitalisierung von Teilen des bisherigen analogen Prozesses bietet großes Potenzial, die Bauwerksprüfer:innen in deren Tätigkeit zu unterstützen. Die Prozessbestandteile der analogen Bauwerksprüfung, wie das Klassifizieren, Vermessen, Verorten und Bewerten der Schäden, ist oft inkonsistent, fehleranfällig und langwierig. Laut Phares et al. [9] variieren die Prüfergebnisse zwischen Bauwerksprüfenden stark. Es wurden Prüfberichte von 49 Brückeninspekteur:innen aus 25 verschiedenen staatlichen Verkehrsministerien (DOT) an je sieben Bauwerken in den USA ausgewertet. Es stellte sich heraus, dass ca. 56 % der Zustandsnoten deutlich von der Referenz-Zustandsnote abwichen.

Die bisher gesammelten Erkenntnisse zeigen, dass durch die digitale Bauwerksprüfung und den damit einherge-



Quelle: Mundt et al. [4] und Hüthwohl et al. [5]

Bild 1 Schadensbilder v. l. n. r. der beiden Open-Source-Datensätze CODEBRIM (oben): kein Schaden, Riss, Ausblühung/Korrosion, Abplatzung, Abplatzung/offenliegende Bewehrung/Korrosion und MCDS (unten): kein Schaden, Riss, Ausblühung, Abplatzung/offenliegende Bewehrung/Korrosion [15]
Example damages from CODEBRIM (top line) and MCDS (bottom line); the labels from left to right for CODEBRIM are: No Damage, Crack, Efflorescence/Rust, Spalling, Spalling/Exposed Bars/Rust; the labels for MCDS are: No Damage, Crack, Efflorescence, Spalling/Exposed Bars/Rust [15]

henden Einsatz von aktuellen digitalen Methoden Inspektionen genauer und effizienter durchgeführt werden können. Ein wesentlicher Bestandteil der digitalen Inspektion ist die automatisierte Erkennung von Fehlstellen, auch Schadenserkennung genannt. Die Schadenserkennung umfasst das Klassifizieren, Vermessen, Lokalisieren und Evaluieren von Schäden. Für den Teilprozess des Klassifizierens eignen sich Bilder als Datengrundlage besonders, da – bis auf Hohlstellen – alle Schäden an Massivbrücken visuell erkennbar sind und die nötige Hardware zum Erfassen und Auswerten der Bilder, wie z. B. ein Smartphone, bei Inspektionen in der Regel vorhanden ist.

Künstliche Neuronale Netze, im Englischen „Convolutional Neural Networks“ (CNNs) aus dem Bereich der Künstlichen Intelligenz (KI), lösen seit den 1990er-Jahren erfolgreich Probleme der Bilderkennung und verbessern damit Prozesse in vielen Bereichen erheblich [10]. Durch die Nutzung von CNNs können innerhalb von Millisekunden Alltagsgegenstände mit Smartphones erkannt werden [11]. Im medizinischen Bereich werden Neuronale Netze angewendet, um z. B. Hautkrebs zu erkennen [12] oder dem Arzt, entsprechend seiner operativen Handlung, das richtige Werkzeug zu reichen [13]. Des Weiteren sind CNNs ein wesentlicher Baustein im autonomen Fahren, um Fußgänger, Radfahrer oder Ampeln im dreidimensionalen Raum zu detektieren [14]. CNNs sind ebenfalls für die automatisierte Erkennung von visuellen Schäden an Brücken hervorragend geeignet.

Der folgende Beitrag zeigt (*i*) das Konzept einer digitalen Bauwerksprüfung, (*ii*) Grundlagen zu Neuronalen Netzen und ihre Ergebnisse auf einem Open-Source-Datensatz sowie (*iii*) die Erstellung und den aktuellen Stand des eigenen Datensatzes, um CNNs das pixelgenaue Erkennen von Schäden zu lehren.

2 Digitale Bauwerksprüfung

Die Grundlage für eine digitale Bauwerksprüfung stellt ein Building Information Model (BIM) dar, das während der Planungsphase oder, im Falle von Bestandsbrücken, vor der Inspektion generiert wird (Bild 2). Am Bauwerk erfassen die Inspizierenden die Schäden handnah im Rahmen einer Hauptprüfung, oder ausschließlich visuell, im Zuge einer einfachen Prüfung [6]. Bei Inspektionen, die nicht handnah erfolgen, ist der Einsatz von Drohnen (engl.: Unmanned Aerial Vehicles, UAV) zur Erfassung der Schäden möglich [15]. Für die handnahe Prüfung sind mobile Endgeräte zur Unterstützung der Inspizierenden am geeignetsten, da diese handlich und einfach zu bedienen sind. Der zweite wesentliche Punkt einer DI ist die automatisierte Schadenserkennung. Entsprechend dem Prozessablauf in Bild 2 zählen zum vollständigen Erkennen eines Schadens dessen Klassifikation, Vermessung, Lokalisierung und Evaluation. Nur mit diesen Informationen können die Zuweisung einer Beispiel-ID gem. RI-EBW-PRÜF [7] sowie eine Bewertung hinsichtlich Standsicherheit, Verkehrssicherheit und Dauerhaftigkeit erfolgen. Die Eigenschaften der erkannten Fehlstellen werden bei jeder DI als Metadaten gespeichert. Infolgedessen kann das BIM mit den Metadaten durchgeführter Inspektionen fusioniert werden, wodurch die Generierung eines Digitalen Zwillings zu jeder Prüfung ermöglicht wird. Dadurch kann auch die Entwicklung des Brückenzustands über die Zeit betrachtet werden.

Die derzeit eingesetzten Produkte und Konzepte zur digitalen Bauwerksprüfung bestehen meist aus vier konsekutiven Schritten: Drohnenflug, Erstellung eines 3D-Modells mithilfe von Fotogrammmetrie, automatisierte Schadenserkennung und Speichern der Daten in einer Datenbank [16, 17]. Dies hat zur Folge, dass zu jeder Inspektion ein neues 3D-Modell erzeugt wird und dadurch mit jeder Prüfung das Datenvolumen erheblich ansteigt. Zudem werden die Prüfenden im Rahmen dieses Konzepts erst nach der Inspektion des Bauwerks im Büro in den Prozess der Schadensanalyse einbezogen. Sollte bspw. die visuelle Aufnahme eines Schadens zur eindeutigen Beurteilung nicht ausreichen, besteht keine unmittelbare Möglichkeit, zusätzliche haptische oder auditive Information des entsprechenden Schadens zu erfassen.



Bild 2 Prozessablauf einer digitalen Bauwerksprüfung mit einem mobilen Endgerät
Workflow during a digital inspection with a mobile device

Zudem ist das Erfassen nicht visuell erkennbarer Schäden generell nicht möglich.

Der in Bild 2 beschriebene Prozess einer DI in Kombination mit einem mobilen Endgerät setzt sich als Ziel, im Rahmen existierender Normung, Richtlinie und Anweisung [6, 7, 18] die Prüfenden bei der Durchführung von einfachen Prüfungen und Brückenhauptprüfungen maximal zu unterstützen. Zudem birgt dieser Prozess den Vorteil, dass er die handnahe Prüfung vereinfacht und beschleunigt, während die Inspizierenden zentraler Bestandteil des Prozesses bleiben (engl.: Human Oversight). Die Beschleunigung der Inspektion folgt hauptsächlich aus der automatisierten Schadenserkennung und dem Wegfall der händischen Dokumentation (Ort, Art und Ausmaß eines jeden Schadens) in der Schadensskizze. Dabei liefert die Software zu jedem erkannten Schaden einen Vorschlag zu dessen Eigenschaften. Diese Empfehlung kann von Prüfenden am Bauwerk bei Bedarf korrigiert und gespeichert werden. Somit erfolgt ein direktes Qualitätsmanagement, was besonders bei Schäden wichtig ist, deren Klassifikation nicht visuellen Kontextwissens bedarf. Außerdem steigt die Datenmenge durch das einmalige Erstellen des BIM im Laufe der Prüfungen nur infolge der Aufsummierung der Metadaten an.

3 Bildklassifikation mit Künstlichen Neuronalen Netzen

3.1 Allgemein

Künstliche Neuronale Netze können viele Problemstellungen exakter als andere Algorithmen des Bereichs KI (lineare Regression, Entscheidungsbaum etc.) lösen. Dabei ist die Bildklassifikation eine komplexe Aufgabe, zu deren Lösung sich insbesondere CNNs eignen. Die Architektur von CNNs beruht auf biologischen Nervenzellen, sogenannten Neuronen. Dabei ist die Besonderheit der CNNs gegenüber anderen Algorithmen des KI-Bereichs, dass die enthaltenen Nodes (Neuronen) gewichtete Informationen zu benachbarten Neuronen weiterleiten. Diese Gewichte (Synapsen) sind trainierbar, wodurch ein übertragenes Signal verstärkt oder abgeschwächt werden kann [19]. Wesentliche Meilensteine der Bildklassifizierung mit Neuronalen Netzen sind die Datensätze MNIST [10], ImageNet [20] und COCO [21] sowie die Netzwerkarchitekturen LeNet [10] AlexNet [22] und ResNet [23].

Bild 3 zeigt den Ablauf des Forward-Prozesses, der bei der Klassifizierung eines Schadens durchlaufen wird. Im ersten Schritt, dem Preprocessing, werden die Bilder skaliert, zurechtgeschnitten und normalisiert. Im Anschluss wird jedes Pixel eines jeden Farbkanals in das Neuronale Netz gegeben. Die im CNN enthaltenen Convolutional-, Pooling- und Normalization-Layer erzeugen Feature Maps. Die Features repräsentieren die im jeweiligen Bild erkannten Muster und dienen daraufhin als Input für die linearen Layer im Klassifizierer bzw. Head. Durch das Anwenden einer Aktivierungsfunktion auf die Nodes der letzten Schicht des Klassifizierers werden schließlich die Wahrscheinlichkeiten zu den jeweils vorhandenen Schäden berechnet (vgl. Abschn. 3.3).

3.2 Training Neuronaler Netze

Die Entwicklung eines CNN, das mit einer überwachten Lernmethode trainiert wird, kann in drei Hauptbestandteile gegliedert werden: Trainieren, Validieren und Testen (Bild 4). Dabei wird für jeden Schritt ein separater Daten-Split verwendet. Üblicherweise werden 80 % der Daten für das Training und jeweils 10 % für das Validieren und Testen verwendet. Das Optimieren der Netzwerkarchitektur ist nicht Gegenstand dieses Aufsatzes (vgl. hierzu [24]). Entsprechend der gewählten Gesamtzahl an Trainingsepochen (e_ges) wiederholt sich der Kreislauf aus Trainieren und Validieren. Der Trainingsprozess besteht aus dem Forward-Prozess (vgl. Bild 3), der Berechnung des Trainingsfehlers, der Backpropagation und der Gewichtsanpassung. Dabei beschreibt der Trainingsfehler, wie nah die berechneten Klassenwahrscheinlichkeiten dem Soll-Ergebnis sind. Der Fehler wird aufgrund der Kettenregel partiell nach den Gewichten abgeleitet, um den jeweils zugehörigen Gradienten zu finden und die Gewichte des Netzes anschließend zu aktualisieren, was ein Lernen des CNN zur Folge hat. Während des Validierens hingegen erfolgt nach dem Forward-Prozess lediglich die Berechnung des Validierungsfehlers. Das Validieren ermöglicht das Bewerten der Leistung des Modells auf unbekannte Daten bzw. Daten, auf die das Modell nicht trainiert wurde. Der Validierungsfehler ist die entscheidende Ergebnisgröße für das Optimieren der Hyperparameter, die es zu minimieren gilt. Die Hyperparameter steuern u. a., wie viele Daten je Forward-Prozess und damit in das Berechnen der Fehlerfunktion inklusive anschließender Gewichtsanpassung mit einfließen (Batch-Größe) oder wie stark die Gewichte je Trainingsepoche

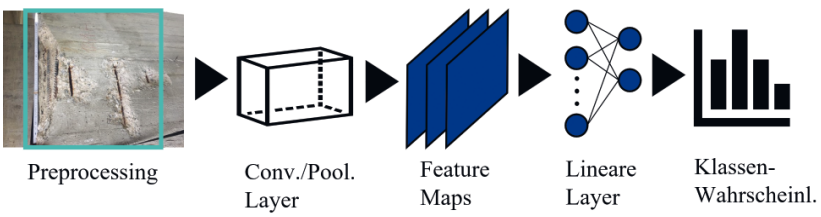


Bild 3 Schematische Darstellung des Forward-Prozesses zur Klassifizierung eines Schadensbilds mit einem CNN
Forward-propagation-process for the classification of a defect image with a CNN

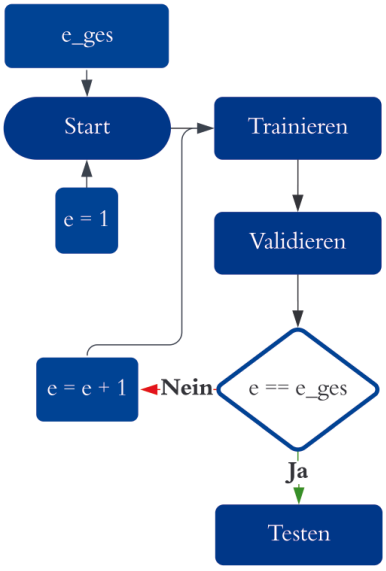


Bild 4 Ablaufdiagramm des Trainingsprozesses für ein CNN
Flowchart for the training of a CNN

angepasst werden (Lernrate). Sobald die festgelegte Anzahl an Trainingsepochen (e_ges) erreicht ist, kann das Testen des Modells erfolgen. Dabei werden Metriken wie die Exact Match Ratio und die klassenweise Sensitivität aufgrund der Ergebnisse auf den Testdatensatz ermittelt. Diese Ergebnisse werden herangezogen, um die Leistung des Neuronalen Netzes endgültig zu beschreiben. Das Testen eines Netzwerks erfolgt, nachdem die idealen Hyperparametereinstellungen bzw. der minimale Validierungsfehler für die jeweilige Architektur erreicht wurde [25].

Für das Trainieren von CNNs zur Erkennung von Schäden an Massivbauwerken stehen derzeit sechs Open-Source-Datensätze zur Verfügung (vgl. Tab. 1). Konsistente Daten-Splits, welche für ein Vergleichen der darauf trainierten Modelle unabdingbar sind, existieren lediglich bei der Hälfte der aufgeführten Datensätze. Vier davon sind in einem binären Klassifikationsbereich verfügbar, die zwischen Riss oder kein Riss, bzw. Schaden oder kein Schaden differenzieren. Die beiden Datensätze MCDS [5] und CODEBRIM [4] dagegen bieten die Möglichkeit, Modelle zur Multi-Target-Klassifikation (vgl. Abschn. 3.3) zu trainieren, wobei MCDS aufgrund der Aufbereitung durch Flotzinger et al. [24] in diesem Setting vorhanden ist. Hinsichtlich der Menge annotierter Bilder ist der CODEBRIM-Datensatz derzeit am größten. Deshalb werden die Ergebnisse im Abschn. 3.3 ausschließlich für diesen Datensatz präsentiert. Dem Datensatz entsprechend unterscheiden die Modelle zwischen den Klassen *Abplatzung, offenliegende Bewehrung, Korrosion, Ausblühung, Riss* und *Kein Schaden*.

Tab. 1 Open-Source-Datensätze der Massivbau-Schadens-Domäne; unterschieden wird zwischen den Datensatz-Typen: Binär (B), Multi-Target (MT) und Objektdetektion (OD); die Spalte „Splits" zeigt, ob im Originaldatensatz fixe Trainings-, Validierungs- und Test-Splits vorhanden sind
Open-source datasets for damage recognition on reinforced concrete bridges; dataset types are binary (B), multi-target (MT) and object detection (OD); splits refer to the presence of fix training, validation and test splits in the original publication

Name	Klassen	Typ	Größe	Splits
CDS [26]	2	B	1028	×
BCD [27]	2	B	5390	✓
SDNET2018 [28]	2	B	13.620	×
ICCD [29]	2	B	60.010	✓
MCDS [5]	8	MT	3607	×
CODEBRIM [4]	6	MT	7261	✓
BiNet [30]	4	OD	3588	×

3.3 Multi-Target-Klassifikation

Multi-Target-Klassifikation bedeutet, dass auf einem Bild mehrere Label bzw. Schadensklassen unabhängig voneinander durch ein CNN erkannt werden können. Bild 5 zeigt die von einem Neuronalen Netz bestimmten Wahrscheinlichkeiten der jeweiligen Schadensklassen zum nebenstehenden Bild. Der entsprechende Schaden gilt als festgestellt, sobald der Grenzwert von 0,5 überschritten ist. Demnach hat das Modell in diesem Beispiel die Schäden *Abplatzung* und *offenliegende Bewehrung* erkannt, jedoch die *Korrosion* der Bewehrungsstäbe nicht.

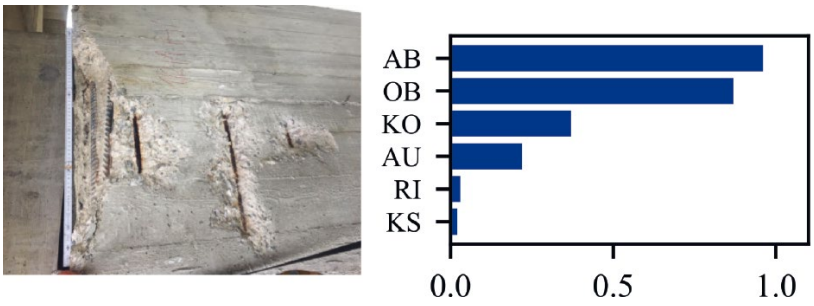


Bild 5 Wahrscheinlichkeiten der Klassifizierung (rechts) eines Schadensbilds (links) mit Abplatzungen (AB), offenliegender Bewehrung (OB) und Korrosion (KO); die restlichen Schadensklassen sind: Ausblühung (AU), Riss (RI) und Kein Schaden (KS)
Predicted probabilities of the defects (right) occurring on an image (left) including the defects: spalling (AB), exposed reinforcement (OB) and corrosion (KO); the other defects are efflorescence (AU), crack (RI) and no damage (KS)

Tab. 2 Testergebnisse in Prozent von ResNet50 (RN50) und MobileNetV3-Large (MNV3); Abkürzungen analog zu Bild 5
EMR and classwise recall in percent for ResNet50 (RN50) and MobileNetV3-Large (MNV3); abbreviations according to Fig. 5

CNN	EMR	Klassenweise Sensitivität					
		KS	RI	AB	AU	OB	KO
RN50	74	95	88	84	76	89	79
MNV3	69	94	79	83	70	89	77

Quelle: Flotzinger et al. [5]

Flotzinger et al. [24] stellten Untersuchungen zum Finden einer optimalen Transfer-Learning-Strategie für je drei CNNs sowie deren ideale Hyperparameter an. Die besten Ergebnisse konnten mit der Strategie Head-Then-All erzielt werden, wobei zunächst die linearen Layer (Head) am Ende des Netzes auf den Schadensdatensatz trainiert wurden und im Anschluss alle Parameter des Netzwerks. In Bezug auf die Wahl der Netzwerkarchitektur haben sich ResNet50 [23] und MobileNetV3-Large [31] als sehr gut bzw. gut (vgl. Tab. 2) erwiesen. Die Leistung des ResNet50-Netzes zeigt dabei eine Exact Match Ratio (EMR) von knapp 74 % (vgl. Tab. 2) und damit eine bessere Leistung als zuvor veröffentlichte Modelle [4]. Die EMR beschreibt den Anteil der gänzlich richtig klassifizierten Bilder im Verhältnis zur Gesamtmenge an Bildern. Die Klassifikation eines Bilds ist dann vollständig richtig, wenn kein Schaden bzw. Label zu viel, zu wenig oder falsch erkannt wird. Dies ist die anspruchsvollste Metrik im Kontext der Multi-Target-Klassifikation. Daneben gibt die klassenweise Sensitivität Auskunft darüber, auf wie vielen der im Datensatz vorhandenen Bilder, in denen das entsprechende Label vorhanden ist, der jeweilige Schaden korrekt erkannt wurde. Dies ist aus Ingenieurssicht eine wichtige Metrik, da so ersichtlich wird, welcher Schaden ungenügend erkannt wird. Hierbei ist es wichtiger, dass keine Schäden unerkannt bleiben, als dass nichtvorhandene Schäden klassifiziert werden. Die beste Sensitivität wird, unabhängig vom Modell, beim Label *Kein Schaden* und die schlechteste für *Ausblühung* festgestellt.

Um weitere Schlüsse aus den Ergebnissen ziehen zu können, erfolgt zunächst die Analyse der hinsichtlich EMR nicht exakt klassifizierten Bilder (Non Exact Matches, kurz: *NEMs*) gegenüber der Anzahl an Label, die auf einem Bild auftreten (Tab. 3). Außerdem wird die Gesamtzahl vorhandener Bilder im Trainings- und Testdatensatz betrachtet. Dabei werden ausschließlich die Ergebnisse des besten CNN, ResNet50, untersucht. Es kann festgestellt werden, dass das Modell Bilder mit vier und fünf Schäden einmal vollständig richtig klassifiziert. Die geringste Anzahl an falsch klassifizierten Bildern in Relation zur Menge im Datensatz kann für Bilder mit insgesamt drei Labeln beobachtet werden, gefolgt von denen, die einen Schaden beinhalten. Das Verhältnis aus Test- und Trainingsdaten beträgt durchschnittlich 10 %, wodurch eine Balance zwischen den beiden Splits gegeben ist. Die Anzahl der Bilder mit einem Label ist mit

Tab. 3 NEMs in Zusammenhang mit der bildspezifischen Schadensanzahl sowie entsprechende Bildmengen im Trainings- und Test-Split
NEMs in relation to defect count and number of samples in the train and validation split

Schäden/Bild	Train	Test	NEM
0	26	4	4
1	4124	461	92
2	853	76	51
3	930	81	10
4	75	7	6
5	5	3	3
Σ	6013	632	166

Tab. 4 Häufigste NEMs von Schadenskombinationen
Most frequent NEMs of damage combinations

Kombination	Train	Test	NEM	Fehlerbeschreibung
AB+OB	380	34	21	17x KO erk., 4x OB fehlt, 3x RI erk.
AB+OB+KO	750	78	8	5x KO fehlt, 4x RI erk., 4x AB fehlt
RI+AU	182	7	7	3x RI fehlt, 4x AU fehlt

4124 Stück im Trainingsdatensatz groß und damit überrepräsentiert. Bilder mit mehr als drei Label hingegen sind unterrepräsentiert, was die geringe Leistung des CNN an dieser Stelle erklärt.

Tab. 4 stellt die am häufigsten falsch bestimmten Schadenskombinationen mit mindestens zwei Labeln dar. Die schlechtesten Ergebnisse werden bei der Klassifikation von *Rissen* mit *Ausblühung* sowie *Abplatzung* mit *offenliegender Bewehrung* beobachtet. Von den sieben Bildern mit der Kombination aus *Riss* und *Ausblühung* wird jedes falsch klassifiziert. Infolge der Betrachtung der Bilder, in denen das CNN fälschlicherweise keine *Ausblühung* erkennt, ist festzustellen, dass meist nur eine geringe Weißfärbung des Betons mit dem menschlichen Auge (Bild 6 oben) erkennbar ist.

Laut dem Testergebnis erkennt das Modell bei der Kombination aus *Abplatzung* und *offenliegender Bewehrung* in 17 Fällen zusätzlich *Korrosion*. *Korrosion* ist gemäß der Ground Truth des Datensatzes auf diesen Bildern nicht vorhanden. Infolge der Analyse entsprechender Bilder wird deutlich, dass auf allen Bildern tatsächlich korrodierte Bewehrung (vgl. Bild 6) und bei einem Bild eine starke Überbelichtung vorliegt. Das CNN erkennt somit richtigerweise, entgegen den Labeln des Testdatensatzes, *Korrosion*. Infolgedessen wäre die EMR des Modells bei einem bereinigten Testdatensatz größer. Um die Vergleichbarkeit zu vorherigen Ergebnissen weiterhin zu gewährleisten, wird von einem Reinigen der Daten vorerst



Bild 6 NEMs mit den Labeln entsprechend der Ground Truth: Riss/Ausblühung (oben) und Abplatzung/offenl. Bewehrung (unten) [14]
NEMs according to the ground truth with the labels: crack/efflorescence (top line) and spalling/exp. reinforcement (bottom line) [14]

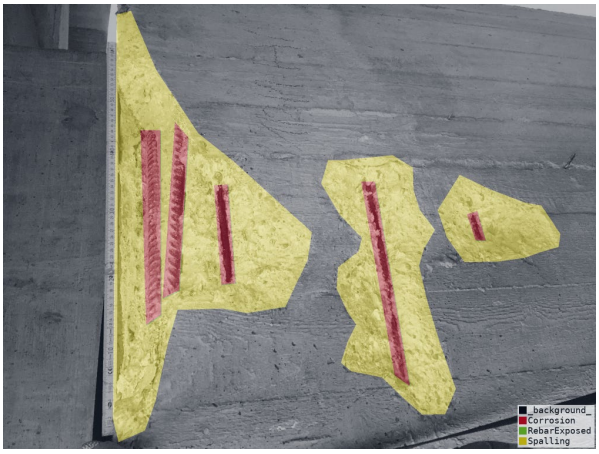


Bild 7 Annotiertes Schadensbild aus dem dacl-Datensatz
Annotated image of a defect from dacl-dataset

abgesehen. Die Leistung des Netzes bei der Kombination aus *Abplatzung*, *offenliegender Bewehrung* und *Korrosion* ist mit acht falschen aus insgesamt 78 Bildern zufriedenstellend.

Mit der Python-API Building Inspection Toolkit (bikit) [32] werden alle zuvor dargestellten Modelle open source zur Verfügung gestellt. Zudem sind über bikit alle veröffentlichten Datensätze der Domäne „Schäden an Massivbrücken“ leicht zugänglich. Die Datensätze sind – wenn nicht im Original vorhanden – in konsistente Datensplits unterteilt und bei Bedarf bereinigt. Zusätzlich sind die beiden wichtigsten Metriken, EMR und die klassenweise Sensitivität, über bikit ebenfalls erhältlich. Für das Verstärken des Wettbewerbs um die besten CNNs zur Bilderkennung von Schäden wurde zudem die Website dacl.ai [33] entwickelt. Dabei ist dacl das Akronym für den englischen Begriff „damage classification“, welcher für den folgenden Datensatz Namensgeber ist. dacl.ai dient als Plattform für das Vergleichen von Modellen der Massivbau-Schadensdomäne. Durch die Bestenliste der Internetseite, das sog. Leaderboard, ist leicht ersichtlich, wie sich die Leistung von Neuronalen Netzen zur Bilderkennung in dieser Domäne entwickelt hat. Außerdem besteht die Möglichkeit, eigene Ergebnisse hochzuladen und dadurch Teil der Bestenliste zu werden. Des Weiteren wurden die in Tab. 2 dargestellten Modelle auf mehreren Android Smartphones innerhalb der dacl-App [34] implementiert, um die Inferenzzeit verschiedener Modelle auf unterschiedlicher Hardware zu untersuchen.

4 dacl-Datensatz

4.1 Allgemein

Der dacl-Datensatz beinhaltet mit Polygonen annotierte Bilder im Kontext von Schäden an Massivbrücken. Aktuell umfasst der Datensatz 5663 Bilder mit insgesamt 16.043 Polygonen. Die Daten dienen dem Trainieren von Modellen zur Objektdetektion und semantischen Segmentierung und somit der nahezu pixelgenauen Erkennung und Verortung der Defekte auf Bildern. Die Erstellung des Datensatzes gliedert sich in die Bestandteile: Datenauswahl, -annotation und Qualitätsmanagement. Im Rahmen einer Vorselektion wird darauf geachtet, dass die Bilder, welche zum Annotieren weitergegeben werden, hauptsächlich relevante Daten enthalten. Aussortiert werden bspw. Bildserien von Landschaften sowie von Holz- und Stahlbrücken. Bei der anschließenden Annotation werden die Bilder betrachtet und die von den Bearbeitenden erkannten Schäden jeweils mit Polygonen umrandet und entsprechende Label zugewiesen (vgl. Bild 7). Zusätzlich werden Flags mit Checkboxes vergeben, die die Bildqualität beschreiben, um jene Bilder im Rahmen des Qualitätsmanagements auszusortieren.

4.2 Leitfäden zur Annotation

Vor Beginn der Datenannotation werden Leitfäden definiert, die ein exaktes Labeln durch die Bearbeitenden ermöglichen, welche ggf. kein Expertenwissen im Bereich „Schäden an Massivbrücken“ vorweisen. Der dacl-Datensatz verfügt über zwei Leitfäden: den Klassen- und den Polygon-Leitfaden. Der Klassen-Leitfaden basiert auf den visuell erkennbaren und einzigartigen Schäden der Schadensbeispiele der RI-EBW-Prüf [7], die an Brücken in Massivbauweise auftreten können (Tab. 5). Dieser Leitfaden beinhaltet und beschreibt alle zu vergebenden Label. Die Label der Polygone werden in drei Hauptgruppen unterteilt: Betonschadensklassen, Objektklassen und Objektschadensklassen. Die Betonschadensklassen umfassen Schäden, die ausschließlich auf Betonoberflächen auftreten, während die Objektklassen Brückenbestandteile umfassen, die nicht aus Beton sind und nicht zwingend einen Schaden aufweisen müssen. *Korrosion* ist der einzige Schaden, der an den Objekten annotiert wird. Hintergrund für die Nichtklassifizierung weiterer Schäden an Objekten (z. B. verdrehtes/verschobenes/gerissenes/be-

Tab. 5 Auszug aus dem Klassenleitfaden
Extract of the class-guideline

Label			Flags
Betonschadensklassen	**Objektklassen**	**Objektschadensklassen**	
Riss	Bewehrung	Korrosion	KeinSchadenKeinObjekt
Netzriss	Lager		Bildqualität_gut
Feuchtstelle	Üko		Bildqualität_schlecht
Ausblühung	Entwässerung		NichtAnnotieren
Rostfahne	Schutzeinrichtung		
KiesnestAuswaschungBetonkorrosion	Fugenband		
Hohlstelle	Beschilderung		
Abplatzung			
Graffiti			
Bewuchs			
Bewitterung			
Schalungsreste			
SchalungsMaterialPolystyrol			
Vogelkot			

wachsenes Fugenband, verschmutzte Übergangskonstruktion, unzulässige Schiefstellung/Verdrehung von Lagern etc.) ist deren komplexe Erkennung und der derzeitige Mangel an entsprechenden Daten. Das qualitative Bewerten der einzelnen Bilder ermöglichen Flags. „*KeinSchadenKeinObjekt*“ bedeutet, dass weder ein Schaden noch ein Objekt auf dem Bild vorhanden ist. Die Flag „*NichtAnnotieren*“ ist ein Indiz für sensible, meist personenbezogene Daten, die entweder im Rahmen des Qualitätsmanagements unkenntlich gemacht oder gelöscht werden. Eine schlechte Bildqualität ist dann festzustellen, wenn eine ungünstige Belichtung, Auflösung oder irrelevanter Inhalt vorliegt. Eine gute Bildqualität hingegen setzt voraus, dass alle enthaltenen Schäden eindeutig klassifizierbar und abzugrenzen sind. *Hohlstellen*, welche nicht visuell erkennbar sind, werden indirekt anhand von Kreidemarkierungen annotiert. Das Neuronale Netz erlernt somit, dass mit Kreide gezeichnete, meist kreisförmige Formen ein Indiz für die Schadensklasse Hohlstelle sind.

Der Polygon-Leitfaden legt fest, in welchem Detaillierungsgrad die Polygonpunkte zu setzen sind. Eine einheitliche Feinheit der Polygone über den kompletten Datensatz hinweg ist von großer Bedeutung für die Qualität bzw. Homogenität des Datensatzes. Der Polygon-Leitfaden setzt sich aus Positiv- und Negativ-Beispielen zusammen und wird fortlaufend mit den häufig begangenen Fehlern der Annotierer:innen aktualisiert.

4.3 Stand des dacl-Datensatzes

Bild 8 zeigt die Verteilung der vergebenen Flags. 4360 Bilder erhalten eine positive Bewertung hinsichtlich ihrer Qualität, während 612 Bildern eine schlechte Qualität attestiert wird. Eine mit 1247 geringe Anzahl irrelevanter Bilder, auf denen weder Schäden noch Objekte oder sensible Daten vorhanden sind, lässt auf eine insgesamt gute Qualität des Datensatzes schließen. Die in Bild 9 dargestellte Labelverteilung zeigt, dass überproportional viele Schäden, wie *Korrosion*, *Abplatzung*, *Ausblühung*, *Riss*, *Rostfahnen*, *Bewehrung* und *Schutzeinrichtung* annotiert werden. Andere Klassen wie *Schalungsreste*, *Beschilderung*, *Kein Schaden*, *Vogelkot* und *Netzrisse* hingegen sind unterrepräsentiert. Aufgrund dieser Erkenntnisse werden die beiden Label *Schalungsreste* und *SchalungsMaterialPolystyrol* zusammengefasst und andere Klassen, welche unter dem Schwellenwert von 100 Labeln liegen, aus dem Datensatz entfernt.

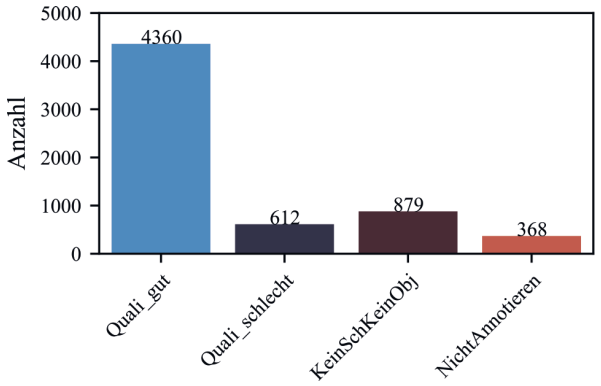


Bild 8 Übersicht über vergebene Flags im dacl-Datensatz
Counts of flags annotated in the dacl-dataset

5 Zusammenfassung und Ausblick

Die digitale Transformation der Bauwerksprüfung an Brücken kann von großer Bedeutung in Hinblick auf eine sichere und wirtschaftliche Infrastruktur sein. Eine wesentliche Effizienzsteigerung bei digitalen Prüfungen ist durch die automatisierte Klassifizierung von Schäden

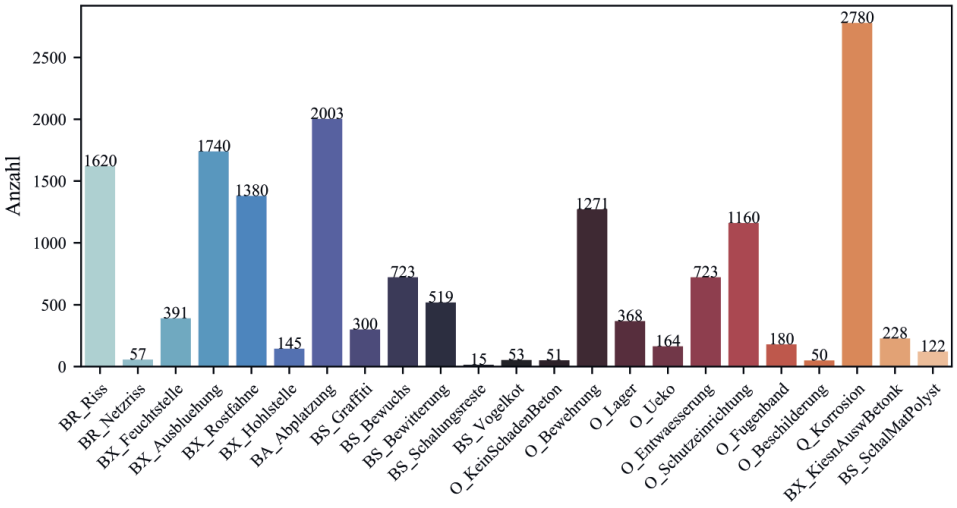


Bild 9 Übersicht über Mengen vergebener Label im dacl-Datensatz
Counts of labels annotated in the dacl-dataset

durch Künstliche Neuronale Netze möglich. Die Autoren zeigen, wie Neuronale Netze zum Erkennen von Defekten an Massivbauwerken trainiert werden. Außerdem wurden Testergebnisse der CNNs analysiert, um zukünftige Trainingsprozesse zu verbessern. Die erzielten Ergebnisse bestätigen die These, dass die Muster von Schäden an Massivbrücken trainiert werden können, jedoch sind bis zu deren vollständiger Erkennung weitere Verbesserungen vorzunehmen. Das größte Verbesserungspotenzial birgt ein größerer, die Realität besser abbildender Datensatz. Der hierfür vorgesehene dacl-Datensatz beinhaltet, im Gegensatz zu den existierenden Open-Source-Datensätzen, Bilder aus durchgeführten Bauwerksprüfungen nach DIN 1076 [6] und ermöglicht durch die Annotation mit Polygonen das Verorten der Schäden auf den Bildern. Ferner demonstriert der dacl-Datensatz, wie die Schadensklassen und Qualität der Bilder, die bei Prüfungen erzeugt werden, verteilt sind. In zukünftigen Arbeiten wird der Datensatz weiter vergrößert, um Modelle zur Objektdetektion und semantischen Segmentierung zu trainieren. Darüber hinaus bedarf es im Rahmen des Qualitätsmanagements noch weiterer Iterationen, um die Datenpunkte zu säubern. Die Autoren sind zuversichtlich, dass infolge zukünftiger Arbeiten auch die noch fehlenden Punkte der Schadenserkennung mittels Deep Learning gelöst werden können.

Dank

Die Autoren danken dem Institut für Verteilte Intelligente Systeme für die bereitgestellte Rechenkapazität am GPU-Cluster „Monacum One“ der Universität der Bundeswehr München. Zudem danken die Autoren dem Bayerischen Staatsministerium für Wirtschaft, Landesentwicklung und Energie für die finanzielle Förderung des Forschungsprojekts MoBaP. Besonderer Dank gilt den Bauämtern und Ingenieurbüros, die große Mengen an Bilddaten zur Verfügung stellten.

Literatur

[1] Bundesministerium für Verkehr, Bau und Stadtentwicklung (2013) *Bauwerksprüfung nach DIN 1076 Bedeutung, Organisation, Kosten – Dokumentation* [online]. https://www.bmvi.de/SharedDocs/DE/Anlage/StB/dokumentation-bauwerkspruefung-nach-din-1076.pdf [Zugriff am: 31. Mai 2022]

[2] Bundesanstalt für Straßenwesen (2021) *Brückenstatistik* [online]. https://www.bast.de/DE/Statistik/Bruecken/Brueckenstatistik.pdf [Zugriff am: 31. Mai 2022]

[3] Zimmert, F.; Braml, T.; Keuser, M. (2020) *Decision-making for the efficient life cycle management of structures*. Structural Concrete 21, No. 4, pp. 1217–1226. https://doi.org/10.1002/suco.201900408

[4] Mundt, M. et al. (02.04.2019) *Meta-learning Convolutional Neural Architectures for Multi-target Concrete Defect Classification with the COncrete DEfect BRidge IMage Dataset*. https://arxiv.org/pdf/1904.08486

[5] Hüthwohl, P.; Lu, R.; Brilakis, I. (2019) *Multi-classifier for reinforced concrete bridge defects*. Automation in Construction 105, 102824. https://doi.org/10.1016/j.autcon.2019.04.019

[6] DIN 1076 (1999) *Ingenieurbauwerke im Zuge von Straßen und Wegen – Überwachung und Prüfung*. Berlin: Beuth.

[7] Bundesministerium für Verkehr und digitale Infrastruktur (2019) *Richtlinie zur einheitlichen Erfassung, Bewertung, Aufzeichnung und Auswertung von Ergebnissen der Bauwerksprüfungen nach DIN 1076 – RI-EBW-PRÜF Schadensbeispiele* 1.93

[8] WPM-Ingenieure GmbH (2022) *SIB-Bauwerke1.94.1.1* [Software]. https://sib-bauwerke.de/.

[9] Phares, B. M. et al. (2004) *Routine Highway Bridge Inspection Condition Documentation Accuracy and Reliability*. Journal of Bridge Engineering 9, No. 4, pp. 403–413. https://doi.org/10.1061/(ASCE)1084-0702(2004)9:4(403)

[10] Lecun, Y. et al. (1989) *Backpropagation Applied to Handwritten Zip Code Recognition*. Neural Computation 1, No. 4, pp. 541–551. https://doi.org/10.1162/neco.1989.1.4.541
[11] Google LLC (2022) *Google Lens* [Software]. https://play.google.com/store/apps/details?id=com.google.ar.lens&hl=en&gl=US [Zugriff am: 1. Jun. 2022]
[12] Badlani, S. et al. (2021) *Melanoma Detection Using Convolutional Neural Networks and Group Normalization*. 5th International Conference on Trends in Electronics and Informatics (ICOEI), pp. 1412–1416.
[13] Nwoye, C. I. et al. (2022) *Rendezvous: Attention mechanisms for the recognition of surgical action triplets in endoscopic videos*. Medical Image Analysis 78, No. 1, p. 102433. https://doi.org/10.1016/j.media.2022.102433
[14] Qian, R.; Lai, X.; Li, X. (2021) *3D Object Detection for Autonomous Driving: A Survey*. https://arxiv.org/pdf/2106.10823
[15] Hamdan, A.-H. et al. (2021) *A semantic modeling approach for the automated detection and interpretation of structural damage*. Automation in Construction 128, 103739. https://doi.org/10.1016/j.autcon.2021.103739
[16] Morgenthal, G. (09.03.2022) *INFRA//TWIN by Infralytica – Digitale Bauwerkserhaltung* [online]. https://infratwin.com [Zugriff am: 23. Jun. 2022]
[17] Karlusch, A.; Forstner, E. (11.03.2022) *STRUCINSPECT – Infrastructure Lifecycle Hub* [online]. https://strucinspect.com [Zugriff am: 23. Jun. 2022]
[18] Bundesministerium für Verkehr, Bau- und Stadtentwicklung – Abteilung Straßenbau (2013) *ASB-ING – Anweisung Straßeninformationsbank Segment Bauwerksdaten*.
[19] Wartala, R. (2018) *Praxiseinstieg Deep Learning – Mit Python, Caffe, TensorFlow und Spark eigene Deep-Learning-Anwendungen erstellen*. Heidelberg: O'Reilly.
[20] Deng, J. et al. (2009) *ImageNet: A large-scale hierarchical image database*. *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*. Miami, FL. Piscataway, N.J.: IEEE, pp. 248–255.
[21] Lin, T.-Y. et al. (01.05.2014) *Microsoft COCO: Common Objects in Context*. http://arxiv.org/pdf/1405.0312v3
[22] Krizhevsky, A.; Sutskever, I.; Hinton, G. E. (2017) *ImageNet classification with deep convolutional neural networks*. Communications of the ACM 60, No. 6, pp. 84–90. https://doi.org/10.1145/3065386
[23] He, K. et al. (10.12.2015) *Deep Residual Learning for Image Recognition*. https://arxiv.org/pdf/1512.03385
[24] Flotzinger, J. et al. (2022) *Building Inspection Toolkit: Unified Evaluation and Strong Baselines for Damage Recognition*. *2022 IEEE International Conference on Image Processing (ICIP)*. Bordeaux, FR. Piscataway, NJ: IEEE.
[25] Goodfellow, I. et al. (2018) *Deep Learning – Das umfassende Handbuch: Grundlagen, aktuelle Verfahren und Algorithmen, neue Forschungsansätze*. Frechen: mitp.
[26] Hüthwohl, P.; Brilakis, I. (2018) *Detecting healthy concrete surfaces*. Advanced Engineering Informatics 37, pp. 150–162. https://doi.org/10.1016/j.aei.2018.05.004
[27] Xu, H. et al. (2019) *Autonomous Bridge Crack Detection Using Deep Convolutional Neural Networks*. 3rd International Conference on Computer Engineering, Information Science & Application Technology (ICCIA 2019), pp. 274–284. https://doi.org/10.2991/iccia-19.2019.42
[28] Dorafshan, S.; Thomas, R. J.; Maguire, M. (2018) *SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks*. Data in brief 21, pp. 1664–1668. https://doi.org/10.1016/j.dib.2018.11.015
[29] Li, S.; Zhao, X. (2019) *Image-Based Concrete Crack Detection Using Convolutional Neural Network and Exhaustive Search Technique*. Advances in Civil Engineering 2019, pp. 1–12. https://doi.org/10.1155/2019/6520620
[30] Bukhsh, Z. A.; Anžlin, A.; Stipanović, I. (2022) *BiNet: Bridge Visual Inspection Dataset and Approach for Damage Detection* in: Pellegrino, C. et al. [eds.] *Proceedings of the 1st Conference of the European Association on Quality Control of Bridges and Structures: EUROSTRUCT 2021*. Cham: Springer, pp. 1027–1034.
[31] Howard, A. et al. (06.05.2019) *Searching for MobileNetV3*. https://arxiv.org/pdf/1905.02244
[32] Rösch, P. J.; Flotzinger, J. (2022) *bikit (building-inspection-toolkit)* [online]. https://github.com/phiyodr/building-inspection-toolkit [Zugriff am: 24. Mai 2022]
[33] Flotzinger, J.; Rösch, P. J. (2022) *dacl.ai – Leaderboard for Models Tackling the Problem of Damage Classification on Built Structures* [online]. https://dacl.ai/ [Zugriff am: 24. Mai 2022]
[34] Flotzinger, J. et al. (2022) *Applying Automated Damage Classification During Digital Inspection of Structures* in: Zingoni, A. [Ed.] *Current Perspectives and New Directions in Mechanics, Modelling and Design of Structural Systems: Proceedings of the 8th International Conference on Structural Engineering, Mechanics and Computation (SEMC 2022)*. September 5–7, 2022, Cape Town, South Africa. Milton: CRC Press LLC.

Autoren



Johannes Flotzinger, M.Eng. (Korrespondenzautor)
johannes.flotzinger@unibw.de
Universität der Bundeswehr München
Fakultät für Bauingenieurwesen und Umweltwissenschaften
Institut für Konstruktiven Ingenieurbau
Werner-Heisenberg-Weg 39
85577 Neubiberg



Prof. Dr.-Ing. Thomas Braml
thomas.braml@unibw.de
Universität der Bundeswehr München
Fakultät für Bauingenieurwesen und Umweltwissenschaften
Institut für Konstruktiven Ingenieurbau
Werner-Heisenberg-Weg 39
85577 Neubiberg

Zitieren Sie diesen Beitrag

Flotzinger, J.; Braml, T. (2022) *Automatisierte Klassifizierung von Schäden an Massivbrücken mittels Neuronaler Netze.* Beton- und Stahlbetonbau. 117, H. 10, S. 786–794.
https://doi.org/10.1002/best.202200068

Dieser Aufsatz wurde in einem Peer-Review-Verfahren begutachtet. Eingereicht: 25. Juni 2022; angenommen: 17. August 2022.

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Research paper

dacl1k: Real-world bridge damage dataset putting open-source data to the test

Johannes Flotzinger [a,*], Philipp J. Rösch [b], Norbert Oswald [b], Thomas Braml [a]

[a] Institute for Structural Engineering, University of the Bundeswehr Munich, Werner-Heisenberg-Weg 39, Neubiberg, 85577, Bavaria, Germany
[b] Institute for Distributed Intelligent Systems, University of the Bundeswehr Munich, Werner-Heisenberg-Weg 39, Neubiberg, 85577, Bavaria, Germany

ARTICLE INFO

Keywords:
Building inspection
Damage recognition
Computer vision

ABSTRACT

Recognising reinforced concrete defects (RCDs) is a crucial element for determining the structural integrity, traffic safety and durability of bridges. However, most of the existing datasets in the RCD domain are derived from a small number of bridges acquired in specific camera poses, lighting conditions and with fixed hardware. These limitations question the usability of models trained on such open-source data in real-world scenarios.

We address this problem by testing such models on our "dacl1k" dataset, a highly diverse RCD dataset for multi-label classification based on building inspections including 1,474 images. Thereby, we trained the models on different combinations of open-source data (meta datasets) which were subsequently evaluated both extrinsically and intrinsically. During extrinsic evaluation, we report metrics on dacl1k and the meta datasets. The performance analysis on dacl1k shows practical usability of the meta data, where the best model shows an Exact Match Ratio of 32%. Additionally, we conduct an intrinsic evaluation by clustering the bottleneck features of the best model derived from the extrinsic evaluation in order to find out, if the model has learned distinguishing datasets or the classes (RCDs) which is the aspired goal. The dacl1k dataset and our trained models will be made publicly available, enabling researchers and practitioners to put their models to the real-world test.

1. Introduction

Against the backdrop of an ageing structure stock as well as the steady increase in heavy traffic, regular and high-quality bridge inspections are indispensable. Simultaneously, affected countries hold to inspection processes that are out-of-date while being confronted with staff shortages. The final goal of building inspections is the building assessment included in the inspection report. Within this document the damage-informations and -valuations as well as consequential actions (e.g. restoration works, traffic load limitations or bridge closures) are recorded. The recommended actions are determined by the damage-valuation which is based on the damage-information, in addition to the inspector's expertise. Thus, the damage-information is the decisive element. Thereby, each visually and acoustically (Hollowareas) recognisable defect is classified, measured and localised. However, in accomplishing this task, the use of computer vision approaches for acquiring the defect-information, within the framework of digitised inspections (DIs), offers great potential for improvement in terms of cost-effectiveness and quality control.

Major contributions in the field of damage recognition on built structures were made with the advent of datasets for the task of binary (Dorafshan et al., 2018; Hüthwohl and Brilakis, 2018; Xu et al., 2019; Li and Zhao, 2019), multi-class (Hüthwohl et al., 2019; Bianchi and Hebdon, 2021), multi-label classification (Mundt et al., 2019) as well as object detection (Mundt et al., 2019; DANG et al., 2021) and semantic segmentation (Benz and Rodehorst, 2022; Benz et al., 2019; Kulkarni et al., 2023; Flotzinger et al., 2024; Fujishima et al., 2023). Kulkarni et al. (2023) combined multiple image datasets of cracked and uncracked surfaces which is, to the best of our knowledge, the only work intersecting with the RCD domain, that is making use of a dataset compilation.

In general, the research area of reinforced concrete damage (RCD) recognition still faces the problem that only few datasets with mostly binary classifications tasks exist. The datasets are often small in terms of size and class variety. Moreover, the images are taken under restricted laboratory conditions. They use only one camera with fixed focal length and a specific acquisition setup concerning the relative pose of camera and objects as well as lighting conditions. Real-world data, in contrast, is strongly heterogeneous because of the big variety of building types, environmental conditions and image qualities depending on the hardware and the inspector. This opens up the questions: how do models

* Corresponding author.
E-mail address: johannes.flotzinger@unibw.de (J. Flotzinger).

https://doi.org/10.1016/j.engappai.2024.109106
Received 22 January 2024; Received in revised form 6 June 2024; Accepted 2 August 2024
Available online 16 August 2024
0952-1976/© 2024 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).



Fig. 1. Example images from the four analysed open-source datasets and our dacl1k dataset. Firstly, we train models based on results of previous research (Baseline training) and then apply five improvements to the training process (Improved training). Secondly, models are evaluated on datasets (Extrinsic) and the best model is further analysed (Intrinsic).

trained on existing RCD datasets perform on real-world data? How can existing open-source knowledge be exploited best and which existing datasets are useful?

This work introduces the dacl1k (damage classification) dataset, the first RCD multi-label dataset that originates from real building inspections including 1474 RCD images which were labelled by civil engineers inspired by German inspection standards. In order to train baseline models for dacl1k dataset, we conduct a proven transfer learning strategy, also called baseline training, and an improved training setting. For the improved setting, we examine five steps to leverage performance, choosing a different transfer learning approach, optimising the data augmentation, raising the training resolution, optimising the validation resolution and applying multi-label oversampling. Models resulting from both training strategies are evaluated on dacl1k. Hereby, we examine which open-source dataset combinations (we call them *meta* datasets) are useful for our real-world defect images. The model showing best performance during extrinsic evaluation is subsequently evaluated intrinsically through a bottleneck feature analysis to further investigate the cause of its performance. Thereby, we use dimensionality reduction techniques to identify, if the trained image representations are clustered according to data labels or data source.

To summarise, we make the following contributions: (*i*) dacl1k, the first RCD dataset for multi-label classification originating from real-world inspections, (*ii*) models that result from computational expensive hyperparameter search and several improvements, (*iii*) an extrinsic and intrinsic evaluation of the baselines. The dacl1k dataset, the meta datasets and the baselines will be made publicly available (see https://github.com/phiyodr/building-inspection-toolkit).

2. Related work

Dataset compilations. The Scene UNderstanding (SUN) database (Xiao et al., 2010) was generated by collecting images retrieved from multiple search engines that were proposed after seeking for terms that describe scenes, places, and environments. In the field of few-shot classification ten popular heterogeneous datasets, such as ImageNet (Russakovsky et al., 2015) and MSCOCO (Lin et al., 2014) were combined to one meta dataset while two of them were used for validation (Triantafillou et al., 2020). Others combined cross-domain and in-domain data to generate meta datasets for unlabelled, weakly-labelled or sparsely labelled target datasets (Ullah et al., 2022; Xue et al., 2023). To the best of our knowledge, Kulkarni et al. (2023) is the only work that combines previously available datasets, inter alia, from the RCD domain. They compile a semantic segmentation dataset, called CrackSeg9k, with 9255 images of cracks from ten sub datasets on various surfaces. Before unifying the datasets, their individual problems (e.g. noise and distortion) are addressed by applying Image Processing. SDNET (Dorafshan et al., 2018), which is used in the underlying work, is also part of CrackSeg9k for which no Image Processing was utilised.

Transfer learning. Bukhsh et al. (2021) analysed combinations of six RCD (binary and multi-label) datasets from which four are used in our analysis (see Section 3). They aimed to find the most valuable transfer learning dataset for each RCD dataset using a model initialised with and without weights from ImageNet.

The best performance for all datasets was obtained when using models initialised with ImageNet weights. On CODEBRIM (Mundt et al., 2019) the best performance was achieved when only weights from ImageNet were used, but no other RCD dataset. Thus, no additional training with an in-domain dataset was beneficial. MCDS (Hüthwohl et al., 2019), in contrast, benefited from training on the CODEBRIM dataset. The performance on each binary crack dataset, including SDNET (Dorafshan et al., 2018) and BCD (Xu et al., 2019), benefited from training with a different crack dataset.

Improvements. There are several levers that can be applied to enhance model performance, e.g. increasing or decreasing model capacity, regularising features or improving optimisation. Many improvements regarding model capacity for datasets in the RCD domain were examined in previous work. In Flotzinger et al. (2022) an extensive hyperparameter tuning for several state-of-the-art CNNs and transfer learning strategies were analysed. E.g., they examined different constellations of hidden layers in the classifier, optimiser types, learning rates and learning rate schedulers. Yet, only basic image augmentation was used. Unlike these approaches, we focus on the problem of regularisation by utilising basic image transformations, such as random horizontal and vertical flip, Random Erasing (Zhong et al., 2017) as well as automatic augmentation methods combined with a multi-label oversampling approach. There are several advanced augmentation algorithms (Random Erasing (Zhong et al., 2017), AugMix (Hendrycks

et al., 2020), AutoAugment (Cubuk et al., 2019), RandAugment (Cubuk et al., 2020), TrivialAugment (Müller and Hutter, 2021)) available, which use a large set of image mutations and help to improve classification performance. Most of these methods provide a set of augmentations which are applied at a number of so-called "strength bins". In addition, the range of their augmentation strengths can be defined. The final augmentation, Random Erasing, selects an arbitrary image region and overwrites its pixels with random values.

Furthermore, current work (Mundt et al., 2019) suggests to increase the train crop size in the RCD domain. Recent work demonstrated that directly integrating multiple resolutions inside the network at train and test time raises performance, especially in category-level detection (Lin et al., 2017).

Increasing evaluation resolution compared to training resolution improved performance in previous work (Touvron et al., 2019). This may be beneficial for our models, although, we do not expect a notable disparity in the size of objects observed by the network between train and test phase since our image transformation, in both phases, includes no random resizing and cropping but resizing and centre-cropping.

3. Datasets

For our experiments we use four open-source RCD datasets. We call each of their combination "meta dataset" from which three versions exist. Moreover, we introduce our real-world dataset dacl1k. This dataset is used for the evaluation of models trained on the meta data with respect to practical use. Example images are shown in Fig. 1.

Open-source datasets. We use four open-source datasets which are relevant to us. There are two binary crack datasets, **BCD** (Xu et al., 2019) and **SDNET** (Dorafshan et al., 2018). While BCD targets bridge cracks, SDNET includes images of cracks on walls, decks and pavement. Both are highly standardised datasets regarding hardware, object distance and camera angle which is orthogonal to the reference plane. Since SDNET has many incorrectly labelled data, we used a cleaned version (Rösch and Flotzinger, 2022).

The largest and most realistic dataset in terms of damage appearance is **CODEBRIM** (Mundt et al., 2019). The images were taken in a less standardised setting in comparison to BCD and SDNET. CODEBRIM and dacl1k share the same damage classes, which are very relevant for real-world inspections. There are two issues considering the practical transferability of this dataset to real-world scenarios. Firstly, CODEBRIM is made up from image crops, thus, single images are split into rectangular patches depending on the maximum size of the defects. In addition, undamaged surface is extracted to act as background (*No Damage*), leading to atypical image shapes. Long patches are often associated with cracks (see most left CODEBRIM image in Fig. 1) and second most frequently with long shaped exposed reinforcement bars. Secondly, due to this cropping approach, some resulting patches become very small. The minimum image height and width is 22 and 40 pixels respectively.

As the second dataset including multiple classes, we use an updated version (Rösch and Flotzinger, 2022) of **MCDS** (Hüthwohl et al., 2019). From originally eight classes, the labels "scaling" and "spalling" are merged, since they show the same defect type and only differ with regard to the cause of the damage. Moreover, the class "general" is removed which summarises graffiti and vegetation. This is done due to the fact that "general" neither represents severe damage nor can the non-existence of this class in the other datasets be guaranteed. Consequently, the other multi-class datasets would have to be screened for general defects and labelled to avert false labels in the meta dataset compilations.

dacl1k. We release a novel multi-label classification dataset called dacl1k. This dataset focuses on real-world inspection images in the RCD domain. While the heterogeneity of the dataset presents challenges for model training, it ensures that successful models have practical value in real-world scenarios. Our dataset includes five damage classes *Crack*, *Efflorescence*, *Spalling*, *Bars Exposed*, and *Rust* and the label *No Damage*. These classes are inherited from CODEBRIM with small changes to the original nomenclature. CODEBRIM is the most similar open-source dataset to dacl1k. In contrast to CODEBRIM and MCDS, we supply uncropped images. Examples of dacl1k images are displayed in Fig. 1 where in the first two rows of the image tiles the following defects are shown: *Cracks* with *Efflorescence* (top-left), *Crack* (top-right), *Spalling* with *Bars Exposed* and *Rust* (bottom-left) and *No Damage* (bottom-right). The dataset comprises a total of 1474 images, each with a unique set of challenges including variations in camera types, poses, lighting conditions, and resolutions. The total number of labels amounts to 2367. Our images derive from real inspections and were sourced from databases at authorities and engineering offices. We partitioned the dataset based on an equal label distribution into three subsets, with 67% for training, 13% for validation, and 20% for testing (see Fig. 2). Thus, dacl1k not only allows for testing but also training which is necessary due to lack of performance as described in Section 4.

Meta datasets. Our meta datasets are collections of aforementioned open-source data. The last three columns of Table 1 indicate in which meta dataset the according open-source dataset is included. In order to build the meta datasets, all datasets are transformed to a six-class dataset according to dacl1k. The datasets are sequentially merged with the aim of assessing the impact of each additional data batch. We start with CODEBRIM because it is the most realistic and currently largest multi-label RCD dataset. Subsequently, MCDS is added (*meta2*) since it has the same classes, but being smaller in size. Afterwards, we append BCD (*meta3*) to increase the amount of crack images and healthy surface slightly. Finally, we add SDNET to create *meta4* which is the meta dataset with the strongest class imbalance due to the high amount of binary crack data.

4. Models

This section provides an overview of the trainings conducted during the development of the baselines for dacl1k. First, we examine a proven transfer learning approach named baseline training (Flotzinger et al., 2022). Second, we apply improvement steps to the training pipeline to further leverage performance.

Baseline training. The settings of the default training are derived from previous work in RCD domain (Flotzinger et al., 2022). Here, three different CNN architectures and three transfer learning strategies were compared. Moreover, a computationally expensive hyperparameter search was conducted. In our baseline training we use their best MobileNetV3-Large (Howard et al., 2019) model. MobileNet represents the best trade-off between parameter count and performance for the underlying domain according to results from (Flotzinger et al., 2022). Furthermore, their best transfer learning strategy called "head then all" (HTA) is applied. Here, in the first step the model base is frozen and only the classification head is trained. In the second step, all parameters can be updated. They applied basic image augmentation including resizing, cropping, random rotating and flipping. The training and validation crop size is 224×224. The models are initialised with weights from ImageNet (Deng et al., 2009). Compared to Flotzinger et al. (2022), we adjust the learning rate after a grid search while evaluating on the meta4 dataset.

Improved training. To further improve model performance five additional setups are evaluated. Thereby, we focus on leveraging the performance of models fine-tuned on the most promising datasets, meta2+dacl1k and meta3+dacl1k as well as dacl1k itself. In order to improve the performance, we decided to examine another transfer learning approach applying different learning rates for model head and base (DHB) (Flotzinger et al., 2022; Howard and Ruder, 2018).

Furthermore, in initial experiments, we examined five different automatic augmentation methods: AugMix (Hendrycks et al., 2020), AutoAugment (Cubuk et al., 2019), RandAugment (Cubuk et al., 2020), TrivialAugment (Müller and Hutter, 2021) (TA) and TA with a custom

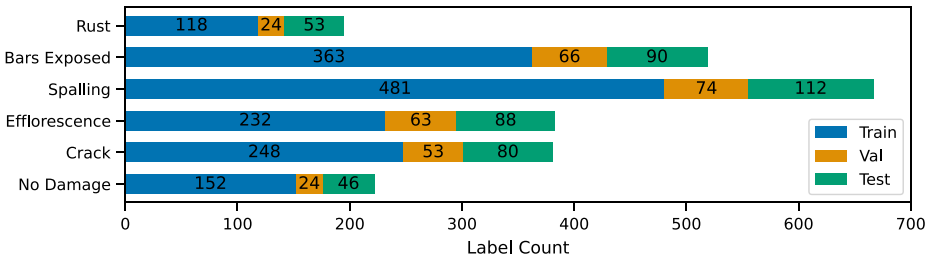


Fig. 2. Label distribution according to train, validation and test split in dacl1k.

Table 1
Dataset statistics of the four open-source datasets as well as dacl1k, displaying the number of classes, samples, image size (height × width) and their meta affiliation.

Dataset	Class	Samples	Image size (min, median, max)	meta2	meta3	meta4
CODEBRIM	6	7729	(22, 359, 3638) × (40, 826, 5997)	✓	✓	✓
MCDS	8	2597	(24, 356, 2830) × (47, 692, 4585)	✓	✓	✓
BCD	2	6069	(224, 224, 224) × (224, 224, 224)	–	✓	✓
SDNET	2	55 449	(256, 256, 256) × (256, 256, 256)	–	–	✓
dacl1k	6	1474	(245, 1024, 5152) × (336, 1365, 6000)	–	–	–

augmentation policy. The custom augmentation policy originates from our presumption that not all augmentations and their ranges inherited from the TA wide-custom augmentation space are beneficial to our data. Hence, we selected representative images from the dataset and evaluated each augmentation on the selection of data manually, based on subjective visual criteria. The custom policy neglects solarisation, posterisation and colourisation because they are considered as non-beneficial. Also, the minimum value of contrast augmentation is raised to 0.5 because the original range results in greying out the image completely. Before applying the augmentation method, random horizontal flip is applied and after the automatic augmentation method, Random Erasing (Zhong et al., 2017) is used. Supplementary, we execute a grid search for the ideal amount of magnitude bins in TrivAug. The search space is uniformly distributed in the interval 25 and 35. The best validation loss is obtained at a number of 34 magnitude bins.

In addition, training crop size is raised to 512 × 512 in expectation of losing less image information due to resizing. This may be beneficial to dacl1k with a relatively large resolution in comparison to the meta datasets (see Table 1). Another improvement step is the increase of the test resolution. Therefore, we test the models on different test crop sizes, starting from a size of 512 × 512 and subsequently adding 16 pixels until 656.

Finally, we tackle the problem of class imbalance by applying a multi-label oversampling strategy. This is especially useful when fusing multi-class datasets with cracks-only datasets, such as SDNET, to correct the predominance of images showing cracks. The algorithm upsamples minority classes regarding the following steps: (*i*) calculate class counts in the current dataset and then calculate the standard deviation based of the count values, (*ii*) randomly draw an image from the dataset and update counts and standard deviation, (*iii*) if the new standard deviation is reduced, this sample is added to the dataset, if not, another sample is drawn. This is repeated a fixed number of times.

As in the default setting, both learning rates (for head and base) were adjusted based on a grid search.

5. Evaluation

In the following, we examine which dataset combination is the most valuable for practical use. Moreover, the best training settings regarding the improvement steps are analysed. In Section 5.1 the models from the baseline and improved training are evaluated. Section 5.2 includes an intrinsic analysis of the model showing the best performance according to its extrinsic evaluation on dacl1k.

Table 2
EMR on the model's source test set (itself) and on dacl1k with the according classwise Recall on dacl1k. All values in percent.

Trained on	EMR		Recall on dacl1k					
	itself	dacl1k	NoDam.	Crack	Effl.	Spall.	BExp.	Rust
CODEBRIM	70.57	16.89	**73.91**	15.00	34.09	24.44	**9.43**	**38.39**
meta2	70.41	**17.35**	63.04	15.00	37.50	**28.89**	7.55	14.29
meta3	**81.52**	**17.35**	**73.91**	12.50	**40.91**	**28.89**	5.66	16.07
meta4	77.84	16.44	71.74	**21.25**	38.64	27.78	1.89	7.14
dacl1k	23.29	23.29	**65.22**	22.50	43.18	44.44	35.85	70.54
meta2+dacl1k	49.22	**27.85**	63.04	31.25	**53.41**	**61.11**	**41.51**	74.11
meta3+dacl1k	75.22	27.40	60.87	**32.50**	48.86	**61.11**	**41.51**	68.75
meta4+dacl1k	**76.81**	26.94	63.04	31.25	43.18	53.33	35.85	**75.89**

5.1. Extrinsic evaluation

We report the Exact Match Ratio (EMR) and classwise Recall to evaluate the models extrinsically. From a machine learning perspective, the EMR is a challenging metric for multi-label classification problems:

$$EMR = \frac{1}{n}\sum_{i=1}^{n} I(y_i = \hat{y}_i), \tag{1}$$

where n is the number of samples, I is the indicator function, y_i is the ground truth or target value and $\hat{y}_i$ is the predicted label. A correct prediction is only given if all classes of an image are classified correctly. In the underlying work all six damage predictions of a given sample have to match the ground-truth. This metric is provided for the according source datasets (*itself*) and for *dacl1k* in Table 2. The classwise Recall indicates how many defects of the according class are overlooked, which is especially interesting from a civil engineer's perspective. The Recall is also called True Positive Rate or Sensitivity. It is the ratio of the true positives, and the sum of true positives and false negatives.

We analyse results from models trained on CODEBRIM, meta2, meta3, meta4 and the combination with dacl1k datasets. It is important to note that the models are always trained once on one dataset. These datasets – treated as one set in one training step – are listed in the first column of Table 2. There is no incremental training on the sub-datasets of the meta datasets or their combinations with the dacl1k set performed.

Baselines results. The models trained on CODEBRIM and meta datasets show similar results, when evaluated on their own test split,

Table 3
Best improvement setting regarding augmentation (Aug), test crop size (TestCS) and multi-label oversampling (MO). EMR is reported on the test split of the datasets the models were trained on (itself). EMR and classwise Recall is reported for dacl1k test set.

Trained on	Improvements			EMR		Recall on dacl1k					
	Aug	TestCS	MO	itself	dacl1k	NoDam.	Crack	Effl.	Spall.	BExp.	Rust
dacl1k	default	528	–	31.51	31.51	**73.91**	**42.50**	52.27	68.89	56.60	68.75
meta2+dacl1k	custom	560	–	61.14	31.51	65.22	31.25	**54.55**	65.56	56.60	**74.11**
meta3+dacl1k	triv-aug	624	✓	**74.57**	**32.42**	67.39	36.25	50.00	**73.33**	**60.38**	**74.11**

compared to performances reported by others [REFS] ranging from 70.41% to 81.52% EMR (see upper part of Table 2). Weak performance is reported when the networks are evaluated on dacl1k (16.44% to 17.35%). The best results are obtained by meta2 and meta3.

The model trained on dacl1k achieves an EMR of 23.29% (see lower part of Table 2), which is approximately a 6 percent points improvement over meta2 or meta3 . When meta datasets and dacl1k are combined, the performance on dacl1k raises. Models benefit from the additional amount of real-world data. The classwise Recall shows weak performance for models solely fine-tuned on meta data. Especially the classes *Bars Exposed*, *Rust*, and *Crack* indicate that the domain shift between meta and target data is big. In other words, knowledge gathered from open-source data only, is in the current setting properly transferable to the real-world dataset. Like the analysis on EMR, only the combinations of dacl1k and meta data leverages Recalls compared to the dacl1k trained model. Considering all displayed metrics, training on meta2 together with dacl1k is the most promising approach, followed by meta3+dacl1k. Thus, we apply the improvements described in Section 4 to raise the model performance.

Improved results. Table 3 presents the best performances achieved after applying the improvements described in Section 4. All models show better results when being trained at a crop size of 512 compared to 224. Additionally, all models benefit from an increase in test resolution. The best performance was achieved by training on meta3+dacl1k in combination with TrivialAugment, a test crop size of 624 and multi-label oversampling (32.42%). This is about a 1 percent point increase in comparison to training on dacl1k dataset only. However, training on meta2+dacl1k did not lead to a better performance compared to the model trained on dacl1k only. With respect to the best result from the default training (see Table 2), EMR is increased by 4.57%. Regarding the classwise Recall, it can be stated that – apart from *Crack* and *Efflorescence* – good performance is achieved. For comparison, previous work (Flotzinger et al., 2022) reported on the balanced version of CODEBRIM an EMR of 74% and the following classwise Recalls: 95% (*No Damage*), 88% (*Crack*), 76% (*Efflorescence*), 89% (*Spalling*), 93% (*Bars Exposed*), 85% (*Rust*).

5.2. Intrinsic evaluation

Performance on our new dacl1k dataset is weak. Therefore, we want to analyse the capabilities of our best model intrinsically. We want to understand if our model mainly gained information from the image content or the dataset source. In the desired setting the model should be able to differentiate between image content, which are the six labels, and not between image sources.

Approach. Our implementation is as follows. We extract bottleneck features from our best model according to EMR on dacl1k (see Table 3) and a model initialised with ImageNet weights for all five datasets. We randomly keep 330 images per dataset to obtain an evenly distributed number of images per data source. Then, we run a non-linear dimensionality reduction from 960 to 2 dimensions using t-SNE (van der Maaten and Hinton, 2008). Here, we carefully select the t-SNE hyperparameters. We found a perplexity of 20, 5000 optimisation steps, and a learning rate of 200 useful.

Results. In the visualisation of our best model in Fig. 3 we see clear clusters for the *Crack* and *No Damage* class of the BCD dataset. Moreover, a dedicated cluster containing SDNET images only. Here, no clear distinction between the two classes is visible. Right to the centre, there is a mixed cluster for CODEBRIM and MCDS. No clear differentiation between the classes is recognisable. On the far right one cluster for dacl1k datasets exists. Compared to the ImageNet visualisation on top, denser clusters are visible. This shows that the learned features have adapted to the underlying RCD domain. However, the created clustering is still coarse and does not show clearly separated clusters according to damage types, but mainly relative to datasets. To summarise, our best model mainly learned to differentiate between datasets and not between image content.

5.3. Discussion

Our work refers to the portability of features learned by open-source RCD data to our real-world dataset dacl1k. During the development of our baselines, we face domain shifts regarding the underlying datasets. The first and largest shift is present between ImageNet and meta data because of the differences regarding their feature distribution and marginal distribution (6 vs. 1000 classes). The second and smaller shift appears as we combine the four different RCD datasets, which do not share the same marginal distribution by default (binary crack vs. multi-class). A third domain shift arises as we evaluate the models on real-world data while having pre-trained on ImageNet and trained – apart from the meta-dacl1k combinations – on meta data that is limited in terms of diversity (Weiss et al., 2016; Pan and Yang, 2010). The performance displayed by the extrinsic evaluation (see Tables 2 and 3) indicates that models have difficulties to predict the samples in the dacl1k test set. Furthermore, the intrinsic analysis shows that the latent image representation adapts to the RCD domain but does not sufficiently learn features in order to clearly differentiate between damage classes. Another reason for the underlying performance can be shortcuts, or rather decision rules, learned from the source dataset hindering generalisation. These shortcuts can lead to the failure of models, especially, when they are tested on real-world data (Geirhos et al., 2020). Additional experimental results and discussions are presented in Appendix C.

6. Conclusion

In this work, we presented dacl1k, a novel real-world dataset for the multi-label classification of defects occurring on massive bridges. We investigated compilations of open-source datasets and improved the training process significantly over previous work. Selected models were evaluated extrinsically and intrinsically. Results lead to the conclusion that in the RCD domain transferring knowledge from open-source data to real-world data shows weak performance. This also holds after having applied multiple improvement steps. The intrinsic evaluation – of the model showing the highest EMR – underlines that the meta and dacl1k image representation strongly differs from each other. Moreover, achieving a successful domain transfer between ImageNet, meta, and our dacl1k dataset requires further research. The applied improvement steps raised the performance significantly to an Exact Match Ratio of over 32%. Yet, this is still insufficient for practical use in the digitised inspection framework. The classwise Recall shows that, apart from *Crack* and *Efflorescence*, 60 to 74% of the real-world defects are recognised.

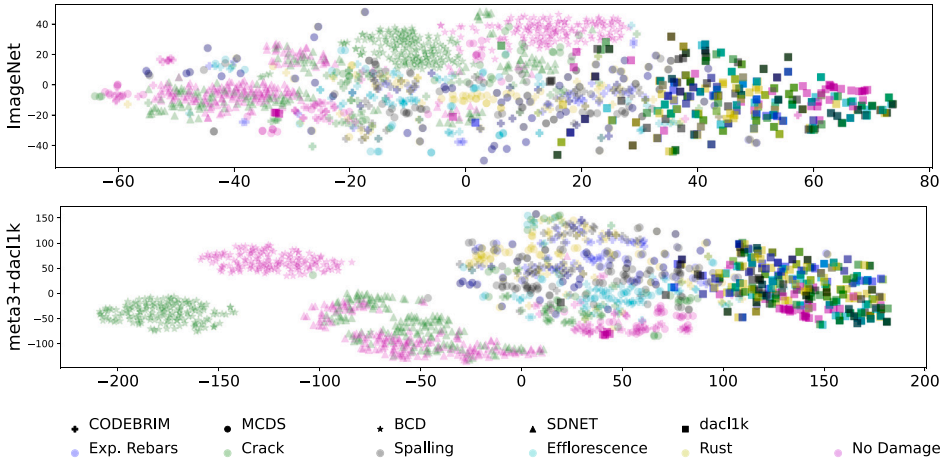


Fig. 3. Clustering of images from all five datasets using bottleneck features after t-SNE dimensionality reduction. Features come from our best model trained on meta3+dacl1k (see Table 3).

Our work can be a starting point to label further real-world data in a semi-automated fashion (Yu et al., 2015). Here, our real-world dataset and corresponding models can be used to pre-label unseen images. In a subsequent step, labels can be assigned automatically when a certain prediction probability has been reached. If this is not achieved, the image must be annotated manually. In addition, dacl1k enables estimating and comparing the usability of models for practical use. This is especially of great value for authorities which are confronted with products offering "damage recognition through AI". Our work acts as a benchmark for evaluating such applications.

CRediT authorship contribution statement

Johannes Flotzinger: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Philipp J. Rösch:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software. **Norbert Oswald:** Supervision. **Thomas Braml:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

All code and the dataset is available on GitHub, as mentioned in the paper.

Acknowledgements

We are most grateful to Heiko Neumann and Johannes Kreutz for their insights and suggestions. We thank the Bavarian Ministry of Economic Affairs for funding the MoBaP research project (IUK-1911-0004// IUK639/003) from which this work originates, the engineering offices and authorities for providing data. We gratefully acknowledge the computing time granted by the Institute for Distributed Intelligent Systems and provided on the GPU cluster Monacum One at the University of the Bundeswehr Munich.

Appendix A. Datasets

A.1. dacl1k

During labelling and quality assessment for dacl1k we followed a two stage annotation process. First, civil engineering students labelled the real-world inspection images after having completed an initial training. Additionally, a detailed class and labelling guideline as well as a sample batch of labelled data was handed out. On the one hand, the class guideline clearly describes each damage class by naming the abbreviation, a detailed description of the visual appearance and the defect cause (see Table A.4). On the other hand, the labelling guideline points out the caveats based on previous labelling processes. Defects often overlap with each other, which often led to false negative labels because only the most obvious defects were recognised. The most common case for overlapping defects is *Spalling* exposing the reinforcement (*Bars Exposed*) which is covered with *Rust*. Another common co-appearance is the combination of *Efflorescence* and *Crack*. The efflorescence stains complicate the detection of the subjacent cracks leading to many un-labelled cracks. Another error source is the presence of heavy weathering on surfaces that also show defects. After each initially submitted batch of data (≈100 images) the labelling team continually received feedback and subsequently repaired their labels.

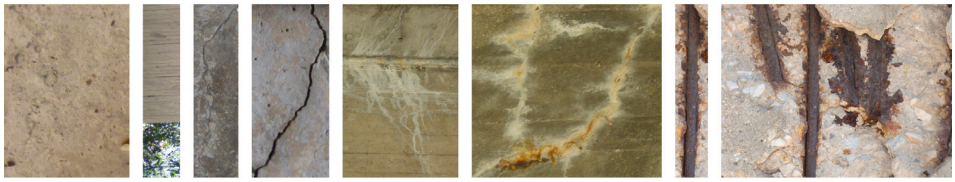
In a second quality assurance step, we appended samples showing *No Damage* because the original label distribution showed an underrepresentation of healthy concrete surfaces. This is especially important for testing the models with regards to false positives. Furthermore, Fig. A.4 provides an overview of dacl1k's and the meta datasets' images in high resolution. The unified nomenclature from dacl1k and the meta datasets' original label names are displayed in Table A.5.

A.2. Open-source datasets

This section provides details on data acquisition of the open-source datasets. The authors of **BCD** (Xu et al., 2019) used one camera. Based on our manual data validation, the samples in BCD were acquired under constant object distance, camera angle and lighting conditions. The bridge deck samples in **SDNET** (Dorafshan et al., 2018) stem from a number of bridge deck sections that were stored in a laboratory. Images of walls and pavements were taken on a university campus. They used one camera. The surface illumination was between 1500



(a) dacl1k. Top row, from left to right: *No Damage*, *No Damage*, *Crack*, *Crack*. Bottom row, from left to right: *Efflorescence* with *Crack*, *Efflorescence* and *Rust*, *Spalling* with *Bars Exposed* and *Rust*, *Rust*.



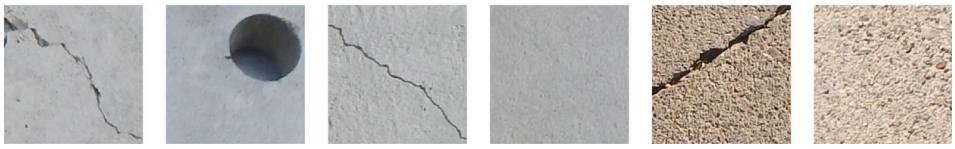
(b) CODEBRIM. From left to right pairs of: "Background", "Crack", "Efflorescence", "Spallation" with "ExposedBars" and "CorrosionStain".



(c) MCDS. One sample for each damage class. From Left to right: "NoDefect", "Cracks", "Efflorescence", "Scaling", "Spalling", "Exposed Reinfor." without "Rust Staining", "Rust Staining", "General".



(d) BCD. Three samples of the label "Crack" (left) and "NoCrack" (right).



(e) SDNET. Pairwise "Crack" (cracked) and "Uncracked" surfaces of different building parts, from left to right: wall, deck and pavement.

Fig. A.4. Detailed view of all five datasets used in our experiments. The sub-captions include the according dataset's original damage names (see Table A.5).

and 3000 lx. **CODEBRIM** (Mundt et al., 2019) includes images from 30 bridges. They tried to acquire data at varying distance and angles on purpose. In total, four different cameras were used. To homogeneously illuminate the darker bridge areas, they utilised a diffused flash. The authors of **MCDS** (Hüthwohl et al., 2019) aimed for consistent lighting conditions and a perpendicular camera angle to the surface during data acquisition. They used one camera in combination with a set of four lenses.

Appendix B. Training settings

In the following, additional information regarding the training procedure is documented, concentrating on parameters deviating from Flotzinger et al. (2022).

Baseline training. For the activation of the network's output layer, a Sigmoid layer is applied while using Binary Cross Entropy to compute the loss. For the first training step (HO), the learning rate is chosen according to a grid search, while evaluating on meta4 dataset that is the largest meta dataset. The learning rate is varied in the interval $1e^{-2}$ and $1e^{-4}$ which resulted in the best value at $5e^{-4}$. The second training step (HTA), which includes the training of all layers of the model, has a learning rate of $1e^{-5}$. We make use of the Adam optimiser with weight decay (Loshchilov and Hutter, 2017) together with a learning rate scheduler (cosine with warm-up). We trained the models for 100 epochs. The image pre-processing was held constant as follows: Resizing to 1.1*train resolution using bilinear interpolation and centre-crop according to the chosen test crop size. We repeated each training with the same setting two times at different seeds. Finally, the best of the two models with respect to the maximum EMR is reported.

Improved training. In contrast to the baseline training, for the improved training, 50 epochs were considered and the DHB approach was utilised. As in the default setting, both learning rates (for head

Table A.4
Descriptions of dacl1k's damage classes.

Damage	Description
No damage	No damage label describes images that contain healthy concrete surface or irrelevant content.
Crack	Cracks appear when the concrete's tensile strength is exceeded or during hardening, if the post-treatment or the concrete recipe was inadequate.
Efflorescence	Efflorescence is usually whitish, or yellowish. It appears when salts (calcium, sodium, potassium) of the cement stone get dissolved. Cement stone is the hardened cement paste (cement and water) that binds the other concrete components, sand and gravel (aggregate), together. This is a frequently occurring defect emerging when the according building part is constantly in contact with running water.
Spalling	Spalling can appear due to freeze thaw changes, corrosion of the subjacent reinforcement or impact, e.g. from cars that hit the structure.
Rust	Rust appears on metallic objects such as reinforcement and concrete. Rust that is visible on the concrete surface originates from neighbouring metallic parts or the subjacent reinforcement. Reinforcement can corrode as a result of loss of the alkaline protective layer provided by un-carbonated concrete. If the pH value drops due to the carbonation of the concrete, which is unavoidable over time, the reinforcement can oxidise.
BarsExposed	Reinforcement that is visible due to the spalling of the overlying concrete, which is usually the consequence of corrosion of the reinforcement. Another cause for visible reinforcement are rockpockets which arise if the cement paste did not fill all volume between the coarse aggregate. Rock pockets can follow, if the concrete's rheological properties or the compacting of the concrete was inadequate.

Table A.5
Our unified nomenclature in the first column and the original class names in BCD, SDNET, MCDS, and CODEBRIM. General damage class from MCDS is not considered.

meta4/dacl1k	BCD	SDNET	MCDS	CODEBRIM
NoDamage	NoCrack	Uncracked	NoDefect	Background
Crack	Crack	Crack	Cracks	Crack
Efflorescence	∅	∅	Efflorescence	Efflorescence
Spalling	∅	∅	Scaling; Spalling	Spallation
Rust	∅	∅	Rust Staining	CorrosionStain
BarsExposed	∅	∅	Exposed Reinfor.	ExposedBars
∅	∅	∅	General	∅

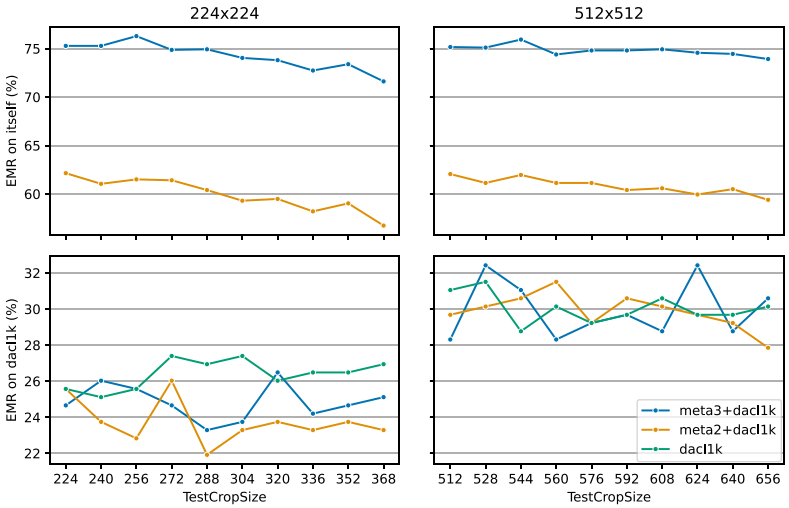


Fig. C.5. EMR at varying test crop size on the according model's test split (itself) and on dacl1k. Three differently trained models are evaluated (dacl1k, meta2+dacl1k and meta3+dacl1k). Each model was trained in compliance with the best training setting (see Table 3 in the main paper for resolution 512 × 512 and Table C.6 for 224 × 224). The charts share the y-axis row-wise and the x-axis column-wise.

and base) were adjusted based on a grid search. This led to a base learning rate of $1e^{-5}$ and a head learning rate of $1e^{-3}$. Regarding the data augmentation, it is important to note that before applying the automatic augmentation method, random horizontal flip is applied with a probability of 50%. After the augmentation method, Random Erasing (Zhong et al., 2017) is used with a probability of 10%. The exact setting of the used TrivAug parameters are shown in Table C.8.

Appendix C. Further extrinsic evaluation

In the following, further extrinsic results are reported. The displayed models were trained on images with a resolution of 224 × 224 (see Table C.6) and 512 × 512 (see Table C.7). We only show results for models trained and tested with the same resolution in the tables. The investigations regarding the test resolution are displayed in Fig. C.5.

Table C.6
Improvement setting regarding augmentation (Aug), and multi-label oversampling (MO) as well as test results on dacl1k (except EMR on itself) for models fine-tuned on three different datasets with a train resolution of 224 × 224. Underlined values represent the best results depending on the training dataset. Bold values represent the best overall result for the according metric.

Trained on	Improvements		EMR		Recall on dacl1k					
	Aug	MO	itself	dacl1k	NoDam.	Crack	Effl.	Spall.	Bexp.	Rust
dacl1k	default	–	25.11	25.11	58.70	27.50	43.18	47.78	43.40	72.32
		✓	24.20	24.20	67.39	28.75	51.14	44.44	60.38	73.21
	triv-aug	–	24.66	24.66	60.87	26.25	38.64	51.11	49.06	68.75
		✓	25.11	25.11	**71.74**	28.75	44.32	43.33	**69.81**	**75.89**
	custom1	–	23.74	23.74	63.04	27.50	43.18	43.33	50.94	70.54
		✓	25.57	25.57	69.57	37.50	50.00	43.33	71.70	67.86
meta2+dacl1k	default	–	62.42	23.29	56.52	31.25	55.68	58.89	50.94	54.46
		✓	63.34	21.92	54.35	21.25	72.73	55.56	49.06	67.86
	triv-aug	–	64.80	25.57	58.70	30.00	56.82	50.00	50.94	63.39
		✓	65.54	24.66	65.22	22.50	52.27	52.22	47.17	66.96
	custom	–	64.71	**26.03**	60.87	25.00	54.55	52.22	49.06	62.50
		✓	64.53	24.20	63.04	32.50	55.68	44.44	50.94	61.61
meta3+dacl1k	default	–	75.75	22.83	45.65	21.25	68.18	45.56	30.19	75.00
		✓	75.64	19.63	32.61	27.50	57.95	50.00	28.30	59.82
	triv-aug	–	77.17	25.11	63.04	22.50	46.59	**60.00**	33.96	63.39
		✓	77.41	23.29	50.00	32.50	**76.14**	36.67	47.17	61.61
	custom1	–	76.64	24.66	58.70	**36.25**	56.82	54.44	39.62	69.64
		✓	**78.06**	22.83	56.52	28.75	63.64	40.00	54.72	61.61

Table C.7
Improvement setting regarding augmentation (Aug), and multi-label oversampling (MO) as well as test results on dacl1k (except EMR on itself) for models fine-tuned on three different datasets with a train resolution of 512 × 512. Underlined values represent the best results depending on the training dataset. Bold values represent the best overall result for the according metric.

Trained on	Improvements		EMR		Recall on dacl1k					
	Aug	MO	itself	dacl1k	NoDam.	Crack	Effl.	Spall.	Bexp.	Rust
dacl1k	default	–	31.05	**31.05**	69.57	40.00	51.14	71.11	56.60	68.75
		✓	27.40	27.40	**71.74**	25.00	43.18	48.89	45.28	78.57
	triv-aug	–	30.14	30.14	69.57	32.50	45.45	52.22	47.17	74.11
		✓	28.77	28.77	**71.74**	35.00	53.41	43.33	49.06	68.75
	custom1	–	23.74	23.74	56.52	32.50	39.77	46.67	60.38	74.11
		✓	28.31	28.31	67.39	**42.50**	50.00	38.89	56.60	65.18
meta2+dacl1k	default	–	61.96	28.31	56.52	38.75	55.68	58.89	54.72	69.64
		✓	62.24	24.20	52.17	35.00	52.27	40.00	50.94	75.00
	triv-aug	–	63.61	28.77	67.39	35.00	51.14	65.56	62.26	70.54
		✓	63.34	29.22	65.22	30.00	**60.23**	55.56	60.38	72.32
	custom	–	63.43	29.68	63.04	33.75	59.09	71.11	60.38	72.32
		✓	63.06	26.48	63.04	28.75	57.95	46.67	58.49	73.21
meta3+dacl1k	default	–	74.51	24.66	54.35	25.00	44.32	64.44	47.17	**79.46**
		✓	75.16	25.57	45.65	31.25	54.55	68.89	**66.04**	67.86
	triv-aug	–	75.93	28.77	58.70	36.25	51.14	**74.44**	52.83	73.21
		✓	75.93	28.31	65.22	30.00	52.27	67.78	60.38	71.43
	custom1	–	76.17	28.31	56.52	32.50	46.59	70.00	49.06	72.32
		✓	**76.40**	24.20	43.48	30.00	**60.23**	63.33	52.83	62.50

Comparing both tables makes clear that training on a resolution of 512 × 512 shows better results on dacl1k compared to training on 224 × 224. This cannot be stated for models that are evaluated on their affiliated test set. Here, the EMR on itself is reduced by approximately 2% for meta+dacl1k and meta3+dacl1k. Comparing the on-itself EMR, the dacl1k model is the only one profiting from the higher train resolution. With respect to the bigger average resolution of samples in dacl1k, this is reasonable. The best EMR on dacl1k is achieved by the model trained on meta2+dacl1k making use of custom data augmentation (26.03%). The best performance of models trained on the 512 × 512 train crop size can be observed for the dacl1k model making use of no improvement step (31.05%).

Fig. C.5 shows the EMR of the models at varying test crop size. We analyse models trained on a resolution of 224 × 224 and 512 × 512, again, trained on dacl1k, meta2+dacl1k and meta3+dacl1k. Regarding the EMR on itself, it can be stated that the performance of the models trained with a train crop size of 224 × 224 is nearly identical to the performance of models trained with 512 × 512. Considering models trained with 512 × 512 images and tested on dacl1k, it can be stated

Table C.8
Our custom augmentation pipeline based on Trivial Augment Wide (Müller and Hutter, 2021) with manipulated range of contrast in bold.

	Augmentation	Range/Probability
Basic augmentation	Random horizontal flip	0.5
Trivial augment	Shear X	0.0–0.99
	Shear Y	0.0–0.99
	Translate X	0–32
	Translate Y	0–32
	Rotate	−135°–+135°
	Brightness	0.01–2.0
	Contrast	**0.5–1.8**
	Sharpness	0.5–1.8
Advanced augmentation	Random erasing	0.1

that the best resolution for meta2+dacl1k is achieved at 560 while for meta3+dacl1k two peaks at a value of 528 and 624 with the same EMR are visible. The model trained on dacl1k shows its best performance at a test crop size of 528.

References

Benz, C., Debus, P., Ha, H.K., Rodehorst, V., 2019. Crack segmentation on UAS-based imagery using transfer learning. In: 2019 International Conference on Image and Vision Computing New Zealand. IVCNZ, pp. 1–6. http://dx.doi.org/10.1109/IVCNZ48456.2019.8960998.

Benz, C., Rodehorst, V., 2022. Image-based detection of structural defects using hierarchical multi-scale attention. In: DAGM German Conference on Pattern Recognition. GCPR, Springer, pp. 337–353.

Bianchi, E., Hebdon, M., 2021. Bearing Condition State Classification Dataset. University Libraries, Virginia Tech, http://dx.doi.org/10.7294/16628698.

Bukhsh, Z.A., Jansen, N., Saeed, A., 2021. Damage detection using in-domain and cross-domain transfer learning. Neural Comput. Appl. 33, http://dx.doi.org/10.1007/s00521-021-06279-x.

Cubuk, E.D., Zoph, B., Mané, D., Vasudevan, V., Le, Q.V., 2019. AutoAugment: Learning augmentation strategies from data. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, pp. 113–123. http://dx.doi.org/10.1109/CVPR.2019.00020.

Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V., 2020. Randaugment: Practical automated data augmentation with a reduced search space. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. CVPRW, pp. 3008–3017. http://dx.doi.org/10.1109/CVPRW50498.2020.00359.

DANG, J., MIZUMOTO, T., jo CHUN, P., LIU, J., FUJISHIMA, T., 2021. Multi-type bridge damage detection method based on yolo. Artif. Intell. Data Sci. 2 (J2), 447–456. http://dx.doi.org/10.11532/jsceiii.2.J2_447.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., Fei-Fei, L., 2009. ImageNet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255. http://dx.doi.org/10.1109/CVPR.2009.5206848.

Dorafshan, S., Thomas, R.J., Maguire, M., 2018. SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks. Data Brief 21, 1664–1668. http://dx.doi.org/10.1016/j.dib.2018.11.015.

Flotzinger, J., Rösch, P.J., Braml, T., 2024. Dacl10k: Benchmark for semantic bridge damage segmentation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. WACV, pp. 8626–8635.

Flotzinger, J., Rösch, P.J., Oswald, N., Braml, T., 2022. Building inspection toolkit: Unified evaluation and strong baselines for bridge damage recognition. In: 2022 IEEE International Conference on Image Processing. ICIP, pp. 1221–1225. http://dx.doi.org/10.1109/ICIP46576.2022.9897743.

Fujishima, T., Dang, J., jo Chun, P., 2023. Training images for semantic segmentation of bridge damage detection. http://dx.doi.org/10.50915/data.jsceiii.24750210.v1.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., Wichmann, F.A., 2020. Shortcut learning in deep neural networks. Nat. Mach. Intell. 2 (11), 665–673. http://dx.doi.org/10.1038/s42256-020-00257-z.

Hendrycks, D., Mu, N., Cubuk, E.D., Zoph, B., Gilmer, J., Lakshminarayanan, B., 2020. AugMix: A simple method to improve robustness and uncertainty under data shift. In: International Conference on Learning Representations. URL https://openreview.net/forum?id=S1gmrxHFvB.

Howard, J., Ruder, S., 2018. Universal language model fine-tuning for text classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics, Melbourne, Australia, pp. 328–339. http://dx.doi.org/10.18653/v1/P18-1031, URL https://aclanthology.org/P18-1031.

Howard, A., Sandler, M., Chen, B., Wang, W., Chen, L.C., Tan, M., Chu, G., Vasudevan, V., Zhu, Y., Pang, R., Le, Q., Adam, H., 2019. Searching for MobileNetV3. In: Proceedings of the IEEE International Conference on Computer Vision. 2019-October, pp. 1314–1324. http://dx.doi.org/10.1109/ICCV.2019.00140, arXiv:1905.02244.

Hüthwohl, P., Brilakis, I., 2018. Detecting healthy concrete surfaces. Adv. Eng. Inf. 37, 150–162. http://dx.doi.org/10.1016/j.aei.2018.05.004.

Hüthwohl, P., Lu, R., Brilakis, I., 2019. Multi-classifier for reinforced concrete bridge defects. Autom. Constr. 105, http://dx.doi.org/10.1016/j.autcon.2019.04.019.

Kulkarni, S., Singh, S., Balakrishnan, D., Sharma, S., Devunuri, S., Korlapati, S.C.R., 2023. CrackSeg9k: A collection and benchmark for crack segmentation datasets and frameworks. In: Karlinsky, L., Michaeli, T., Nishino, K. (Eds.), Computer Vision – ECCV 2022 Workshops. Springer Nature Switzerland, Cham, pp. 179–195.

Li, S., Zhao, X., 2019. Image-based concrete crack detection using convolutional neural network and exhaustive search technique. Adv. Civ. Eng. 2019 (Ml), http://dx.doi.org/10.1155/2019/6520620.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017. Feature pyramid networks for object detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition. CVPR, pp. 936–944. http://dx.doi.org/10.1109/CVPR.2017.106.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft COCO: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (Eds.), Computer Vision – ECCV 2014. Springer International Publishing, Cham, pp. 740–755.

Loshchilov, I., Hutter, F., 2017. Decoupled weight decay regularization. In: International Conference on Learning Representations.

Müller, S.G., Hutter, F., 2021. TrivialAugment: Tuning-free yet state-of-the-art data augmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. ICCV, pp. 774–782.

Mundt, M., Majumder, S., Murali, S., Panetsos, P., Ramesh, V., 2019. Meta-learning convolutional neural architectures for multi-target concrete defect classification with the concrete defect bridge image dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR.

Pan, S.J., Yang, Q., 2010. A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22 (10), 1345–1359. http://dx.doi.org/10.1109/TKDE.2009.191.

Rösch, P.J., Flotzinger, J., 2022. Building inspection toolkit. https://github.com/phiyodr/building-inspection-toolkit.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., 2015. ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. (IJCV) 115 (3), 211–252. http://dx.doi.org/10.1007/s11263-015-0816-y.

Touvron, H., Vedaldi, A., Douze, M., Jégou, H., 2019. Fixing the train-test resolution discrepancy. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. pp. 8250–8260, URL https://proceedings.neurips.cc/paper/2019/hash/d03a857a23b5285736c4d55e0bb067c8-Abstract.html.

Triantafillou, E., Zhu, T., Dumoulin, V., Lamblin, P., Evci, U., Xu, K., Goroshin, R., Gelada, C., Swersky, K., Manzagol, P.-A., Larochelle, H., 2020. Meta-dataset: A dataset of datasets for learning to learn from few examples. In: International Conference on Learning Representations. URL https://openreview.net/forum?id=rkgAGAVKPr.

Ullah, I., Carrión-Ojeda, D., Escalera, S., Guyon, I., Huisman, M., Mohr, F., van Rijn, J.N., Sun, H., Vanschoren, J., Vu, P.A., 2022. Meta-album: Multi-domain meta-dataset for few-shot image classification. In: Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A. (Eds.), Advances in Neural Information Processing Systems. Vol. 35, Curran Associates, Inc., pp. 3232–3247, URL https://proceedings.neurips.cc/paper_files/paper/2022/file/1585da86b5a3c4fb15520a2b3682051f-Paper-Datasets_and_Benchmarks.pdf.

van der Maaten, L., Hinton, G., 2008. Visualizing data using t-SNE. J. Mach. Learn. Res. 9 (86), 2579–2605, URL http://jmlr.org/papers/v9/vandermaaten08a.html.

Weiss, K., Khoshgoftaar, T.M., Wang, D., 2016. A survey of transfer learning. J. Big Data 3 (1), 1–40. http://dx.doi.org/10.1186/s40537-016-0043-6.

Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A., 2010. SUN database: Large-scale scene recognition from abbey to zoo. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 3485–3492. http://dx.doi.org/10.1109/CVPR.2010.5539970.

Xu, H., Su, X., Wang, Y., Cai, H., Cui, K., Chen, X., 2019. Automatic bridge crack detection using a convolutional neural network. Appl. Sci. 9 (14), http://dx.doi.org/10.3390/app9142867, URL https://www.mdpi.com/2076-3417/9/14/2867.

Xue, Z., Yang, F., Rajaraman, S., Zamzmi, G., Antani, S., 2023. Cross dataset analysis of domain shift in CXR lung region detection. Diagnostics 13 (6), http://dx.doi.org/10.3390/diagnostics13061068, URL https://www.mdpi.com/2075-4418/13/6/1068.

Yu, F., Zhang, Y., Song, S., Seff, A., Xiao, J., 2015. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365.

Zhong, Z., Zheng, L., Kang, G., Li, S., Yang, Y., 2017. Random erasing data augmentation. arXiv:1708.04896.

5 Developing a Damage Segmentation Benchmark for Automated Bridge Inspection

Publications in this chapter:

- J. Flotzinger, P. J. Rösch, and T. Braml. dacl10k: Benchmark for Semantic Bridge Damage Segmentation. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pages 8626–8635, January 2024.
  doi: 10.1109/WACV57701.2024.00843

- J. Flotzinger, P. J. Rösch, C. Benz, M. Ahmad, M. Cankaya, H. Mayer, V. Rodehorst, N. Oswald, and T. Braml. Dacl-challenge: Semantic segmentation during visual bridge inspections. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops, pages 716–725, January 2024.
  doi: 10.1109/WACVW60836.2024.00084

dacl10k: Benchmark for Semantic Bridge Damage Segmentation

Johannes Flotzinger Philipp J. Rösch Thomas Braml
University of the Bundeswehr Munich
{johannes.flotzinger, philipp.roesch, thomas.braml}@unibw.de

Abstract

Reliably identifying reinforced concrete defects (RCDs) plays a crucial role in assessing the structural integrity, traffic safety, and long-term durability of concrete bridges, which represent the most common bridge type worldwide. Nevertheless, available datasets for the recognition of RCDs are small in terms of size and class variety, which questions their usability in real-world scenarios and their role as a benchmark. Our contribution to this problem is "dacl10k", an exceptionally diverse RCD dataset for multi-label semantic segmentation comprising 9,920 images deriving from real-world bridge inspections. dacl10k distinguishes 12 damage classes as well as 6 bridge components that play a key role in the building assessment and recommending actions, such as restoration works, traffic load limitations or bridge closures. In addition, we examine baseline models for dacl10k which are subsequently evaluated. The best model achieves a mean intersection-over-union of 0.42 on the test set. dacl10k, along with our baselines, will be openly accessible to researchers and practitioners, representing the currently biggest dataset regarding number of images and class diversity for semantic segmentation in the bridge inspection domain.

1. Introduction

Bridges are an essential component of the infrastructure worldwide. They are exposed to many impacts causing damage, such as high traffic loads, extreme weather events, sea salt in coastal areas or treatment with deicing chemicals in cold regions. Due to their age, especially bridges in countries with an economic upswing between the 1950s and 1980s show an increased occurrence of damage by now. These defects are monitored within the scope of bridge inspections aiming to assess the condition of buildings in order to find the ideal timing for rehabilitation steps but also to take immediate actions, *e.g.*, limiting heavy traffic or closing a bridge. Such inspections are usually carried out "analogously" by professionally trained civil engineers who visually examine the complete surface of the bridge while taking photos of the defects, and documenting damage class, measurements and location on a 2D sketch [1,3,15]. If these examinations, specifically of buildings in a critical state, could take place more frequently, several bridges may be operated longer and without significant restrictions affecting rail passengers, car drivers and logistics. However, authorities often fail to keep up with the necessary inspection and restoration intervals due to staff shortages and budget limitations, but also because of the commonly applied, time-consuming, analogue inspection process used in practice. In many countries, this leads to a steadily growing stock of built structures in poor condition. In governmental reports, the current state of bridges is described as "1 in 3 U.S. bridges needs repair or replacement ..." [2] or "... at least 25,000 road bridges [in France] are in poor structural condition ..." [5]. This underlines the demand of a more efficient examination pipeline with respect to cost and time [5, 16, 17, 31]. The greatest potential for improvement during the bridge inspection process – irrespective of the used device (unmanned aerial vehicle (UAV) [19], smartphone [45], augmented reality displays [36]) – is yielded by automating the defect recognition which is crucial to a final assessment in order to determine actions to be taken. The inspection framework that makes use of automated defect recognition is called "digitized inspection" (DI). DIs aim for a detailed bridge assessment according to existing country-specific guidelines where the automatized documentation of damage allows reliably classifying, measuring and localizing each existing defect on a given building. Within the scope of DIs, inspectors are strongly supported. The inspections become more efficient and engineers can focus more on the evaluation as well as the context in which defects appear.

The field of damage recognition on built structures is still unexplored. In contrast to the fields of autonomous driving [9,12,18,34,44] or medicine [29,32,35], semantic segmentation benchmarks for damage recognition are rare. To the best of our knowledge, only two relevant benchmarks in the domain of reinforced concrete defects (RCDs) exist: CrackSeg9k [26] and S2DS [7]. CrackSeg9k is a collection of various image datasets showing cracked and uncracked surfaces of multiple building materials. However, it aims to

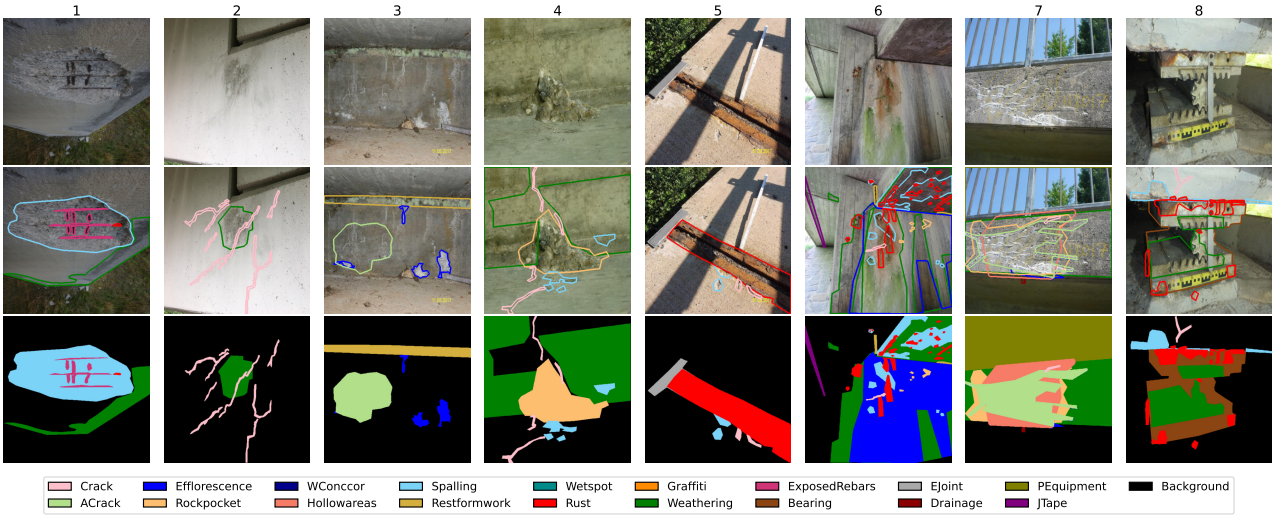


Figure 1. Example annotations from dacl10k. Top row: original image. Middle row: polygonal annotations. Bottom row: stacked masks. The following classes are abbreviated: *Alligator Crack* (ACrack), *Washouts/Concrete corrosion* (WConccor), *Expansion Joint* (EJoint), *Protective Equipment* (PEquipment) and *Joint Tape* (JTape). From left to right, the images display the individual classes: 1. *Weathering, Spalling, Exposed Rebars, Rust*; 2. *Weathering, Crack*; 3. *Alligator Crack, Restformwork, Efflorescence*; 4. *Weathering, Crack, Spalling, Rockpocket*; 5. *Crack, Rust, Expansion Joint, Spalling*; 6. *Weathering, Rockpocket, Spalling, Efflorescence, Crack, Rust, Restformwork, Joint Tape*; 7. *Weathering, Protective Equipment, Rockpocket, Efflorescence, Crack, Hollowareas, Alligator Crack, Drainage*; 8. *Weathering, Spalling, Crack, Rust, Bearing*.

segment one damage type, whereby, at least 9 types must be recognized in order to be used in practice [1, 3, 4, 15]. S2DS is the first multi-class semantic segmentation dataset in RCD domain which includes 743 samples differentiating between five common defects occurring on concrete bridges and control points for georeferencing. Thus, S2DS represents a small variety and complexity with respect to real-world scenarios, with labels assigned to each pixel in a manually exclusive way.

In conclusion, the significant deterioration of bridges worldwide as well as the lack of visual data for effectively monitoring their defects in a digitized manner emphasize the urgent need for establishing a benchmark in the bridge inspection domain.

We take the problem of semantic segmentation of bridge defects out of the niche by introducing *dacl10k*, the biggest real-world inspection dataset for multi-label semantic segmentation making it possible to perform damage classification, measurement and localization on a pixel-level. Thereby, we enable recognizing 12 frequently occurring defects on reinforced concrete bridges (*e.g.*, *Crack*, *Spalling*, *Efflorescence*) and 6 important building parts (*e.g.*, exposed reinforcement bar *Exposed Rebar*, *Bearing*, *Expansion Joint*, *Protective Equipment*). All these classes play an important role for determining the building's structural integrity, traffic safety and durability. dacl10k includes 9,920 images from more than 100 different bridges, specifically designed for practical use, as it comprises all visually unique damage types defined by bridge inspection standards. dacl10k surpasses previous work significantly in terms of its scale, class variety, and the complex nature of its captured scenes. Besides, we provide essential background knowledge from the civil engineering perspective, which is important for a deeper understanding of the RCD domain. In addition, we supply strong baselines to benchmark against. In our model analysis, two semantic segmentation architectures in combination with three encoders are examined. The dacl10k dataset, and according baselines, will be publicly released, fostering research within the field of damage recognition on concrete structures using computer vision.

2. Related datasets and baselines

Within the last six years, major contributions in the field of damage classification on built structures have been made through the introduction of datasets for binary classification [14, 23, 27, 43], multi-class classification [8, 22], multi-label classification [30], object detection [30], and semantic segmentation [6, 7, 26]. In the following, we discuss datasets for the last three named tasks. Examples for the subsequently named damage types can be obtained from Figure 1.

Mundt *et al.* [30] developed CODEBRIM which is cur-

rently the biggest and most realistic dataset for the multi-label classification of RCDs. They differ between the damage types: crack, spallation, exposed reinforcement bar, efflorescence, corrosion and background. The unbalanced version of CODEBRIM comprises 7,729 patches of defect images gathered from 30 bridges, chosen based on varying levels of deterioration, defect size, severity, and surface appearance. The images were acquired under changing weather conditions using multiple cameras at varying scales, with high-resolution. A subset of the data was acquired using an UAV, due to the inaccessibility of defects at high locations. Their annotation process was structured as follows: (*i*) selecting bounding boxes (patches) enclosing defects, (*ii*) iterating over the bounding boxes for each damage class and label accordingly, and finally (*iii*) sampling patches of healthy concrete surfaces as well as irrelevant content (background).

For solving the task of binary crack segmentation, Kulkarni *et al*. [26] combine previously available datasets, inter alia, from the RCD domain. They compile a semantic segmentation dataset, called CrackSeg9k, with 9,255 images of cracks from ten sub datasets on different surfaces. Before unifying the datasets, their individual problems (e.g. noise and distortion) are addressed by applying image processing. In addition, they provide baselines where the best model, based on DeepLabv3 [11], achieves 77% mean intersection-over-union (IoU).

Benz & Rodehorst [7] introduced the Structural Defect Dataset (S2DS) which is the first RCD dataset enabling semantic segmentation of multiple damage types, such as crack, spalling, corrosion, efflorescence, vegetation, and control point which is used for georeferencing. The dataset consists of 743 patches of size 1024x1024 pixel extracted from 8,435 images taken during structural inspections. They used DSLR cameras, mobile phones, and UAVs for acquiring the data. The labeling was executed by one trained computer scientist and had a high level of fineness. Their best model, based on hierarchical multi-scale attention [39], achieves a mean IoU of 92%, at joint scales of 0.25, 0.5, and 1.0.

3. dacl10k dataset

dacl10k is the first large-scale dataset for semantic bridge damage segmentation, comprising 9,920 annotated images from real-world inspections. During its creation, our primary objective was to develop a dataset that enables the training of models which later support the inspector during damage recognition and documentation to a maximum. Hence, we analyzed several guidelines determining the level of detail of structural inspections [1, 3, 4, 15], specifically the visually recognizable defects which must be collected in order to produce a legal bridge assessment. We listed all defects defined by the guidelines accordingly and crossed out the ones that are doppelgangers with respect to visual appearance. Finally, this resulted in the underlying class variety of dacl10k. In the following, we discuss dacl10k's data acquisition, classes, statistics and a comparison to related open-source data.

3.1. Data acquisition

Approximately one half of the images originate from databases of engineering offices, while the other half was provided by local authorities from Germany. The images were taken between 2000 and 2020. Both data sources supplied highly heterogeneous images regarding camera type, pose, lighting condition, and resolution. However, models performing well on dacl10k, most likely generalize well in real-world scenarios.

3.2. Damage types and annotation

The 18 classes considered within dacl10k are separated into three groups: concrete defects, general defects and objects. The class names are shown in the first column of Table 1. The concrete defects appear only on building parts made of (reinforced) concrete, while general defects may be present on all materials (*e.g*., concrete or steel). The only defect within dacl10k that is not visually recognizable, per se, is *Hollowareas*. This damage is usually identified by hammering on the concrete surface, thus, it can only be detected acoustically (not visually) but, as it is bordered with chalk during hands-on inspections, we annotated its markings. The objects group includes all components of a bridge that are not made of concrete, such as *Joint Tapes*, Railings or impact attenuation devices (*Protective Equipment*). The objects often show defects such as geometrical irregularities or deficits in structural capacity. Geometrical irregularities can arise from wrong distances between the railing rods or if the railing height is less than the minimum according to the given national standard. These visually challenging recognizable issues are not part of the dataset. We provide a detailed overview, descriptions and examples of the defect types and objects in the supplementary material.

According to the definition of Cordts *et al*. [12] our dataset comprises coarse pixel-level annotations. We border each defect and object on a given image with one polygon (shape) and assign its label. Furthermore, we include polygons of the same class that overlap with each other in one shape. With respect to inspection standards and application, for all the defect classes, it is not important to differentiate between instances of a given class. Instead, their size and localization on a class-basis is important. Thereby, we utilize the open-source labeling tool LabelMe [33, 40]. Example annotations are shown in Figure 1. It often appears that shapes of different damage or object classes overlap with each other, *e.g*., *Spalling* with *Exposed Rebars* covered by *Rust* (see example 1 in Figure 1). Consequently, the under-

lying task can be described as multi-label semantic segmentation because one pixel can be part of multiple defects and objects. In other words, the labels are not assigned mutually exclusive to the pixels.

The two main components during labeling are the class guideline and the annotation guideline. The class guideline clearly defines the visual appearance, most commonly occurrence and cause of each defect, *e.g.*, *Efflorescences* (see example 7 in Figure 1):

- look like stalactites of white to yellowish or reddish color hanging from the bottom of building parts which may also appear to be printed on the buildings surface,
- often occur in wet (*Wetspot*) or weathered areas (*Weathering*) of the building and in combination with *Crack* and/or *Rust*,
- result from the dissolving of salts from the concrete which consequently carbonate.

Thus, the annotator – independently from his domain expertise – can understand the color, shape and texture of a given class. Furthermore, flags are defined to mark images with personal data (faces, license plates) or images of bad quality. In addition, the provided annotation guideline describes the fineness after which the polygon points shall be set. The goal is to have consistent class and object-distance-dependent density of points over the whole dataset.

Our labeling process consisted of two consecutive steps: Annotation of data received from engineering offices by civil engineering students (in-house) and annotation data from the authorities by an external annotation team, previously filtered for relevant image content. The students labeled approximately 7,000 images in accordance to our guidelines. Nearly 30% of the images had to be rejected due to flags indicating bad quality (blurring, overexposure) or personal data. The pipeline during in-house labeling was separated into three parts. The first part consisted of the regular annotation which comprises annotating a batch of 100 images, getting feedback by a domain expert and correcting the failures accordingly. The second part included an extensive analysis of the dataset to find structural failures in the annotations. Thirdly, the dataset was divided into subtasks with respect to the failures most commonly made. Then, one student corrected each failure type consecutively.

The quality assessment of the data annotated by the external team was divided into four quality checks for each data batch. In average, one batch included 250 images. Each check included one iteration over the annotated data by experts. Based on the analysis, the error rate was determined which is the ratio of false-labeled images and total amount of frames in the according batch. Starting with an error rate of 60%, the rate could be lowered to a final value of 1%.

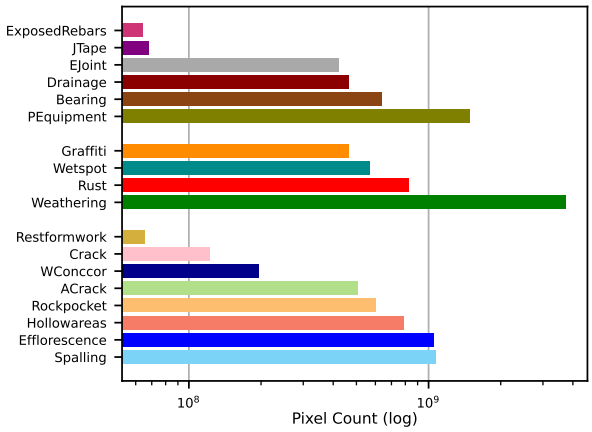


Figure 2. Pixel counts with respect to each class in dacl10k based on the original image sizes. The bars are arranged according to the group affiliation.

3.3. Statistical analysis

In the following, we discuss split-independently (*i*) the classwise pixel counts, (*ii*) averages and shares regarding the polygons and pixels as well as (*iii*) the split-dependent statistics, based on the original image resolutions.

Regarding the pixel counts over the whole dataset (see Figure 2), it can be stated that *Weathering*, followed by *Protective Equipment*, are the most dominant classes with nearly four and 1.5 billion pixels. Within the range of 0.1 billion and 1 billion pixels, the majority of defects and objects with respect to the number of pixels can be found. Clearly underrepresented are the defects *Restformwork* (66 million) and the objects *Joint Tape* (68 million) and *Exposed Rebars* (65 million).

The average image size is 1581 px in height and 1950 px in width. The mean image area is approximately 4 megapixels while the total pixel area in the dataset is approx. 43 billion px. Table 1 provides an overview of statistics describing the classwise density of polygons, size of polygons, density of pixels, share of polygons and share of pixels over the whole dataset. In average, 1.8 crack shapes are present on a crack image, whereby, one polygon includes 27,467 px. The average crack image shows approx. 50,000 px labeled as crack. With respect to the displayed shares, we observe that four out of 100 polygons are labeled as crack and 0.3% of the total pixel area received the label *Crack*. Furthermore, Table 1 reveals the cause of the overrepresented classes *Weathering* and *Protective Equipment*. They display a share regarding the number of polygons of 5.31% (top 20%) and 2.09% (exactly the median). This, in combination with the fact that an according image shows 900,000 px or rather 786,000 px of that class, leads to their dominant role. The overrepresentation of *Weathering* can be

Class	#polyg./ image	#pixels/ polyg.	#pixels/ image	%polyg.	%pixels
Crack	1.81	27,467	49,605	4.02	0.30
ACrack	1.12	950,694	1,064,777	0.48	1.25
Efflorescence	2.29	208,565	478,395	4.54	2.59
Rockpocket	4.75	50,564	240,079	10.74	1.48
WashoutsC.	1.34	807,721	1,079,936	0.22	0.48
Hollowareas	1.21	415,627	504,482	1.72	1.96
Spalling	2.68	83,298	223,185	11.60	2.64
Restformw.	1.19	50,170	59,757	1.18	0.16
Wetspot	1.48	271,408	400,778	1.89	1.40
Rust	3.62	46,680	168,997	16.01	2.04
Graffiti	2.29	172,917	395,596	2.44	1.15
Weathering	1.41	639,776	903,974	5.31	9.28
ExposedR.	2.25	25,770	58,034	2.26	0.16
Bearing	1.45	421,784	612,581	1.37	1.57
EJoint	1.12	700,054	783,023	0.55	1.05
Drainage	1.37	230,815	316,099	1.83	1.15
PEquipment	1.22	641,715	785,601	2.09	3.67
JTape	1.19	49,401	58,569	1.25	0.17
Background	3.47	871,631	3,020,476	30.51	72.69

Table 1. Overall statistics of the dataset regarding average number of polygons per image, number of pixels per polygon, number of pixels per image, share of polygons and share of pixels. Midrules separate the classes according to their group affiliation.

more fatal than the one of *Protective Equipment* with respect to the model performance. This is due to the fact that the features (shape and texture) of *Weathering* are similar to the ones from *Wetspot*. Both are of round shape and represented by a darker area surrounded by a brighter "rest". They vary slightly with regards to their texture. *Weathering* is more noisy and more matt than *Wetspot* which is smooth and sometimes mirroring. In addition, *Wetspot* and *Weathering* often overlap which makes it difficult to distinguish between them from the model's perspective. The features of the object *Protective Equipment*, in contrast, are unique and therefore shouldn't interfere with other classes during learning. The lack of pixels representing *Restformwork*, *Exposed Rebars* and *Joint Tape*, as mentioned before, originates from their relatively rare occurrences but mostly from their small shapes. In average, polygons bordering *Restformwork* or *Joint Tape* have a size of approx. 50,000 px while polygons labeled as *Exposed Rebars* include 26,000 px. The average size of their polygons is equal to or less than the lower quartile (50,367 px). Additionally, *Exposed Rebars* shows the smallest average polygon size of all classes.

To ensure similar data distributions (regarding damage and object classes) in each split, we create data partitions according to image similarity. For that, we employ K-means clustering [28] with 20 clusters. Then, we proportionally draw samples from each cluster for the train (70%), valid (10%), testdev (10%), and testpriv (10%) set with respect to the number of polygons and images (see Table 2). The testdev and testpriv split are summarized within the test split.

The Table presents the classwise statistics of dacl10k separated into the given splits. Again, taking the *Crack* class as an example, the aforementioned proportions can be observed in terms of number of pixels with a number of 89,316,599 px (73%) in the train and approx. 11,000 px (9%) in the validation split. Compared to the validation split, the values of the test split are approx. twice as high. For the number of polygons and images, Table 2 displays the same targeted proportions as for the pixel count.

3.4. Comparison to other datasets

Compared to CrackSeg9k [26] and S2DS [7], the crack annotations of dacl10k are coarser. CrackSeg9k is a collection of multiple available binary crack segmentation datasets where each was acquired in a standardized setting with respect to camera pose, lighting condition and hardware. S2DS is a single-label semantic segmentation dataset that includes images of RCDs (and control points) captured during real structural inspections, where the fineness of crack annotations is also high. According to the inspection guidelines, the recognition of cracks requires the highest degree of accuracy because their width plays an important role during their assessment. *E.g.*, in Germany the minimum crack width, which must be documented, is 0.2mm [4, 15]. Consequently, for the damage type *Crack*, finer annotations are definitely useful when it comes to practical use.

To sum up, both related datasets show a higher level of detail regarding crack annotations than dacl10k but are less diverse with respect to class variety and real-world scenarios. CrackSeg9k enables the training of models for binary crack segmentation only. The annotations of S2DS provide single-target information at most, meaning that one pixel can only belong to one class. In addition, S2DS consists of a relatively small number of images. However, we focus on the multi-label semantic segmentation of all visually unique defects and objects on concrete bridges. Thereby, we take into account the frequently occurring case that in real-world scenarios multiple defects overlap. Regarding the crack annotations, it can be stated that a pixel-accurate classification of our crack data may be possible by applying methods from the field of weakly-labeled data [21]. In the CityScapes dataset [12], for example, the majority of the samples were coarsely annotated (20,000 images) with the intention to foster research specifically in this field. With regards to applications within the framework of existing standards, all other classes in our dataset do not require a finer annotation and prediction, respectively. *E.g.*, during currently practiced analogue inspections, the diameter and corresponding area of a *Spalling* is usually measured in a rough manner with a folding rule which is sufficient for its assessment.

Class	Train			Valid			Test		
	#pixels	#polyg.	#images	#pixels	#polyg.	#images	#pixels	#polyg.	#images
Crack	89,316,599	3,092	1,720	11,462,336	457	254	21,199,921	892	485
ACrack	379,337,903	378	336	33,145,894	48	42	93,285,498	106	97
Efflorescence	773,838,619	3,378	1,523	69,295,099	502	206	204,072,743	1,141	460
Rockpocket	416,831,564	8,241	1,712	51,701,480	1,207	259	131,663,248	2,422	529
WConccor	153,351,739	176	133	7,781,177	23	15	34,335,547	43	33
Hollowareas	589,108,842	1,327	1,100	68,277,162	193	155	133,137,140	382	312
Spalling	754,421,298	8,638	3,289	94,740,019	1,444	485	218,557,811	2,736	1,010
Restformwork	42,413,460	841	716	6,143,333	164	132	17,116,171	304	251
Wetspot	402,563,879	1,436	972	47,002,103	216	144	117,133,455	436	298
Rust	592,984,180	12,272	3,451	79,135,671	1,801	465	153,938,160	3,623	972
Graffiti	308,779,763	1,866	797	71,491,732	317	146	85,741,013	512	235
Weathering	2,563,181,494	4,056	2,830	367,950,798	572	407	823,072,419	1,240	916
ExposedRebars	44,829,810	1,720	773	3,824,934	244	104	15,821,469	538	234
Bearing	431,016,513	1,039	731	89,236,423	160	105	116,219,161	310	203
EJoint	335,469,869	446	396	21,145,699	56	51	66,216,825	102	93
Drainage	368,660,666	1,393	1,030	35,297,153	244	151	62,288,022	383	294
PEquipment	1,070,432,844	1,616	1,320	103,082,479	211	175	312,055,649	488	396
JTape	47,582,309	911	772	8,569,024	152	128	12,022,699	317	264
Background	21,196,542,072	23,347	6,801	2,681,148,040	3,279	962	5,514,566,563	7,095	1,968

Table 2. Splitwise statistics regarding the number of pixels, polygons and images.

4. Baselines

In the following, we describe the development of the baseline models and their test results. In order to evaluate the baselines and to demonstrate challenges with respect to the annotation of the underlying data, we conduct an "Engineer versus Machine" (EvsM) comparison. Finally, we discuss the results incorporating the findings regarding dacl10k, the baselines and the EvsM comparison.

4.1. Implementation details

For the development of the baselines we analyze two different CNN-based semantic segmentation architectures with three different encoders, and one Transformer-based model. We used DeepLabV3+ [10,11] and Feature Pyramid Network (FPN) [25] as CNN baseline architectures which represent powerful models often used in research and industry. As encoders we utilize MobileNetV3-Large [20] (3M parameters), EfficientNet-B2 [38] (7M parameters) and EfficientNet-B4 (17M parameters). For each of these base models we also investigate a separate model using an auxiliary loss for multi-label classification. The auxiliary classification head placed after the encoder consists of an average global pooling layer, followed by a dropout and linear layer. The weights of the model are updated based on a combined weighted loss including the mask and auxiliary loss. The total loss $\mathcal{L}_{total}$ is computed as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{mask} + 0.1 \times \mathcal{L}_{aux} \quad (1)$$

where $\mathcal{L}_{mask}$ and $\mathcal{L}_{aux}$ are based on Dice [37] and Cross Entropy loss respectively.

To meet current developments in Transformer-based models, we trained a SegFormer model [42]. Since auxiliary loss is unusual for this network, only Dice loss is utilized. Adam optimizer [24] with four different learning rates ($5e^{-3}$, $1e^{-3}$, $5e^{-4}$, $1e^{-4}$) is applied whereof the model with the best loss based on the validation split is reported. We make use of a cosine learning rate scheduler with a warm-up phase over two epochs and train each model for 30 epochs. All models are initialized with ImageNet weights [13,41].

The images and annotations are resized to a resolution of 512×512 before being fed to the network. Thereby, each defect and object class is considered with a separate binary mask, leading to a total number of 18 channels. All following results are reported on the same resolution, enabling an evaluation that is not focused towards large images.

4.2. Baseline results

In order to find the best performing model on dacl10k, we compare the mean IoU of seven different models (see Table 3). The best network using the DeepLabV3+ architecture includes an EfficientNet-B4 backbone without considering auxiliary loss. It achieves a mean IoU of 0.411. The highest mean IoU is obtained at a value of 0.414 by the model consisting of an EfficientNet-B4 encoder and FPN architecture while taking the auxiliary loss into account. For both, FPN and DeepLabV3+, it can be observed that the more parameters the encoder has, the higher is the

Aux	DeepLabv3+			FPN			SegFor
	MN	EN-B2	EN-B4	MN	EN-B2	EN-B4	
–	0.320	0.360	0.411	0.376	0.384	0.364	0.400
✓	0.329	0.400	0.409	0.378	0.395	**0.414**	–

Table 3. Mean IoU on valid split for both architectures and three encoders (MobileNetV3-Large, EfficientNet-B2 and EfficientNet-B4) with and without auxiliary loss, and the SegFormer model.

reached mean IoU. The SegFormer model achieves a mean IoU of 0.400 which is less than the best model based on DeepLabV3+ or FPN.

Table 4 displays the classwise IoUs and the mean IoU of the best model, described in the preceded paragraph. We report results on the validation and test split (see Table 2). Due to the small differences between the metrics over the splits, it can be stated that the data, with respect to complexity, is evenly distributed. In the following, the results on the test split are documented. The lowest IoU is obtained for the defect *Washouts/Concrete Corrosion* with a value of 0.121. This class is not underrepresented (see Figure 2). We observe that its texture and shape (features) are very familiar to the ones from *Rockpocket* and *Spalling* which both are strongly represented. Other classes for which a low IoU is reported are *Wetspot*, *Restformwork*, *Crack* and *Rockpocket*. The possible reason for the bad performance on *Wetspot* is explained in Section 3.3. *Restformwork* has many different visual appearances, which the model probably fails to summarize within one class. The *Crack* class, in average, is represented by the least number of pixels per image, and it has the smallest polygons, after *Exposed Rebars*, as they are elongated and very narrow (see Table 1). Thus, the IoU is a challenging metric for this defect, as the false-negative segmentation of classes represented by more pixel area is less penalized. On *Rockpocket* a low IoU is obtained because of the aforementioned similar-looking defects. The best results can be reported for the objects *Protective Equipment* (0.715) and *Bearing* (0.564) while the best general defect is *Graffiti* (0.623) and with respct to concrete defects, a good IoU can be observed for *Hollowareas* (0.555) and *Alligator Crack* (0.482). *Graffiti* and *Protective Equipment* are the two most individual classes regarding their visual appearance, additionally, they are well represented (see Figure 2) which explains their high IoU compared to the rest of the classes. Summarizing, the best model achieves a mean IoU of 0.424. A more detailed analysis of the problematic classes is provided within the supplementary material.

Class	valid	test
Crack	0.288	0.286
ACrack	0.473	0.482
Efflorescence	0.338	0.415
Rockpocket	0.267	0.294
WConccor	0.085	0.121
Hollowareas	0.536	0.555
Spalling	0.374	0.406
Restformwork	0.336	0.285
Wetspot	0.232	0.243
Rust	0.414	0.450
Graffiti	0.586	0.623
Weathering	0.423	0.395
ExposedRebars	0.393	0.358
Bearing	0.676	0.564
EJoint	0.474	0.524
Drainage	0.521	0.563
PEquipment	0.675	0.715
JTape	0.362	0.362
Mean	0.414	0.424

Table 4. Classwise and mean IoU of the best model (FPN with EfficientNet-B4 and auxiliary loss) on the validation and test split.

5. Engineer vs. Machine

Figure 3 displays the EvsM comparison, which enables a qualitative validation of segmentations from the best model (see Section 4.2) by comparing the annotations of civil engineers with the network's predictions. Therefore, we asked the five experts, two of whom perform bridge inspections frequently, to annotate four representative samples drawn from dacl10k's testdev split. Two samples are considered easy and the others difficult to evaluate because of their low resolution, diversity regarding the lighting condition, camera pose as well as defect types and objects. After an introduction to the class guideline, the experts were instructed to annotate the images with the same quality they would expect an application to highlight defects during a close-up or hands-on bridge inspection. Compared to other inspection types, hands-on inspections require the highest quality with respect to defect classification, measurement and localization, which also includes the detection of *Hollowareas* by hammering the concrete surface. Regarding the differences among engineers, it can be stated that with increasing complexity of the samples also the variance of the chosen labels rises. Engineer 5 annotated at the highest quality level. Thus, especially his annotation is used as a qualitative benchmark for our baselines prediction. For the first two samples, which contain fewer classes compared to the last two, the prediction of the machine is better than the average annotation of the five engineers. Only in the second image the small cavities that are part of the defect *Rockpocket* and the *Protective Equipment*, which is overexposed, are not recognized. The last two images reveal the limits of our baseline. For the third sample the classes *Wetspot*, *Rust*

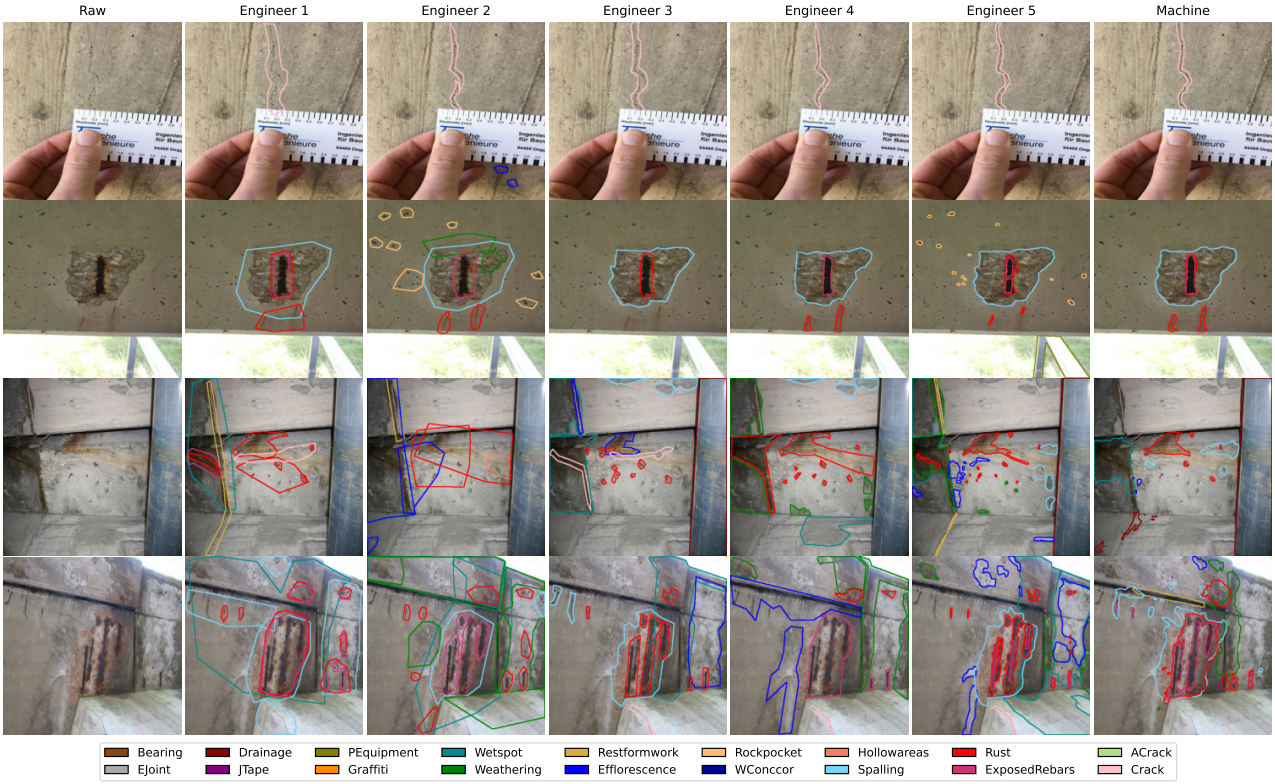


Figure 3. Qualitative evaluation of the best model on four samples (Raw) from the testdev split by comparing annotations of five civil engineers (Engineer 1-5) with the best model's predictions (Machine). From top to bottom, the samples show: *Crack*; a few tiny cavities which are considered as *Rockpockets* and *Spalling, Exposed Rebars, Rust*; *Wetspot, Weathering, Efflorescence, Crack, Spalling, Exposed Rebars and Rust*; *Wetspot, Weathering, Efflorescence, Restformwork, Drainage, Rust, Spalling*.

and *Drainage* are well predicted, whereas, areas of *Efflorescence* and weak *Spallings* are not segmented. *Weathering* is not present at all in the model's prediction. On the last image the model doesn't predict the defects *Crack* at the center bottom and the *Wetspot* which is located mainly on the wall. Furthermore, the model classifies *Restformwork* which is not present on the image, while *Spalling* and *Exposed Rebars* are correctly recognized. The remaining classes are partially classified.

6. Discussion

We have introduced the first large-scale dataset and corresponding baselines for multi-label semantic segmentation in the bridge inspection domain. It's based on real-world data, with a label distribution deriving from the visually distinguishable classes of multiple country-specific inspection standards. It is important to note that the concrete and general defect group labels are not restricted to bridges, as they can occur on any building made of (reinforced) concrete. The evaluation of the baselines, especially the EvsM comparison, generally shows a good performance. However, limitations can be observed for classes with minimal feature differences between each other, *e.g.*, *Weathering* and *Wetspot* or *Washouts/Concrete Corrosion*, *Rockpocket* and *Spalling*. In addition, the datatset is highly unbalanced, leading to biases towards the overrepresented classes.

We are confident that a more sophisticated search for architectures and hyperparameters as well as data augmentation methods will lead to substantial improvements. Additionally, with respect to finer *Crack* segmentations, approaches from the field of weakly-labeled data may better satisfy the geometrical accuracy requirements of this defect type. Overall, we believe that due to its size and diversity, dacl10k makes an important contribution to the field of automated structural inspection.

Acknowledgements. We thank Wolfgang Kusterle, Helmut Mayer and Johannes Kreutz for their insights and suggestions. We are most grateful to the engineering offices and authorities for providing the inspection images. This project has received funding from the Bavarian Ministry of Economic Affairs (MoBaP research project, IUK 639/003).

References

[1] AASHTO. *Manual for Bridge Evaluation, 3rd Edition*. American Association of State Highway and Transportation Officials, Washington, D.C., 2018. 1, 2, 3

[2] American Road & Transportation Builders Association (ARTBA). ARTBA bridge report, 2023. 1

[3] Austrian Research Association for Roads, Railways and Transport (RVS). Quality assurance for structural maintenance; Surveillance, checking and assessment of bridges and tunnels; Road bridges. Standard RVS 13.03.11, Vienna, AUT, 2011. 1, 2, 3

[4] Federal Highway Research Institute (BaSt). Guideline for the uniform acquisition, assessment, recording and evaluation of results of structural inspections (RI-EBW-PRÜF), Feb. 2017. 2, 3, 5

[5] Bruno M. Belin. Rapport d'information. *Session Ordinaire*, 669, 2022. 1

[6] Christian Benz, Paul Debus, Huy Khanh Ha, and Volker Rodehorst. Crack segmentation on uas-based imagery using transfer learning. In *2019 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6, 2019. 2

[7] Christian Benz and Volker Rodehorst. Image-based detection of structural defects using hierarchical multi-scale attention. In *DAGM German Conference on Pattern Recognition (GCPR)*, pages 337–353. Springer, 2022. 1, 2, 3, 5

[8] Eric Bianchi and Matt Hebdon. Bearing Condition State Classification Dataset. University Libraries, Virginia Tech, Oct. 2021. 2

[9] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97, 2009. Video-based Object and Event Analysis. 1

[10] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017. 6

[11] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018. 3, 6

[12] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 3, 5

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 6

[14] Sattar Dorafshan, Robert J. Thomas, and Marc Maguire. SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks. *Data in Brief*, 21:1664–1668, Dec. 2018. 2

[15] German Institute for Standardization Registered Association (DIN). Engineering structures in connection with roads - inspection and test. Standard DIN 1076:1999-11, Berlin, GER, 1999. 1, 2, 3, 5

[16] RAC Foundation. Substandard road bridges in Great Britain 2021, 2022. 1

[17] BaSt Bundesanstalt für Strassenwesen. Brückenstatistik, 2023. 1

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 1

[19] Gi-Hun Gwon, Jin Hwan Lee, In-Ho Kim, and Hyung-Jo Jung. Cnn-based image quality classification considering quality degradation in bridge inspection using an unmanned aerial vehicle. *IEEE Access*, 11:22096–22113, 2023. 1

[20] A. Howard, M. Sandler, B. Chen, W. Wang, L. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le. Searching for mobilenetv3. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, Los Alamitos, CA, USA, nov 2019. IEEE Computer Society. 6

[21] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to segment every thing. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4233–4241, 2018. 5

[22] Philipp Hüthwohl, Ruodan Lu, and Ioannis Brilakis. Multi-classifier for reinforced concrete bridge defects. *Automation in Construction*, 105, Sep 2019. 2

[23] Philipp Hüthwohl and Ioannis Brilakis. Detecting healthy concrete surfaces. *Adv. Eng. Inf.*, 37:150–162, Aug. 2018. 2

[24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. 6

[25] A. Kirillov, R. Girshick, K. He, and P. Dollar. Panoptic feature pyramid networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6392–6401, Los Alamitos, CA, USA, jun 2019. IEEE Computer Society. 6

[26] Shreyas Kulkarni, Shreyas Singh, Dhananjay Balakrishnan, Siddharth Sharma, Saipraneeth Devunuri, and Sai Chowdeswara Rao Korlapati. Crackseg9k: A collection and benchmark for crack segmentation datasets and frameworks. In Leonid Karlinsky, Tomer Michaeli, and Ko Nishino, editors, *Computer Vision – ECCV 2022 Workshops*, pages 179–195, Cham, 2023. Springer Nature Switzerland. 1, 2, 3, 5

[27] Shengyuan Li and Xuefeng Zhao. Image-Based Concrete Crack Detection Using Convolutional Neural Network and Exhaustive Search Technique. *Advances in Civil Engineering*, 2019(Ml), 2019. 2

[28] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. 5

[29] Bjoern H. Menze, Andras Jakab, Stefan Bauer, Jayashree Kalpathy-Cramer, Keyvan Farahani, Justin Kirby, Yuliya Burren, Nicole Porz, Johannes Slotboom, Roland Wiest, Levente Lanczi, Elizabeth Gerstner, Marc-André Weber, Tal Arbel, Brian B. Avants, Nicholas Ayache, Patricia Buendia, D. Louis Collins, Nicolas Cordier, Jason J. Corso, Antonio Criminisi, Tilak Das, Hervé Delingette, Çağatay Demiralp,

Christopher R. Durst, Michel Dojat, Senan Doyle, Joana Festa, Florence Forbes, Ezequiel Geremia, Ben Glocker, Polina Golland, Xiaotao Guo, Andac Hamamci, Khan M. Iftekharuddin, Raj Jena, Nigel M. John, Ender Konukoglu, Danial Lashkari, José António Mariz, Raphael Meier, Sérgio Pereira, Doina Precup, Stephen J. Price, Tammy Riklin Raviv, Syed M. S. Reza, Michael Ryan, Duygu Sarikaya, Lawrence Schwartz, Hoo-Chang Shin, Jamie Shotton, Carlos A. Silva, Nuno Sousa, Nagesh K. Subbanna, Gabor Szekely, Thomas J. Taylor, Owen M. Thomas, Nicholas J. Tustison, Gozde Unal, Flor Vasseur, Max Wintermark, Dong Hye Ye, Liang Zhao, Binsheng Zhao, Darko Zikic, Marcel Prastawa, Mauricio Reyes, and Koen Van Leemput. The multimodal brain tumor image segmentation benchmark (brats). *IEEE Transactions on Medical Imaging*, 34(10):1993–2024, 2015. 1

[30] Martin Mundt, Sagnik Majumder, Sreenivas Murali, Panagiotis Panetsos, and Visvanathan Ramesh. Meta-learning convolutional neural architectures for multi-target concrete defect classification with the concrete defect bridge image dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[31] American Society of Civil Engineers (ASCE). 2021 Infrastructure Report Card, 2021. 1

[32] Patrik F. Raudaschl, Paolo Zaffino, Gregory C. Sharp, Maria Francesca Spadea, Antong Chen, Benoit M. Dawant, Thomas Albrecht, Tobias Gass, Christoph Langguth, Marcel Lüthi, Florian Jung, Oliver Knapp, Stefan Wesarg, Richard Mannion-Haworth, Mike Bowes, Annaliese Ashman, Gwenael Guillard, Alan Brett, Graham Vincent, Mauricio Orbes-Arteaga, David Cárdenas-Peña, German Castellanos-Dominguez, Nava Aghdasi, Yangming Li, Angelique Berens, Kris Moe, Blake Hannaford, Rainer Schubert, and Karl D. Fritscher. Evaluation of segmentation methods on head and neck ct: Auto-segmentation challenge 2015. *Medical Physics*, 44(5):2020–2036, 2017. 1

[33] Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. LabelMe: A database and web-based tool for image annotation. *International Journal of Computer Vision*, 77(1-3):157–173, Oct. 2007. 3

[34] Timo Scharwächter, Markus Enzweiler, Uwe Franke, and Stefan Roth. Efficient multi-cue scene segmentation. In Joachim Weickert, Matthias Hein, and Bernt Schiele, editors, *Pattern Recognition*, pages 435–445, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. 1

[35] Amber L. Simpson, Michela Antonelli, Spyridon Bakas, Michel Bilello, Keyvan Farahani, Bram van Ginneken, Annette Kopp-Schneider, Bennett A. Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M. Summers, Patrick Bilic, Patrick F. Christ, Richard K. G. Do, Marc Gollub, Jennifer Golia-Pernicka, Stephan H. Heckers, William R. Jarnagin, Maureen K. McHugo, Sandy Napel, Eugene Vorontsov, Lena Maier-Hein, and M. Jorge Cardoso. A large annotated medical image dataset for the development and evaluation of segmentation algorithms, 2019. 1

[36] Alan Smith, Charlie Duff, Rodrigo Sarlo, and Joseph L. Gabbard. Wearable augmented reality interface design for bridge inspection. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pages 497–501, 2022. 1

[37] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 240–248. Springer International Publishing, 2017. 6

[38] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. 6

[39] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation, 2020. 3

[40] Kentaro Wada. labelme: Image polygonal annotation with python. https://github.com/wkentaro/labelme, 2018. 3

[41] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019. 6

[42] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers, 2021. 6

[43] Hongyan Xu, Xiu Su, Huaiyuan Xu, and Haotian Li. Autonomous bridge crack detection using deep convolutional neural networks. In *3rd International Conference on Computer Engineering, Information Science & Application Technology (ICCIA 2019)*, pages 274–284. Atlantis Press, 2019. 2

[44] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2633–2642, 2020. 1

[45] Mahta Zakaria, Enes Karaaslan, and F. Necati Catbas. Advanced bridge visual inspection using real-time machine learning in edge devices. *Advances in Bridge Engineering*, 3(1), Dec. 2022. 1

dacl10k: Benchmark for Semantic Bridge Damage Segmentation – Supplementary Material –

Johannes Flotzinger Philipp J. Rösch Thomas Braml
University of the Bundeswehr Munich
{johannes.flotzinger, philipp.roesch, thomas.braml}@unibw.de

A. dacl10k

A.1. Additional statistics

Table 1 displays additional statistics of dacl10k. The average image has a picture format of 4:3 and comprises four megapixels. In total, dacl10k includes 40 billion pixels and 110,533 polygons. The average number of polygons per image amounts to eleven.

#images	9,920
Image width (min, mean, max)	336, 1950, 6000
Image height (min, mean, max)	245, 1581, 5152
#pixels of image areas	40,435,268,789
Average #pixels/image	4,076,136
Average #polygons/image	11

Table 1. Additional statistics on dacl10k.

A.2. Accessibility

The dacl10k dataset is made freely available to academic and non-academic entities for non-commercial purposes such as academic research, teaching, scientific publications, or personal experimentation. Permission is granted to use the data, given that you agree to the Attribution-NonCommercial 4.0 International (CC BY-NC 4.0) license.

A.3. Class descriptions

In Table 3 (Concrete Defects), Table 4 (General Defects) and Table 5 (Objects) a detailed description and an example image for each class are listed. Within these tables, we describe the visual appearance and the cause of the defect, or rather the functionality of the given object. The examples include the annotations of the according class exclusively. Additionally, the class abbreviations – used in the tables and figures of this work as well as in the annotation files of dacl10k – are given in parentheses.

For a deeper understanding of the concrete defects it is important to note that ordinary concrete consists of cement, water, sand and coarse aggregate (gravel). In its unhardened state (wet concrete) the mix of cement and water is named cement paste, while the hardened paste is called cement stone. The cement paste – later cement stone – binds the other concrete components, sand and gravel, together. Regarding the concrete defects *Crack*, *Alligator Crack* and *Spalling* it is noteworthy that concrete has a high compressive strength but a low tensile strength ($\approx$10% of its compressive strength). The overstressing of its tensile strength leads to the occurring of the according defect. The cause of the overstressing differs for each of these defects, which is further explained in Table 3.

A.4. Annotation-specific problems

Only a few related datasets are available (see main paper), which are limited to five RCDs. In addition, the labeling of dacl10k requires deep domain knowledge. Furthermore, dacl10k was labeled by two different groups of annotators, civil engineering students and a professional annotation team. On the one hand, civil engineers have in-depth knowledge about RCDs and bridge components. On the other hand, they are usually not familiar with computer vision problems and, therefore, are not aware of the caveats during the development of semantic segmentation datasets. The professional annotation team, against it, has no domain expertise in regards to bridge defects and objects. Thus, their understanding of important context is less pronounced.

A.5. Data-specific problems

In the following, we discuss the most challenging defects (see Figure 1) and general problems across the dataset (see Figure 2). The listed examples are problematic with respect to annotation and their prediction by our baselines – due to the low IoU reported in the main paper.

The images in Figure 1a display *Cracks* that are bordered by strongly varying background. The most left image shows an abrupt change regarding the background at the vertical edge of an abutment wall, where one side represents a clean concrete surface and the other is heavily weathered (*Weathering*). On the middle image, the *Crack* traverses different

colors of a *Graffiti*. The most right sample displays vegetation which covers the underlying *Crack*. The two most left images in Figure 1b display defects that can be easily mixed up. *Alligator Cracks* are areas of multiple *Cracks* that are branched and arbitrary orientated, which makes their differentiation complicated. In the most left image, both of these classes are present. As mentioned in the main paper, differentiating between *Weathering* and *Wetspot* is problematic because they often overlap with each other (see middle image in Figure 1b). Another problematic class is *Restformwork* which is under the lower five classes regarding the achieved IoU by our best model. In the most right image of Figure 1b *Restformwork* is covered by concrete slurry. Furthermore, *Restformwork* can be made of wood and the color of the polystyrene may vary. Thus, this class has many different visual appearances, which the model has to learn to summarize in one damage class. In Figure 1c, we display the three classes of dacl10k which are the most complicated to differ: *Spalling*, *Rockpocket* and *Washouts/Concrete corrosion*. While they look very similar, their cause and assessment differ decisively (see Table 3).

Figure 2 includes images representing class-independent difficulties (not exclusively for one class) and image quality problems across the dataset. Often, defects show no clear border, leading to inconsistent annotations and predictions. This "smearing of damage borders" can be observed for *Efflorescence*, *Weathering*, *Wetspot* and *Rust*. An example of the latter is shown in the most left image tile of Figure 2a. In addition, objects and defects can reach from the foreground to the background (see middle image in Figure 2a), and some defects may show weak differences to their surrounding area due to low contrast (see most right image in Figure 2a). Further problems concern the image quality, such as lens flares, partial overexposure due to the usage of flashlight or reflections (see Figure 2b).

B. Baseline experiments

B.1. Additional training informations

For each of the six combinations of encoder and semantic segmentation architecture, we examined a grid search for the learning rates $1e^{-4}$, $5e^{-4}$, $1e^{-3}$ and $5e^{-3}$ where the best value is chosen based on the validation loss.

B.2. Auxiliary multi-label results

In Table 2 we display additional metrics from the auxiliary head of the best model on the test split.

Class	Precision	Recall	F1 Score	#images
Crack	0.77	0.52	0.62	485
ACrack	0.78	0.61	0.68	97
Efflorescence	0.82	0.60	0.69	460
Rockpocket	0.68	0.55	0.60	529
WConccor	0.42	0.24	0.31	33
Hollowareas	0.84	0.77	0.81	312
Spalling	0.80	0.76	0.78	1010
Restformwork	0.69	0.51	0.59	251
Wetspot	0.59	0.42	0.49	298
Rust	0.88	0.72	0.79	972
Graffiti	0.87	0.67	0.76	235
Weathering	0.75	0.68	0.72	916
ExposedRebars	0.80	0.59	0.68	234
Bearing	0.85	0.82	0.83	203
EJoint	0.74	0.67	0.70	93
Drainage	0.88	0.54	0.67	294
PEquipment	0.88	0.81	0.84	396
JTape	0.75	0.58	0.66	264

Table 2. Test results for auxiliary head of the best model.

C. Bridge inspection

In the following, we describe the currently practiced bridge inspection process (analogue inspection), its limitations and the concept of the digitized inspection including the automated damage recognition enabled by multi-label semantic segmentation models. Thereby, we use the process of hands-on inspections in Germany as an example. It is important to note that inspection processes worldwide, and especially the defect documentation, are similar.

C.1. Analogue inspection

Currently practiced bridge inspections are carried out by a professionally trained civil engineer (bridge inspector). Usually, the inspector observes the complete surface of the given building while capturing each defect. During hands-on inspections, it is mandatory to detect visually recognizable defects and *Hollowareas*, which can be detected by hammering the concrete surface. Thus, *Hollowareas* are recognized based on the sound the hammering provokes[1]. After the detection of a defect the inspector takes an image and notes, the damage class, size and location (damage-information) in a handwritten damage sketch. In Figure 3 such a sketch is displayed. In this case, the inspector numbered the defects chronologically and named the corresponding class and size (if necessary) of each damage, *e.g.*, #32: Cracks ("Risse") with a thickness of "$\leq 0.2mm$", approximately 7m from the cross girder ("Querträger"), between the left side of the web and flange of the most left T-beam (cross-section 4-5). After the inspection was completed, some inspectors additionally transfer the handwritten sketch to a CAD sketch.

Finally, the assessment of the building is determined by calculating the condition grade, which is similar to grades in school, and preparing the inspection report. According to the damage-information noted in the damage sketch, the

[1] *Hollowareas* and *Cracks* are marked with chalk markings to make them easy to find in successive inspections.

defect ID, which is defined by the German standard, can be assigned. For each defect and its ID, the country-specific standard recommends grades with respect to structural integrity, traffic safety and durability. Based on the expertise of the inspector, these recommendations are adapted. In the inspection report all defects are listed including their grades and recommendations regarding restoration, traffic load limitations or building a new construction. Furthermore, the defect sketch is attached to the report in order to enable a visual comparison with other – especially consecutive – inspections. This is important for tracking the defect development.

C.2. Limitations of the analogue inspection

The process components of the analogue bridge inspection, such as classifying, measuring, locating and assessing the defects, is often inconsistent, error-prone, and lengthy. Oftentimes, this is due to the previously described cumbersome damage documentation.

According to Phares *et al*. [1] the assessment results of bridge inspections vary greatly between inspectors. Thereby, Inspection reports from 49 bridge inspectors from 25 different state departments of transportation (DOTs) at seven structures, each in the United States, were evaluated. It was found that approximately 56% of the condition ratings deviated significantly from the reference condition rating (ground truth). The main reasons for the strongly diverging ratings is the variability in inspection documentation (e.g., field inspection notes, photographs, etc.). Additionally, inspection notes concerning important structural defects or corresponding photographs were often omitted.

We think, in addition, the following components of the analogue inspection contribute to the widely varying inspection results:

- The correct image must be found for the corresponding defect in the defect sketch after the inspection. This is problematic, especially on bridges with many defects that are close together.
- Damage size is measured with a pocket rule, thus, it's imprecise.
- Damage localization is often estimated or determined by measuring the distance to the next bridge pillar, which is inaccurate (or impossible) on bridges with a big span.

C.3. Digitized inspection

The basis for a digital building inspection (DI) is a Building Information Model (BIM), which is created during the building's planning phase or, in the case of existing bridges, before the inspection. At the structure the inspector records all defects making use of an UAV or smartphone. For non-hands-on inspections, UAVs can be used to record the defects because only visually recognizable defects have to be documented during this type of inspection. For hands-on (or close-up) inspections, smartphones or tablets are the most suitable devices to support the inspector, as they are handy and easy to use. In addition, close-up inspections require the detection of *Hollowareas* which are detected by hammering the concrete surface, thus, making use of UAVs is not possible. Independent of the device, the automated damage detection is the central part of the inspection process, provided by a multi-label semantic segmentation model. The model enables the classification, measurement and localization on a pixel-level in order to assign an ID to each defect and to obtain the grading recommendations. These properties of each detected defect are stored as metadata for each DI. As a result, the BIM can be merged with the metadata of any performed inspection, allowing the generation of a digital twin. This allows for easily tracking the development of the bridge's condition over time. Consequently, instead of comparing damage sketches and inspection reports that are available in paper form, the evolution of the digital twin are assessed.

The described DI has the goal to assist the inspector as much as possible in performing inspections and the consecutive evaluation within the framework of existing standards. The DI is simpler and more efficient than the analogue inspection, while the inspector remains the central decision-maker (human oversight). The acceleration of the inspection mainly results from the automated damage detection and the elimination of damage documentation on the defect sketch. For each detected damage, the DI-application provides a suggestion for the properties, or rather metadata. This recommendation can be corrected and saved by the inspector on site. This results in a direct quality management, which is particularly important for defects that require visual contextual knowledge.

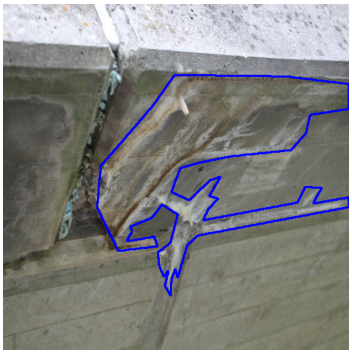
Damage	Visual appearance	Cause	Example
Crack	• Elongated and narrow zigzag line • Clearly darker compared to the surrounding area or black	• Concrete's tensile strength is exceeded • Too high bending or shearing load • Settlement of the substructure	
Alligator Crack (ACrack)	• Many branched cracks • Mostly arbitrarily orientated • Usually with a small crack width (compared to *Crack*)	• Inadequate post-treatment or concrete recipe (shrinkage) • Too high temperatures during hardening of the concrete • Formation of expansive phases leading to a volume increase in the concrete as a result of chemical action	
Efflorescence	• Mostly roundish areas of white to yellowish or reddish color • Strong efflorescence can look similar to stalactites. • Often appears in weathered (*Weathering*) or wet areas (*WetSpot*) of the building and in combination with *Crack* and/or *Rust*	• Dissolving of salts (calcium, sodium, potassium) from the cement stone or aggregate by humidity changes or water ingress, *e.g.* constantly running water through the building part or along its surface • The salts consequently carbonate leading to the final visual appearance. • Note: Water ingress can be caused by other defects imposing the draining of water, *e.g.*, *Restformwork* or damaged *Drainage*. *Efflorescences* are also called Calcium leaching.	

Table 3. Concrete defects, their visual appearance, cause and example images with according annotation.

Damage	Visual appearance	Cause	Example
Rockpocket	• Visible coarse aggregate • Often in tilts of the formwork and the bottom of building parts (opposite side from which the concrete is poured into the formwork)	• Inadequate rheological properties (viscosity, yield point) of the concrete • Appears due to bad compacting after having poured the concrete into the formwork. Therefore, insufficient deaeration of the concrete follows, causing areas where the cement paste didn't fill the volume between the coarse aggregate completely (*Rockpocket*). • Note: Hence, the concrete cover of the reinforcement as well as the bond between the bars and the concrete is not provided or reduced. *Rockpockets* are also called Honeycombing.	
Cavity[1]	• Small air voids • Mostly on vertical surfaces	• Inadequate rheological properties (viscosity, yield point) of the concrete • Appears due to bad compacting after having poured the concrete into the formwork. Therefore, insufficient deaeration of the concrete follows causing small "dots" on the surface. • Note: At these spots the concrete cover is reduced. Cavities of usual size have no impact on the building assessment.	
Concrete Corrosion (ConcreteC)	• Includes the visually similar defects: Washouts, Concrete corrosion and generally all kinds of planar corrosion/erosion/abrasion of concrete. • Note: We summarize all these "planar corrosion defects" in this class because they are visually hard to differ. According to inspection standards they have to subdivided which requires strong expertise in building defects.	• Washouts appear on building parts that are constantly in contact with running water leading to the erosion of the concrete, *e.g.*, abutment walls or bridge piers in rivers. • *Concrete corrosion* can appear as a result of frost-thaw cycles, loss in succession to chemical attacks or abrasion (mechanical or action of acid and salt solutions). • Note: In dacl10k_v1 *Concrete Corrosion* (ConcreteC) was called "Washouts/Concrete corrosion" (WConccor).	

[1] *Cavity* was added in dacl10k_v2 which was formerly included in *Rockpocket*.

Table 3. Concrete defects, their visual appearance, cause and example images with according annotation. (continued)

Damage	Visual appearance	Cause	Example
Hollowarea	• *Hollowareas* are not visually recognizable but their markings made with crayons (mostly yellow, red or blue) during close-up/hands-on inspections. • Note: The outer edge of the marking is considered as the boundary of the according area. We annotate every chalk marking that approximately forms a closed geometric figure. Single lines are not labeled as *Hollowarea* as they are often used for the marking of *Cracks*.	• Corrosion of the subjacent reinforcement which leads to a volume increase surrounding the reinforcement bar and detaching of the concrete area that covers the bars. • *Hollowareas* are usually the preliminary stage of *Spallings* and *Exposed Rebars*. • Note: Recognizing *Hollowareas* is very important as the falling concrete parts can cause severe damage. Therefore, hollow sounding areas are usually removed instantly.	
Spalling	• Spalled concrete area revealing the coarse aggregate • Significantly rougher surface (texture) inside the *Spalling* than in the surrounding surface	• Corrosion of the subjacent reinforcement leading to an increase in volume and consequent spalling of the concrete cover • Frost-thaw cycles of intruded water (deeper than *Washouts/Concrete corrosion*) • Impact from vehicles, damaging during assembly of the building part or removal of the formwork, manufacturing faults	
Restformwork	• Left pieces of formwork in joints or on the structure's surface • Restformwork can be made of wood and polystyrene (PS). • PS is often used as a placeholder in joints during concreting.	• After the concrete hardening has ended, PS is often forgotten to be removed (*e.g.*, in the joint between the abutment wall and the superstructure). • Note: If *Restformwork* is not removed, water may be hindered to drain which can lead to other defects (*e.g.* *Spalling*, *Exposed Rebars*, *Rust*).	

Table 3. Concrete defects, their visual appearance, cause and example images with according annotation. (continued)

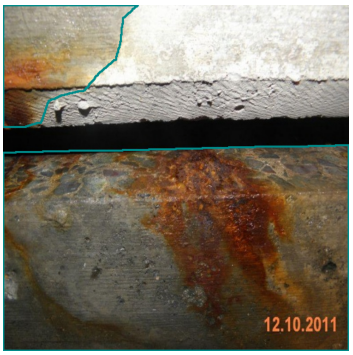
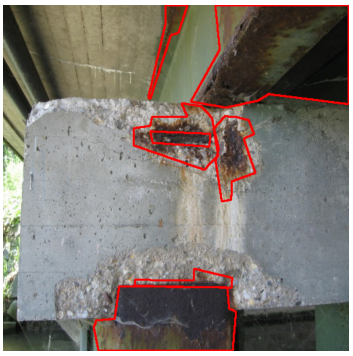
Damage	Visual appearance	Cause	Example
Wetspot	• Wet/darker mirroring area	• Water is hindered to drain (through *Restformwork*) or can't drain properly due to damaged *Drainage*, leaky *Expansion Joints*, *Joint Tapes*, or *Cracks* in the bridge deck. • Note: There may be temporary *Wetspots* due to recent rainfall which can be irrelevant for the bridge assessment. But, they may also indicate that the according area has to be observed in detail due to the greater exposition. In addition, the water can carry deicing salt (on road bridges). Usually, those areas are chosen to execute further investigations such as drill tests in order to determine the carbonation depth or chloride content.	
Rust	• Reddish to brownish area • Often appears on concrete surfaces and metallic objects	• Rust on the concrete surface originates from oxidation of the subjacent or neighboring reinforcement bars, or neighboring metallic building parts. The bars can corrode as a result of loss of the alkaline protective layer provided by "un-carbonated" concrete (pH value > 9.5). If the pH value drops due to the further carbonation of the concrete (pH value ≤ 9.5), which is unavoidable over time, the reinforcement can oxidize. • The carbonation is accelerated by *Cracks Rockpockets* and porous concrete because of faster intruding of water and carbon into the building part. In addition, the oxidation is intensified by deicing salts which is one of the most severe problems on road bridges.	
Graffiti	• All kinds of paintings on concrete and objects apart from defect markings		
Weathering	• Summarizes all kinds of weathering on the structure (*e.g.* smut, dirt, debris) and Vegetation (*e.g.* plait, algae, moss, grass, plants). • Weathering leads to a darker or greenish concrete surface compared to the rest of the surface.	• Directly weathered areas • Result of *Wetspot* • Note: *Weathering* itself is not a severe defect. The main issue is that it can obscure other defects (*e.g.* corroded reinforcement or cracks). *Weathering* is also called Contamination.	

Table 4. General defects, their visual appearance, cause and example images with according annotation.

Object	Visual appearance	Functionality	Example
Exposed Rebars	• Exposed Reinforcement (non-prestressed and prestressed) and cladding tubes of tendons • Often appears in combination with *Spalling* or *Rockpocket*, and *Rust*	• In reinforced concrete structures, the reinforcement's task concerning the load transfer is to absorb the tensile forces. • *Exposed Rebars* occur due to insufficient concrete cover or corrosion and the consequent *Spalling* of the concrete cover. • The reduction of the reinforcement's cross section due to *Rust* significantly influences the structural integrity of the according building.	
Bearing	• All kinds of bearings, such as rocker-, elastomer- or spherical-bearings	• Bearings transfer the load from the superstructure to the substructure. • Note: Bearings can show *Rust* and *Cracks* as well as deformation due to settlements in the abutments, overloading or creeping of the bridge. The deformation or damaging of bearings can result in load redistribution which is not considered in the structural design of the bridge.	
Expansion Joint (EJoint)	• Located at the beginning and end of the bridge • Assembled cross to the longitudinal bridge axis	• *Expansion Joints* compensate the thermal longitudinal expansion of the bridge deck and superstructure. • Note: Mostly, *Expansion Joints* are corroded (*Rust*) and weathered (*Weathering*) which hinders their ability to compensate the enlargements due to changes in temperature.	

Table 5. Objects, their visual appearance, functionality including their role for the bridge assessment and example images with according annotation.

Object	Visual appearance	Functionality	Example
Drainage	• All kinds of pipes and outlets made of Polyvinylchlorid or metal mounted on the bridge.	• The *Drainage* directs water (often contaminated with deicing salt) away from the bridge. • The draining of water is important for the durability of the bridge. If the water can't drain properly, especially when it's contaminated with deicing salts, the bridge's deterioration is accelerated leading to defects, such as *Spalling*, *Exposed Rebars* or *Rust*	
Protective Equipment (PEquipment)	• Railings, traffic safety features (*e.g.*, steel rail, guide rails, impact attenuation device)	• Geometric adequacy and structural capacity of the *Protective Equipment* is important with respect to the traffic safety. • The rail types and installation heights and minimum clearances must be checked. • Note: Mostly, they are corroded (*Rust*) or deformed due to vehicle impact.	
Joint Tape (JTape)	• All joints that are filled with elastomer or silicon • Note: Originally, *Joint Tape* means an elastomer strap at the end and beginning of relatively small bridges.	• *Joint Tapes* compensate longitudinal enlargements due to changes in temperature (like *Expansion Joint*). • Note: A *Joint Tape* is damaged when it's ruptured or twisted. *Joint Tape* is also called Bridge Seal.	

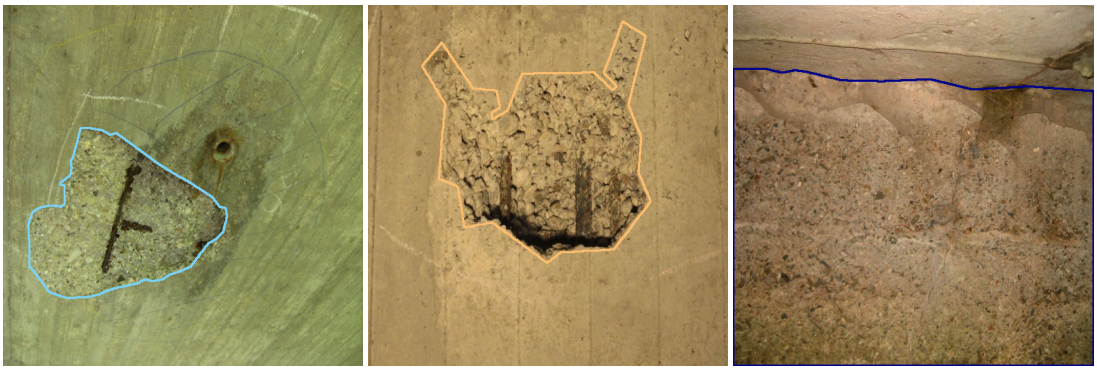
Table 5. Object classes, their visual appearance, functionality including their role for the bridge assessment and example images with according annotation. (continued)



(a) Challenging samples showing a *Crack*: in combination with strong *Weathering*, in combination with *Graffiti* and partially overlapped by vegetation (*Weathering*).



(b) *Crack* vs. *Alligator Crack*, *Weathering* vs. *Wetspot*, *Restformwork* covered with concrete slurry (no ground-truth annotation for better visibility).



(c) *Spalling*, *Rockpocket*, *Washouts/Concrete corrosion*.

Figure 1. Challenging defects. All subcaptions describe the images from left to right.



(a) No clear border of the defect (*Rust*), the class (*Protective Equipment*) is stretched from fore- to background.Many overlapping classes with partially low contrast like the border and hatching of the *Hollowarea* (pale red), and the *Efflorescence* (dark blue) next to the bright, dry and healthy concrete surface.



(b) Image quality problems: lens flares, partial overexposures due to flashlight, and sunlight.

Figure 2. General problems across the dataset. All subcaptions describe the images from left to right.

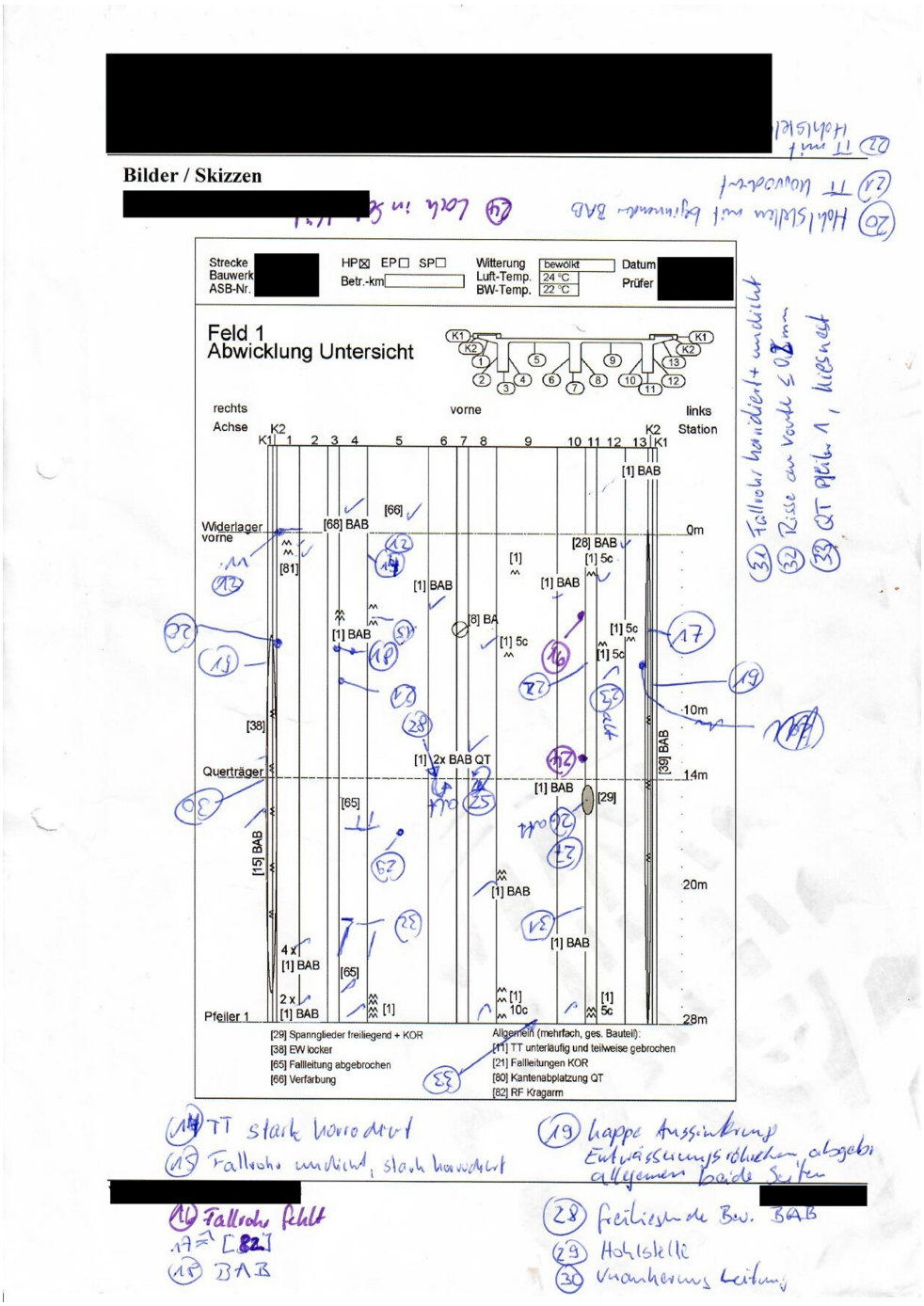


Figure 3. Damage sketch from a real bridge inspection with three T-beams. The sketch represents the bridge flipped-open along its longitudinal axis. Thus, the "vertical lines" represent the edges of the bridge cross-section. The defects are numbered in chronological order. Exemplary damage-information separated by comma (damage-number, damage-class, longitudinal, and cross-section dependent localization): #28, *Spalling* ("BAB") with *Exposed Rebars* ("freiliegende Bewehrung"), at the cross girder ("Querträger"), on the left side of the web (6); #29, *Hollowarea* ("Hohlstelle"), approximately 3m from the cross girder ("Querträger"), on the flange (5); #31, *Drainage* ("Fallrohr") with *Rust* ("korrodiert") which is leaking ("undicht"), approximately 7m from the cross girder ("Querträger"), on the bottom of the web (11).

References

[1] Brent M. Phares, Glenn A. Washer, Dennis D. Rolander, Benjamin A. Graybeal, and Mark Moore. Routine highway bridge inspection condition documentation accuracy and reliability. *Journal of Bridge Engineering*, 9(4):403–413, July 2004. 3

dacl-challenge: Semantic Segmentation during Visual Bridge Inspections

Johannes Flotzinger[1], Philipp J. Rösch[1], Christian Benz[2],
Muneer Ahmad[3], Murat Cankaya[1], Helmut Mayer[1],
Volker Rodehorst[2], Norbert Oswald[1], Thomas Braml[1]

[1]University of the Bundeswehr Munich, Germany, {firstname.lastname}@unibw.de
[2]Bauhaus-Universität Weimar, Germany, {firstname.lastname}@uni-weimar.de
[3]NetApp, Germany, muneer.ahmad@netapp.com

Abstract

Civil engineering structures – such as bridges – form an essential component of the transportation infrastructure. A failure of an individual structure can result in enormous damage and costs. The economic costs caused by the closure of a bridge due to congestion can be many times the costs of the bridge itself and its maintenance. Thus, it is mandatory to keep these structures in a safe and operational state. In order to ensure this, they are frequently inspected. However, the current inspection process is error-prone and lengthy. Especially the damage documentation using a hand-drawn sketch causes inconsistencies in the building assessment. On the other hand, recent advancements in hardware enable the deployment of computer vision models for increasing the quality, traceability, and efficiency of structural inspections. Such models are the key element of digitized structural inspections and the basis for automated damage classification, measurement and localization on a pixel-level. Current datasets available for this task suffer from limitations in both size and diversity of classes, raising concerns about their applicability in real-world contexts and their effectiveness as benchmarks. Addressing this problem, we introduced "dacl10k" (damage classification), a diverse dataset designed for multi-label semantic segmentation. Comprising 9,920 images extracted from real-world bridge inspections, "dacl10k" stands out by its comprehensive coverage. It includes 13 damage classes and 6 crucial bridge components pivotal in assessing structures and guiding decisions on restoration, traffic restrictions, and bridge closures. To accelerate progress in baseline development, we organized the "dacl-challenge", inviting enthusiasts in damage recognition to vie for training the best performing model on the "dacl10k" dataset. The competition is at the core of the "1st Workshop on Vision-Based Structural Inspections in Civil Engineering", hosted at WACV 2024.

In total, 23 participants registered for the challenge, with eight achieving a performance superior to our baseline. The best result shows a mean intersection-over-union of 51%. This paper delineates the challenge's structure, introduces the dataset utilized, presents the achieved outcomes, and outlines prospective avenues for further exploration in this domain.

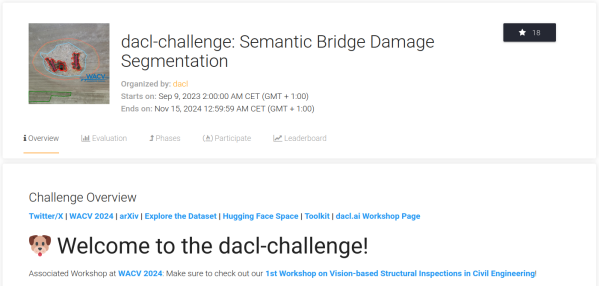


Figure 1. Screenshot of the eval.ai challenge website.

1. Introduction

Civil engineering structures are the key element of public infrastructure. They can be subdivided into the fields of transportation (e.g., road and railway bridges, tunnels, airports, harbors), energy supply (for instance, wind turbines, offshore constructions, power plants), water supply (for example dams, pumping stations, pipelines) and waste management (*e.g.*, sewers, wastewater and storage facilities). It is of great importance for our daily life that these structures operate technically faultlessly and their maintenance measures are well planned. A failure of an individual structure can result in enormous damage and costs. For example, the economic damage caused by the closure of a bridge due to congestion can be many times the cost of the bridge itself and its maintenance. To avoid this, such structures need to

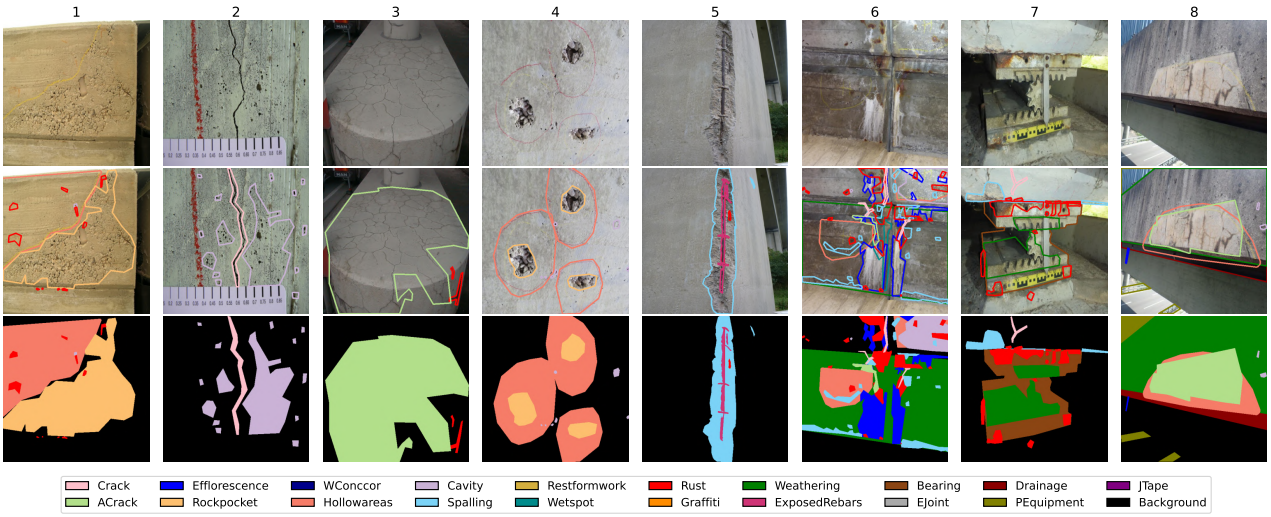


Figure 2. Example annotations from dacl10k. Top row: original image. Middle row: polygonal annotations. Bottom row: stacked masks. From left to right, the images display the individual classes: 1. *Rockpocket, Hollowarea, Rust, Cavities*; 2. *Crack, Cavities*; 3. *Rust, Alligator Crack*; 4. *Hollowareas, Rockpockets, Cavities*; 5. *Spalling, Exposed Rebars, Rust*; 6. *Efflorescence, Rust, Spallings, Cracks, Wetspot, Cavities, Hollowarea, Weathering, ACrack*; 7. *Rust, Weathering, Bearing, Crack, Spallings*; 8. *Protective Equipment, Efflorescence, Drainage, Cavities, Hollowarea, Weathering, ACrack*. The following classes are abbreviated: *Alligator Crack* (ACrack), *Washouts/Concrete corrosion* (WConccor), *Expansion Joint* (EJoint), *Protective Equipment* (PEquipment) and *Joint Tape* (JTape).

be frequently inspected, in order to: (*i*) plan maintenance works, (*ii*) prevent abrupt failure, and (*iii*) avoid unnecessary closures or load limitations.

Facing a continually growing percentage of infrastructure reaching a critical age concerning the occurrence of damage, their inspection is more important than ever. In the case of bridges, many governmental reports describe the current state as severe. For instance, "more than 222,000 U.S. bridges need major repair work or should be replaced [In 2023]" [1] or "... at least 25,000 road bridges [in France] are in poor structural condition ..." [4]. At the same time, governments have to deal with a lack of funding and the construction industry is confronted with a reduced availability of skilled staff [3]. This underlines the demand for a more efficient examination pipeline.

However, the current inspection process for civil engineering structures, consisting of damage recognition, assessment and documentation, is mostly performed in an analogue way. In the case of bridge inspection, this means that an inspector visually recognizes each defect, noting the damage class, values of measurements, and the location in a 2D sketch. In addition, the inspector assigns an image filename in order to combine the defect information with the corresponding image later when writing the assessment, or rather the inspection report.

Using computer vision methods to automatically classify, measure and localize defects and building components as part of smartphone applications or software for unmanned aircraft systems (UAS) makes the inspection of such infrastructure much more efficient [27]. As damage recognition and documentation is in large parts performed automatically by software, the inspector can concentrate on damage assessment and interactions between defects. In addition, the assessment can be done more frequently. This leads to a higher confidence in the determination of structural integrity, traffic safety and long-term durability. Additionally, a higher frequency of inspections allows for extending the useful life of buildings in a critical condition, or avoiding/postponing load limitations (e.g., ban of heavy traffic on bridges). Furthermore, capturing the damage (and objects) may be executed by non-professionally trained personnel, which can compensate for the problem of staff shortages to some extent. This problem is present in many countries such as Germany, France, and the US.

Compared to other domains, such as autonomous driving [7, 11, 15, 30, 35] or medicine [26, 29, 31], infrastructure defect recognition received only little attention in the past. Furthermore, the existing datasets for this particular task are constrained by their limited size and the lack of diversity in the classes. This raises doubts about how well they comply with real-world situations and questions their efficacy as benchmarks.

A central contribution to this problem is the dacl10k dataset [14], which includes images collected during inspections of concrete bridges acquired from databases at authorities and engineering offices, thus, representing real-

world scenarios. Concrete bridges are the most common bridge type along with steel, steel composite, and wooden bridges. The dataset provides polygonal annotations for 13 bridge defects as well as six bridge components that play a key role in the structure assessment. Two groups of annotators, civil engineering students and a professional annotation team, tagged the images with a semantic segmentation by hand. Yet, dacl10k is not restricted to concrete bridges. Its concrete and general defect groups can appear on any building made of concrete (*e.g.*, *Crack*, *Spalling*, etc.), and some on steel structures (for instance, *Rust*, *Graffiti*, etc.). Therefore, it is relevant for most civil engineering structures.

To popularize dacl10k, we organize the "dacl-challenge", which aims to find the best multi-label semantic segmentation models for the novel, highly diverse, large-scale dataset. The aim of the challenge is to provide a benchmarking platform for the automatic visual inspections of bridges. The platform of the challenge will be maintained also after completion of the challenge for future benchmarking.

The challenge is the central part of the "1st Workshop on Vision-Based Structural Inspections in Civil Engineering". It focuses on the visual recognition of defects and building components utilizing innovative computer vision methods to increase the efficiency of the laborious inspection process of civil engineering structures. We invite experts who successfully employ computer vision for visual inspections to present their applications and emerging challenges at the workshop.

The challenge and the workshop will highlight the yet mostly unnoticed problem of visual structural inspections for public infrastructure. Furthermore, a community for the field of computer vision-based inspections of civil engineering structures is to be created promoting research in this field. The gain in efficiency for inspections will lead to safer and more cost-effective public infrastructure.

2. Related Work

Damage Recognition on Built Structures. In recent years, the application of damage recognition on built structure has evolved. Former datasets and models focus on image classification only. Several datasets deal with binary classification (mostly crack vs. no crack) [19, 23] and a few also with multiple classes [18, 28]. Recently, the focus has moved towards pixel-level segmentation tasks, which is more challenging but also more helpful for the inspection process. Important work in this field is given by Crack-Seg9k [22] and UAV75 [5] for semantic crack segmentation and S2DS [6] for multi-class semantic segmentation.

Multi-label Semantic Segmentation. Semantic Segmentation is the task to predict object classes in images on pixel level. Many powerful models have been developed in recent years, e.g. DeepLabV3+ [8, 9], Feature Pyramid Network (FPN) [20] or SegFormer [34]. But with most datasets, there is only one correct class per pixel. In our case, a pixel can belong to several classes. For example, there may be *Spalling* that shows also signs of *Rust*. Therefore, we are no longer referring to multi-class semantic segmentation, but to multi-label semantic segmentation.

Workshops. Civil engineering, particularly visual inspection has recently increased its visibility in the form of workshops at computer vision conferences. For example, at ECCV2022 the *Computer Vision for Civil and Infrastructure Engineering Workshop* [16] was organized. The workshop is motivated by "Civil and infrastructure engineering are corner stones in modern society". It aims at different built structures such as bridges, roads, sewerage, and buildings. Computer vision is seen as an important component for automated inspection, but it is also emphasized that it can "analyze work patterns or detect hazardous situations at, e.g., construction sites".

The *1st workshop on Vision-based InduStrial InspectiON (VISION)* [2] strived to be a platform for sharing improvements in research and addressing emerging practical challenges in industrial inspection using computer vision. It included the two challenges object detection for industrial products (metal cylinder, rings, etc.) and object generation.

3. Challenge Design

After our challenge was accepted on August 22, 2023, we set it up on eval.ai[1] and started on September 9, 2023. See Fig. 1 for a screenshot of the challenge website and Tab. 1 for a detailed timetable. In Tab. 2 you can see which release the splits belong to and which splits were used for the leaderboard.

For promotion a Hugging Face Space[2] was created to give an impression how results can look like (see Fig. 3). Moreover, an interactive data visualization tool was made available via Voxel51.[3] The team also provided a Python toolkit[4] to lower the entry barrier to the challenge. Participants were kept up to date via X.[5] The challenge was split into two phases: the development phase and the final test phase.

Development Phase. In the development phase, images for the training (n=6,935), validation (n=975) and testing

[1] https://eval.ai/web/challenges/challenge-page/2130/overview
[2] https://huggingface.co/spaces/phiyodr/dacl-challenge
[3] https://try.fiftyone.ai/datasets/dacl10k/samples
[4] https://github.com/phiyodr/dacl10k-toolkit
[5] https://x.com/dacl_ai/



Figure 3. Hugging Face Space with dacl-challenge demo using SegFormer MiT-b1.

(n=1,012) split were made available via AWS and GigaMove. For the training and validation split, additionally annotation files in a labelme-like format[6] were provided. Participants could upload their results to eval.ai to receive feedback about their model performance. Some results were publicly visible in the leaderboard, others opted for a private listing only. We also provided a baseline (cf. Sec. 3.3) to act as a reference point for the participants. The development phase was active until October 27, 2023 (48 days).

Testfinal Phase. In the second phase another 998 images – called "testchallenge" – were provided. The task in this phase was to create predictions for the images from testdev and testchallenge combined, termed "Testfinal". Participants could only see their own results and could no longer compare themselves with others. Due to server problems at eval.ai, the end of the challenge had to be postponed from November 10th to November 14th. This phase has been active for 18 days.

[6] https://github.com/wkentaro/labelme

September 9, 2023	Start of Development Phase
October 27, 2023	Start of Final Phase
November 14, 2023	End of Challenge
November 14, 2023	Fact-Sheet Submission
November 21, 2023	Release of Final Results
January 7, 2024	Workshop and Prize Ceremony

Table 1. Timetable of the dacl-challenge.

Split	Released in phase: Devel.	Released in phase: Testfinal	Used in leaderboard: Devel.	Used in leaderboard: Testfinal
train	✓			
validation	✓			
testdev	✓		✓	✓
testchallenge		✓		✓

Table 2. Splits released and used in different phases.

Fact Sheet. In order to win prizes, a fact sheet[7] had to be submitted via the Uni-Weimar cloud. Participants had to explain their approach on two pages and provide a variety of relevant parameters concerning their best model. We also encouraged everyone to openly share their code.

Workshop. The workshop is scheduled to take place on January 7, 2024. It will include keynote presentations and the prize ceremony at which the three best results will be honored. Prize money of $3,000, $2,000 and $1,500, respectively, will be awarded. To this end, the ranking on the "Testfinal" leaderboard for the mIoU metric (see Sec. 3.2) is used combined with the submission of a proper fact sheet. After the end of the challenge, we have moved the evaluation process to CodaLab[8] so that the testdev split can still be used in research.

3.1. Challenge Dataset: dacl10k-v2

The challenge dataset is the second version of dacl10k, a multi-label segmentation dataset consisting of (mostly smartphone) images from bridge inspections in Germany (see Figure 2). Its images are highly diverse with respect to the lighting condition, camera pose and resolution. The annotations were created by civil engineering students and an external team using a detailed annotation guideline.

"dacl10k-v2" is an improved version of the initial dataset, which was introduced in Flotzinger *et al.* [14]. Apart from a general cleaning, the main difference between v2 and v1 is the separation of the *Rockpocket* and *Cavity* defect class. In Figure 2 example 1 shows *Rockpocket* in pale orange and example 2 displays *Cavity* in pale purple. This step has been taken due to the fact that in [14] weak performance has been reported on the *Rockpocket* damage (29%), noting that from the application point of view a further differentiation would be beneficial. In the previous version,

[7] https://dacl.ai/assets/call-for-fact-sheets.pdf
[8] https://codalab.lisn.upsaclay.fr/competitions/16317

both defects were included in *Rockpocket*. They were originally merged in one class as their cause is similar, namely insufficient deaeration of the concrete. Additionally, a *Cavity* looks, when zoomed in, like a small *Rockpocket*. So, they mainly differ in size.

Figure 4 shows the significantly differing number of pixels for each class in the dataset. The classes *Weathering* and *PEquipment* have the highest number of pixels under all the classes within the dacl10k dataset, while *Exposed Rebars*, *Joint Tape*, *Rockpocket* and *Restformwork* show the lowest number of pixels.

Tab. 3 depicts a more detailed insight about the dataset. While *Cavity* has the topmost number of instances, *ACrack* has only a few instances. Yet, *ACrack* shows the highest pixels to polygon and pixel to image ratio – hence indicate large surface damages. The damage with the smallest area per polygon is *Cavity*. Furthermore, *Crack* is the damage with the lowest number of pixels per image. Regarding the major change from dacl10k-v1 to dacl10k-v2 (splitting the *Rockpocket* class) it can be noted that the share of this class with respect to the total pixel area was reduced from 1.48% to 0.14%. The share of *Cavity* in the second version amounts 0.39% signifying that many false positives were deleted.

The data files for the development phase[9] and the final test phase[10] are available online.

3.2. Evaluation Protocol

In the context of the challenge the standard metric for semantic segmentation is used, the *Intersection-over-Union* (IoU, also referred to as *Jaccard index*). Since multiple labels for each pixel are allowed, the IoU is computed class-wise and subsequently aggregated to the *mean IoU* (mIoU). The mIoU is the (unweighted) arithmetic average of the class IoUs. The metrics are defined as:

$$\text{IoU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{1}$$

$$\text{mIoU} = \frac{1}{N} \sum_{i=1}^{N} \text{IoU}_i \tag{2}$$

TP, FP, and FN refer to the true positives, false positives, and false negatives, respectively. N represents the number of classes, here $N = 19$, and IoU_i refers to the IoU of the i-th class. The ranking and challenge winner is determined based on the highest mIoU.

3.3. Baseline

In [14], three architectures were compared. These were DeepLabV3+ and FPN with MobileNetV3-Large, EfficientNet-B2 and EfficientNet-B4 encoders respectively, and SegFormer MiT-b1. Each model was trained on dacl10k-v1 dataset with different learning rates and only the best results were reported.

For the challenge we use "dacl10k-v2" (cf. above). We

[9] https://dacl10k.s3.eu-central-1.amazonaws.com/dacl10k-challenge/dacl10k_v2_devphase.zip (4.8 GB)

[10] https://dacl10k.s3.eu-central-1.amazonaws.com/dacl10k-challenge/dacl10k_v2_testchallenge.zip (0.5 GB)

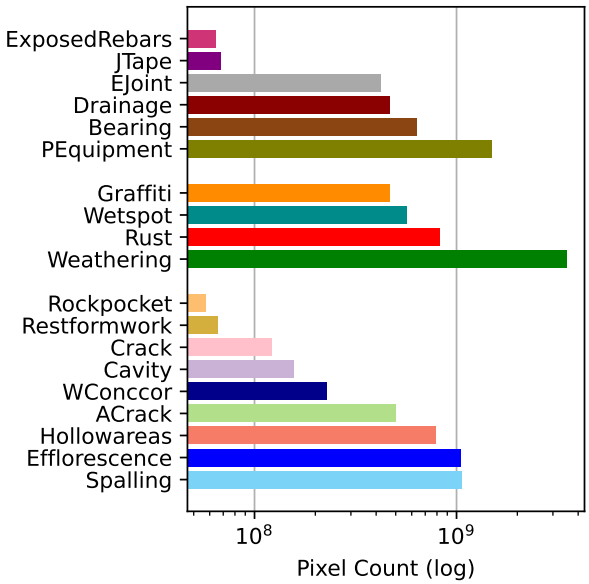


Figure 4. Pixel counts with respect to each class in dacl10k based on the original image sizes. The bars are arranged according to group affiliations.

Class	#polyg./ image	#pixels/ polyg.	#pixels/ image	%polyg.	%pixels
Crack	1.8	27,434	49,503	4.07	0.3
ACrack	1.12	947,430	1,062,822	0.48	1.23
Efflorescence	2.3	208,256	478,408	4.6	2.59
Rockpocket	1.74	71,039	123,780	0.74	0.14
WConccor	1.35	457,780	617,008	0.45	0.56
Hollowareas	1.21	415,309	504,532	1.74	1.95
Cavity	6.77	13,470	91,182	10.67	0.39
Spalling	2.61	85,484	223,106	11.33	2.62
Restformw.	1.2	50,146	59,929	1.2	0.16
Wetspot	1.47	273,825	403,511	1.89	1.4
Rust	3.62	46,640	168,782	16.2	2.04
Graffiti	2.29	172,849	396,007	2.47	1.15
Weathering	1.43	615,624	881,279	5.21	8.66
ExposedR.	2.25	25,770	58,035	2.29	0.16
Bearing	1.45	422,297	613,103	1.38	1.57
EJoint	1.12	701,120	784,370	0.55	1.05
Drainage	1.37	230,665	316,051	1.85	1.15
PEquipment	1.22	641,596	785,870	2.12	3.67
JTape	1.19	49,370	58,529	1.26	0.17
Background	3.3	925,898	3,051,636	29.49	73.72

Table 3. Overall statistics of the dataset regarding average number of polygons per image, number of pixels per polygon, number of pixels per image, share of polygons and share of pixels. Midrules separate the classes according to their group affiliations.

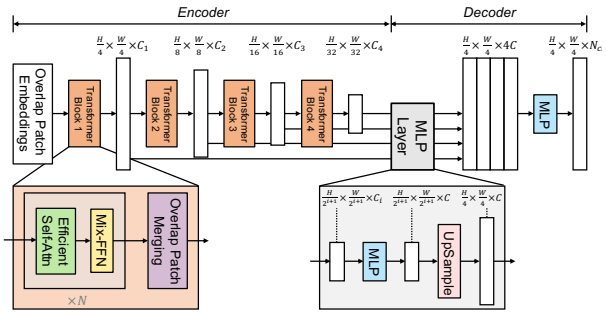


Figure 5. SegFormer architecture. Picture credit: [34].

only provide one model, which serves as a simple baseline and reference point for the participants.

As baseline SegFormer MiT-b1 [34] is used, which has a relatively small 13.1 million parameter encoder with a mulit-label segmentation head. The model was pre-trained on the ImageNet-1k dataset and the architecture can be investigated in Fig. 5. For the baseline, SegFormer was trained for 10 epochs for the development leaderboard (mIoU of 34.6%) and 30 epochs for the Testfinal leaderboard (37.2%). All other training parameters remained unchanged from [14]. In Tab. 4 we named the baseline *dacl-squad*.

4. Challenge Results

In this section the results of the challenge are presented. Furthermore, information about the submitted approaches is provided.

4.1. Leaderboard

Testfinal. Tab. 4 shows the state of the leaderboard at the end of the challenge. The winner is determined based on the Testfinal leaderboard. With an mIoU of 51% *Sheoran* wins the dacl-challenge at WACV 2024 followed by *Bridge Protector* (50.5%) and *Winning Wieners* (50%). *Sheoran* performs on top of five of the 19 classes. The runner-up *Bridge Protector* performs best on six classes, however, showing comparatively low performance on the Washouts/Concrete corrosion and Wetspot class. It performs noticeably well on the Crack class surpassing all others by 4.5% points. *Sheoran* shows a relatively balanced performance over all classes and performs particularly well on the Wetspot and Drainage class, which paved the road to victory. The mIoU of the top 3 is within a range of 1% indicating that nuances lead to the final ranking.

In terms of class performance, Washouts/Concrete corrosion is found to be particularly difficult. As pointed out by *Bridge Protector*, this class is underrepresented in the training set. Furthermore, Cavity and Crack are challenging classes for all approaches. PEquipment is the class best predicted across all approaches peaking at 83.1% for *Bridge Protector*.

Development. The bottom part of Tab. 4 provides insights into the performance of the participants during the development phase. Some participants including *Bridge Protector* and *mp269546* did not submit results in the development phase. Apart from *Shivesh Khaitan*, the order of the other approaches corresponds to the Testfinal set, with *Sheoran* also ranking first.

4.2. Overview

Despite noticeable overlap in the architectures, the landscape of the top 10 approaches is diverse with respect to the choice of backbones and training configurations. Many submissions are based on established segmentation models such as FPN, SegFormer, ConvNeXt, UPerNet, U-Net, and Mask2Former. Others also explore EfficientNet and Yolo-v8. Winning Wieners propose a novel method by combining FPN with MaxViT. Used backbones include EVA-02, ConvNeXt, InternImage, MaxViT, and BEiT, mostly pre-trained on ADE20K or ImageNet. The submissions usually rely on standard augmentation strategies including crops and flips. *Sheoran* emphasizes the use of CutMix and Winning Wieners apply RandAugment. Color jittering showed minor effectiveness especially for color sensitive classes, as mentioned by *mp269546*.

A recurring and effective feature is the use of ensembles. *Sheoran* selectively groups together predictions from multiple models while *Winning Wieners* combine the predictions trained on different folds of the dataset. A number of approaches apply test time augmentation (TTA). The used losses span from BCE, Jaccard, Dice over recall, focal up to Lovasz loss. AdamW is the prevailing optimizer, only *Sheoran* (RangerLars) and *Winning Wieners* (MADGRAD [12]) use different losses. Post-processing by filtering for small object removal is reported by *Bob der Baumeister*.

Four out of the top 10 approaches report the use of the MMSegmentation library even though multi-label segmentation is not natively supported. *Bridge Protector* circumvents this problem by learning 19 separate models, each covering one class. *Sheoran* later moved to segmentation-models-pytorch for finetuning. In total three out of the top 10 report to use the segmentation-models-pytorch library.

4.3. Top 3 Approaches

First Place. The winning approach by *Sheoran*[11] is based on an ensemble of the predictions of multiple models. After training using the MMSegmentation framework, the model was transferred to the segmentation-models-pytorch library for handling of the multi-label segmentation. Softmax was

[11] https://github.com/HarshitSheoran/dacl10k

Rank	Team	mIoU [%]	IoU [%] Concrete Defects: Crack	ACrack	Efflorescence	Rockpocket	WConccor	Hollowareas	Cavity	Spalling	Restformwork	General Defects: Wetspot	Rust	Graffiti	Weathering	Objects: ExposedR.	Bearing	EJoint	Drainage	PEquipment	JTape
												Testfinal									
1	Sheoran	**51.0**	34.1	58.2	49.9	35.3	13.6	**64.7**	22.2	**53.4**	39.9	**33.6**	52.1	**73.4**	43.8	44.6	74.4	66.3	**76.7**	82.5	49.4
2	Bridge Protector	50.5	**41.1**	**58.8**	**50.0**	30.7	10.4	62.0	22.9	51.4	**45.9**	26.7	**52.3**	73.0	42.2	46.0	76.6	66.9	71.6	**83.1**	48.4
3	Winning Wieners	50.0	35.9	50.7	48.9	28.0	16.1	63.8	19.3	50.7	42.2	31.4	**52.3**	73.0	**45.7**	**47.3**	**77.2**	63.8	71.8	82.8	**50.0**
4	Bob der Baumeister	49.7	36.6	50.4	48.3	**37.3**	**20.3**	61.7	19.4	50.2	38.9	29.8	**52.3**	71.9	43.7	42.3	73.7	64.1	75.2	79.3	48.8
5	mp269546	47.6	29.5	50.1	46.6	27.6	15.0	59.2	16.5	50.0	37.1	29.7	51.0	72.5	42.7	42.1	69.2	**68.9**	69.1	82.6	44.3
6	Shivesh Khaitan	45.7	30.4	48.9	43.8	22.6	14.2	57.4	23.2	48.3	33.4	25.9	49.0	70.1	42.0	37.4	68.7	65.3	64.7	78.0	44.4
7	Lars Nieradzik	45.1	31.2	55.5	42.9	29.5	12.5	58.4	**26.1**	46.2	31.4	27.1	47.7	66.3	40.5	34.2	67.7	58.5	62.3	74.6	44.5
8	SoloLearn	37.3	20.3	33.4	34.1	18.9	12.5	49.3	10.2	38.0	28.0	20.4	35.6	69.6	34.7	10.2	69.0	55.1	56.5	78.7	33.3
9	*dacl-squad (30 epochs)*	*37.2*	*23.5*	*38.1*	*40.1*	*19.4*	*7.8*	*46.4*	*10.6*	*42.9*	*24.7*	*20.2*	*44.7*	*62.5*	*36.0*	*33.2*	*53.9*	*49.0*	*52.1*	*69.2*	*33.1*
10	whis	36.1	15.1	25.7	36.0	19.9	11.0	45.5	14.5	43.3	35.7	8.0	3.1	45.2	36.3	33.8	62.4	61.4	71.1	77.7	40.4
												Development									
1	Sheoran	**49.8**	35.9	49.3	**46.4**	31.7	13.6	**63.5**	29.0	**53.0**	41.6	**35.5**	50.9	**71.3**	44.9	34.9	70.0	**67.5**	**78.0**	81.1	47.9
2	Winning Wieners	48.5	37.1	**49.5**	42.8	19.4	**19.6**	62.8	19.9	52.2	**41.9**	29.7	50.6	69.8	**45.4**	**39.2**	**71.0**	65.9	69.2	**82.5**	**53.4**
3	Bob der Baumeister	48.3	**38.3**	44.2	44.5	**32.4**	17.8	58.9	24.4	49.9	38.6	29.8	**51.5**	70.1	42.6	33.9	66.0	66.1	77.8	80.5	50.0
4	Lars Nieradzik	43.3	32.8	46.2	38.1	19.7	15.3	54.3	**32.1**	45.8	30.1	26.5	47.0	63.2	40.1	25.1	65.9	61.0	55.5	78.0	46.3
5	SoloLearn	35.6	18.4	25.4	29.8	22.8	14.1	40.9	24.6	35.5	28.2	18.3	31.1	64.6	40.7	20.7	58.9	44.8	51.4	73.5	32.1
6	whis	35.3	14.2	19.4	30.1	32.0	12.4	40.4	18.9	41.1	34.6	4.6	2.6	42.2	36.6	23.2	62.4	62.3	74.3	75.8	43.7
7	*dacl-squad (10 epochs)*	*34.6*	*22.8*	*34.4*	*34.8*	*12.4*	*7.4*	*45.4*	*19.7*	*41.8*	*22.2*	*20.2*	*42.2*	*54.8*	*35.6*	*32.6*	*49.4*	*45.8*	*47.1*	*54.8*	*33.8*
8	Shivesh Khaitan	31.7	13.7	33.7	27.8	8.7	8.6	45.2	14.8	37.2	18.2	19.6	29.5	55.8	33.8	24.5	49.2	47.1	43.9	61.1	30.2
9	Untitled	31.4	22.9	23.5	25.3	16.3	8.5	44.4	20.0	38.5	22.1	18.3	39.6	50.8	34.4	31.3	47.3	39.6	20.1	60.9	32.4
10	ComputingStones	31.3	24.2	25.7	32.1	5.3	8.9	37.1	21.2	34.8	24.7	16.6	36.8	48.8	30.4	12.2	53.6	46.2	40.5	60.3	36.1

Table 4. Results on the Testfinal and the Development phase. The final ranking is based on the Testfinal set.

replaced by sigmoid and JaccardLoss and BCELoss were employed. Various augmentations based on the albumentations library were applied (randomly resized crops, flips, rotations, coarse dropout, and the ImageNet normalization) complemented by CutMix [36] augmentations. Optimization was performed with the RangersLars optimizer and the scheduler CosineAnnealingLR adjusted the learning rate. The two models ConvNeXt-Large [25] and EVA-02-Large [13] pre-trained on ADE20K were trained with different initial learning rates, augmentation schemes, and heads (UPerNet and U-Net) resulting in six different models. Scoring about 40% mIoU with MMSegmentation, the EVA-02-Large with all augmentations and a UPerNet head achieved 47.8% after transfer to the segmentation-models-pytorch library. The predictions of the six models were combined for some classes, improving the overall performance.

Second Place. *Bridge Protector* also used the MMSegmentation library to train the Mask2Former [10] model with InternImage-H [33] as backbone. The pre-trained weights from ADE20K were used. In terms of augmentation, cropping and flipping were used followed by normalization. The multi-label property of the dataset is approached by decomposing the task into separate models for each category, resultng in 19 separate models. The predictions of the 19 models are concatenated.

Third Place. The *Winning Wieners*[12] combine the feature pyramid network (FPN) [21, 24] with a multi-axis vision transformer (MaxViT) [32]. FPN joins a bottom-up with a top-down pathway to handle objects across scales more effectively. MaxViT serves as a backbone, which integrates convolutional blocks based on MobileNets [17] with the attention logic from vision transformers. The xlarge MaxViT pre-trained on ImageNet was used. Standard augmentations were applied based on the RandAugment suite. The model is trained using five-fold cross-validation yielding five different models. Threshold tuning is performed for generating an ensemble on the level of predictions.

4.4. Results and Findings

The result of the dacl-challenge indicates that transfer learning on established models is a powerful tool for the domain of visual bridge inspections. Available libraries such as segmentation-models-pytorch and MMSegmentation provide good starting points for learning effective

[12] https://github.com/MIC-DKFZ/semantic_segmentation/tree/challenge/DACL

models. The drawback of MMSegmentation not providing multi-label functionality is resolved by training separate models for each class or moving to the segmentation-models-pytorch library, respectively. Training separate, class-wise models, however, induces significant computational overhead. Furthermore, ensembles appear crucial to achieve top performance by exploiting the strengths of different models. Even though achieving a good performance, the customized method from *Winning Wieners* did not exceed that of the transfer learning approaches. Apparently, the specific method configuration during training including augmentation schemes, optimizer, and other hyperparameters, are the major determinants of performance, rather than architectural considerations.

5. Conclusions

In conclusion, the dacl-challenge hosted at the "1st Workshop on Vision-Based Structural Inspections in Civil Engineering", held during WACV 2024, marks a pivotal moment in advancing the field of computer vision for structural assessments. The participation of 23 teams underscores the interest and significance of leveraging computer vision in ensuring the safety and integrity of civil engineering structures.

The competition witnessed remarkable achievements, yet only eight teams surpassed the established baseline. Notably, the team *Sheoran* utilizing a prediction ensemble from multiple models emerged as the frontrunner, showcasing an mIoU of 51%. Their methodology sets a new benchmark in pixel-level damage recognition, laying the groundwork for future advancements in this domain.

The outcomes of this challenge highlight the potential of computer vision models in revolutionizing the inspection process for civil engineering structures. While celebrating the progress made, it also underscores the need for continued efforts in refining algorithms, enhancing dataset diversity, and exploring new avenues for more accurate, efficient, and reliable structural assessments.

As we move forward, the insights gleaned from this challenge serve as a springboard for further exploration and collaboration within the realm of vision-based structural inspections. We think that the successes and lessons learned from this will catalyze continued innovation, contributing to safer, more resilient infrastructure.

Acknowledgments. We thank Sebastian Mayr, Alexander Kern and Jürgen Hamm for their support. We are most grateful to the enterprises NetApp Deutschland GmbH and especially Ultra Consulting Network GmbH which provided the prize money. Moreover, we gratefully acknowledge the compute time granted by the Institute for Distributed Intelligent Systems and provided on the GPU cluster Monacum One at University of the Bundeswehr Munich. The challenge was supported by the MoBaP research project (IUK 639/003) financed by the Bavarian Ministry of Economic Affairs and the ev.AI.luate project as part of the FFG funding program Mobility of the Future (FO999893305).

References

[1] American Road & Transportation Builders Association (ARTBA). ARTBA bridge report, 2023. 2

[2] Haoping Bai, Gokberk Cinbis, Meng Cao, Tatiana Likhomanenko, and Shancong Mou Oncel Tuzel. 1st workshop on Vision-based InduStrial InspectiON (VISION@CVPR2023). https://vision-based-industrial-inspection.github.io/cvpr-2023/. Accessed: 2023-11-15. 3

[3] European Investment Bank. and Ipsos Public Affairs. *EIB investment survey 2023: European Union overview.* Publications Office, 2023. 2

[4] Bruno M. Belin. Rapport d'information. *Session Ordinaire*, 669, 2022. 2

[5] Christian Benz, Paul Debus, Huy Khanh Ha, and Volker Rodehorst. Crack segmentation on uas-based imagery using transfer learning. In *2019 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6, 2019. 3

[6] Christian Benz and Volker Rodehorst. Image-based detection of structural defects using hierarchical multi-scale attention. In *DAGM German Conference on Pattern Recognition (GCPR)*, pages 337–353. Springer, 2022. 3

[7] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97, 2009. Video-based Object and Event Analysis. 2

[8] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017. 3

[9] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018. 3

[10] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022. 7

[11] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[12] Aaron Defazio and Samy Jelassi. Adaptivity without compromise: a momentumized, adaptive, dual averaged gradient method for stochastic optimization. *The Journal of Machine Learning Research*, 23(1):6429–6462, 2022. 6

[13] Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *arXiv preprint arXiv:2303.11331*, 2023. 7

[14] Johannes Flotzinger, Philipp J. Rösch, and Thomas Braml. dacl10k: Benchmark for semantic bridge damage segmentation, 2023. 2, 4, 5, 6

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[16] Joakim Bruslund Haurum, Mingzhu Wang, Ajmal Mian, and Thomas B. Moeslund. Computer Vision for Civil and Infrastructure Engineering Workshop (CVCIE @ ECCV2022). https://vap.aau.dk/cvcie/. Accessed: 2023-11-15. 3

[17] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 7

[18] Philipp Hüthwohl, Ruodan Lu, and Ioannis Brilakis. Multi-classifier for reinforced concrete bridge defects. *Automation in Construction*, 105, Sep 2019. 3

[19] Philipp Hüthwohl and Ioannis Brilakis. Detecting healthy concrete surfaces. *Adv. Eng. Inf.*, 37:150–162, Aug. 2018. 3

[20] A. Kirillov, R. Girshick, K. He, and P. Dollar. Panoptic feature pyramid networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6392–6401, Los Alamitos, CA, USA, jun 2019. IEEE Computer Society. 3

[21] Alexander Kirillov, Kaiming He, Ross Girshick, and Piotr Dollár. A unified architecture for instance and semantic segmentation. In *CVPR*, 2017. 7

[22] Shreyas Kulkarni, Shreyas Singh, Dhananjay Balakrishnan, Siddharth Sharma, Saipraneeth Devunuri, and Sai Chowdeswara Rao Korlapati. Crackseg9k: A collection and benchmark for crack segmentation datasets and frameworks. In Leonid Karlinsky, Tomer Michaeli, and Ko Nishino, editors, *Computer Vision – ECCV 2022 Workshops*, pages 179–195, Cham, 2023. Springer Nature Switzerland. 3

[23] Shengyuan Li and Xuefeng Zhao. Image-Based Concrete Crack Detection Using Convolutional Neural Network and Exhaustive Search Technique. *Advances in Civil Engineering*, 2019(Ml), 2019. 3

[24] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 7

[25] 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*, pages 11976–11986, 2022. 7

[26] Bjoern H. Menze, Andras Jakab, Stefan Bauer, Jayashree Kalpathy-Cramer, Keyvan Farahani, Justin Kirby, Yuliya Burren, Nicole Porz, Johannes Slotboom, Roland Wiest, Levente Lanczi, Elizabeth Gerstner, Marc-André Weber, Tal Arbel, Brian B. Avants, Nicholas Ayache, Patricia Buendia, D. Louis Collins, Nicolas Cordier, Jason J. Corso, Antonio Criminisi, Tilak Das, Hervé Delingette, Çağatay Demiralp, Christopher R. Durst, Michel Dojat, Senan Doyle, Joana Festa, Florence Forbes, Ezequiel Geremia, Ben Glocker, Polina Golland, Xiaotao Guo, Andac Hamamci, Khan M. Iftekharuddin, Raj Jena, Nigel M. John, Ender Konukoglu, Danial Lashkari, José António Mariz, Raphael Meier, Sérgio Pereira, Doina Precup, Stephen J. Price, Tammy Riklin Raviv, Syed M. S. Reza, Michael Ryan, Duygu Sarikaya, Lawrence Schwartz, Hoo-Chang Shin, Jamie Shotton, Carlos A. Silva, Nuno Sousa, Nagesh K. Subbanna, Gabor Szekely, Thomas J. Taylor, Owen M. Thomas, Nicholas J. Tustison, Gozde Unal, Flor Vasseur, Max Wintermark, Dong Hye Ye, Liang Zhao, Binsheng Zhao, Darko Zikic, Marcel Prastawa, Mauricio Reyes, and Koen Van Leemput. The multimodal brain tumor image segmentation benchmark (brats). *IEEE Transactions on Medical Imaging*, 34(10):1993–2024, 2015. 2

[27] Guido Morgenthal, Norman Hallermann, Jens Kersten, Jakob Taraben, Paul Debus, Marcel Helmrich, and Volker Rodehorst. Framework for automated uas-based structural condition assessment of bridges. *Automation in Construction*, 97:77–95, 2019. 2

[28] Martin Mundt, Sagnik Majumder, Sreenivas Murali, Panagiotis Panetsos, and Visvanathan Ramesh. Meta-learning convolutional neural architectures for multi-target concrete defect classification with the concrete defect bridge image dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3

[29] Patrik F. Raudaschl, Paolo Zaffino, Gregory C. Sharp, Maria Francesca Spadea, Antong Chen, Benoit M. Dawant, Thomas Albrecht, Tobias Gass, Christoph Langguth, Marcel Lüthi, Florian Jung, Oliver Knapp, Stefan Wesarg, Richard Mannion-Haworth, Mike Bowes, Annaliese Ashman, Gwenael Guillard, Alan Brett, Graham Vincent, Mauricio Orbes-Arteaga, David Cárdenas-Peña, German Castellanos-Dominguez, Nava Aghdasi, Yangming Li, Angelique Berens, Kris Moe, Blake Hannaford, Rainer Schubert, and Karl D. Fritscher. Evaluation of segmentation methods on head and neck ct: Auto-segmentation challenge 2015. *Medical Physics*, 44(5):2020–2036, 2017. 2

[30] Timo Scharwächter, Markus Enzweiler, Uwe Franke, and Stefan Roth. Efficient multi-cue scene segmentation. In Joachim Weickert, Matthias Hein, and Bernt Schiele, editors, *Pattern Recognition*, pages 435–445, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. 2

[31] Amber L. Simpson, Michela Antonelli, Spyridon Bakas, Michel Bilello, Keyvan Farahani, Bram van Ginneken, Annette Kopp-Schneider, Bennett A. Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M. Summers, Patrick Bilic, Patrick F. Christ, Richard K. G. Do, Marc Gollub, Jennifer Golia-Pernicka, Stephan H. Heckers, William R. Jarnagin, Maureen K. McHugo, Sandy Napel, Eugene Vorontsov, Lena Maier-Hein, and M. Jorge Cardoso.

A large annotated medical image dataset for the development and evaluation of segmentation algorithms, 2019. 2

[32] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer. In *European conference on computer vision*, pages 459–479. Springer, 2022. 7

[33] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14408–14419, 2023. 7

[34] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers, 2021. 3, 6

[35] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2633–2642, 2020. 2

[36] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 7

6 Investigating Synthetic Data Extensions for Bridge Damage Segmentation

Publication in this chapter:
J. Flotzinger, F. Deuser, A. Jaziri, H. Neumann, N. Oswald, V. Ramesh, and T. Braml. synth-dacl: synth-dacl: Does Synthetic Defect Data Enhance Segmentation Accuracy and Robustness for Real-World Bridge Inspections?. doi: 10.48550/arXiv.2506.14255

SYNTH-DACL: DOES SYNTHETIC DEFECT DATA ENHANCE SEGMENTATION ACCURACY AND ROBUSTNESS FOR REAL-WORLD BRIDGE INSPECTIONS?

Johannes Flotzinger[1], Fabian Deuser[1], Achref Jaziri[2], Heiko Neumann[3], Norbert Oswald[1], Visvanathan Ramesh[2], Thomas Braml[1]

[1]University of the Bundeswehr Munich
[2]Goethe University Frankfurt am Main
[3]University of Ulm

ABSTRACT

Adequate bridge inspection is increasingly challenging in many countries due to growing ailing stocks, compounded with a lack of staff and financial resources. Automating the key task of visual bridge inspection – classification of defects and building components on pixel level – improves efficiency, increases accuracy and enhances safety in the inspection process and resulting building assessment. Models overtaking this task must cope with an assortment of real-world conditions. They must be robust to variations in image quality, as well as background texture, as defects often appear on surfaces of diverse texture and degree of weathering. dacl10k [1] is the largest and most diverse dataset for real-world concrete bridge inspections. However, the dataset exhibits class imbalance, which leads to notably poor model performance particularly when segmenting fine-grained classes such as cracks and cavities. This work introduces "*synth-dacl*", a compilation of three novel dataset extensions based on synthetic concrete textures. These extensions are designed to balance class distribution in dacl10k and enhance model performance, especially for crack and cavity segmentation. When incorporating the *synth-dacl* extensions, we observe substantial improvements in model robustness across 15 perturbed test sets. Notably, on the perturbed test set, a model trained on dacl10k combined with all synthetic extensions achieves a 2% increase in mean IoU, F1 score, Recall, and Precision compared to the same model trained solely on dacl10k.

***K*eywords** Automated Damage Recognition · Bridge Inspection · Synthetic Data · Semantic Segmentation

1 Introduction

Worldwide, many bridges are exposed to high traffic loads, extreme weather events, sea salt, and de-icing chemicals, leading to defects. At the same time, most industrialized countries face a growing stock of old infrastructure [2, 3]. To determine rehabilitation measurements and immediate actions, such as traffic restrictions or bridge closures, defects are identified and assessed during inspections. However, the current inspection process is often inefficient and prone to error [4, 5, 6], highlighting the significant potential of automated inspection methods to improve accuracy and reliability. Within an automated inspection workflow semantic segmentation plays a central role as it classifies, measures and localizes damage at pixel level [1, 7, 8].

The largest and most diverse dataset for the segmentation of bridge defects and components is dacl10k [1] due to its variety of buildings and classes. However, a major challenge of dacl10k is the strong class imbalance, affecting not only the number of images but also the pixel-level and instance-level distributions. *E.g.*, the dataset contains approximately ten times more images labeled with *Spalling* than with *Rockpocket*, and about ten times more pixels annotated as *Protective Equipment* than as *Crack*. This imbalance is evident in the dacl10k challenge [9], where participants consistently reported significantly poor performance for two concrete defects, *crack* and *cavity*. Addressing this imbalance is critical to improving model robustness, particularly for real-world bridge inspections, where models

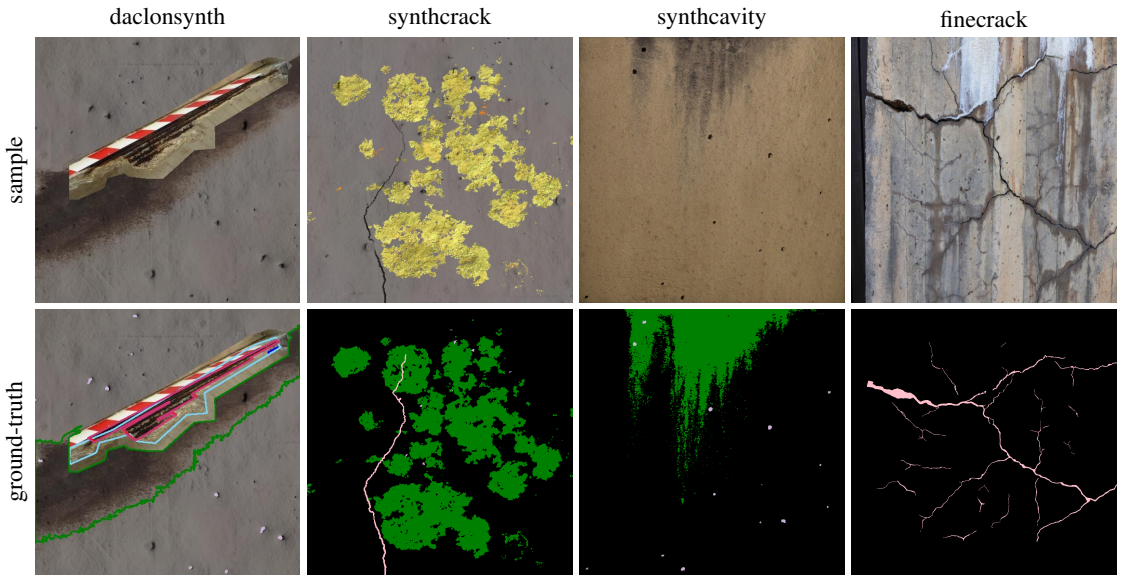


Figure 1: Left column/daclonsynth: Cropped defect shape from dacl10k pasted on synthetic concrete background and ground-truth below showing polygonal annotations of real-world (*Spalling* with corroded *Exposed Rebars*) and synthetic defects which are namely *Cavity* (pale purple) and *Weathering* (green); Middle-left Column/synthcrack: synthetic concrete surface with synthetic defects and ground-truth showing: *Crack* (rose), *Cavity*, *Weathering*; Middle-right column/synthcavity: synthcavity sample and ground-truth showing three combined masks, *Cavity* from both rendered concrete texture and the cavity generative model, and synthetic *Weathering*; Right column/finecrack: Test sample from dacl10k and fine-resolution crack masks.

must be resilient to variations in image quality, camera pose, concrete texture, and the degree of weathering. Real-world applications require models to perform reliably under varying conditions, but despite the wide range of image acquisition scenarios encountered in civil engineering, no study has rigorously investigated the robustness of multi-class or multi-label computer vision models for damage detection in this domain.

In our work, we address these challenges by introducing three synthetic dataset extensions, collectively referred to as "*synth-dacl*", each designed with 5,000 samples. The first extension addresses the issue of class imbalance. This enhancement superimposes real-world damage polygons on synthetic concrete backgrounds to maintain realistic defect shapes while balancing the representation of underrepresented defect classes. The second and third extension focus on improving model performance for challenging single defect types in practice. We simulate concrete surfaces with one primary synthetic defect – either a crack or a cavity – per set to directly target the detection of these classes. We systematically evaluate how synth-dacl extensions affect average performance, individual class performance, and overall model robustness. Robustness is assessed by applying 15 image perturbations to real-world test data from the dacl10k dataset, simulating conditions such as changes in illumination, noise, and contrast. This step ensures that our models are not only accurate, but also resilient to the unpredictable conditions found in real-world bridge inspections.

2 Related Work

2.1 Bridge Inspection Datasets

The S2DS dataset [7], which comprises 743 samples, is the first real-world semantic segmentation dataset for bridge inspection with pixel-wise labels for six classes relevant for concrete bridge inspections. In the field of binary crack segmentation both OmniCrack30k [10] and CrackSeg9k [11] are dataset collections for cracked and uncracked surfaces of various materials.

On the other hand, synthetic data has become widely used to enhance performance on real-world tasks, particularly when there is a shortage of well-labeled images [12]. For example, Dwibedi et al. [13] generate synthetic images by cutting and pasting object instances into diverse environments. Other studies explore the role of synthetic data in improving robustness in medical imaging. For instance, Al Khalil et al [14] examine the usability of synthesized short-axis Cardiac Magnetic Resonance (CMR) images generated using Generative Adversarial Networks to ameliorate the robustness of

heart cavity segmentation models across various conditions. A comprehensive review of such applications can be found in [15].

In civil engineering, synthetic data starts to become a resource to mitigate the limited availability of pixel-accurate annotated datasets. Much of the research in this field focuses on crack segmentation [16, 17, 18]. For instance, [16] developed a simulation model in Blender for generating additional synthetic crack data for concrete surfaces. The synthetic cracks in this dataset are modeled using irregular fractals. Another synthetic crack dataset is the Supervisely Synthetic Crack Segmentation dataset [18] which consists of 1,558 synthetic images for road surface crack detection. This dataset employs various generative algorithms, such as random walk, rapidly exploring random trees, and L-systems, to produce a diverse array of crack patterns. [17] introduced a synthetic dataset specifically for dam crack detection. This dataset integrates crack patterns extracted from real-world, open-source datasets with a 3D mesh model of an actual dam, creating realistic training data suited to the unique structural context of dams.

While these datasets have advanced research in the field of automated bridge inspections, each faces limitations when applied to practical scenarios. They typically focus on a single defect class (*Crack*), offer limited diversity in image quality, concrete texture, and environmental conditions, and lack lack multi-label annotations which is due to the overlapping character of concrete defects crucial. Moreover, existing studies on the robustness of semantic damage segmentation models [19, 20] are, again, restricted to binary crack detection and assess only a narrow set of perturbations, which fall short of representing the wide range of real-world challenges encountered during bridge inspections.

3 Datasets

In the following, detailed information on the investigated real-world dataset dacl10k and its finecrack masks (Section 3.1) as well as the synthetic extensions (Section 3.2) are provided.

3.1 dacl10k

dacl10k [1] is the first large-scale dataset for automated bridge inspections, containing 9,920 polygon-annotated images across 19 classes, grouped into defects and structural components. It features multi-label semantic segmentation with coarse pixel-level annotations, which stem from real bridge inspections. The dataset includes common defect combinations, as shown in the top-left tile of Figure 1. For the underlying work a new version v3 of this dataset [21] (referred to as dacl10k) was developed. During transition class ambiguities were resolved: *Spalling* and *Rockpocket* were often confused in v2 due to their visual similarity, even though they arise from different causes: corroding reinforcement versus poor concrete deaeration. The same applied for *Joint Tape* vs. *Restformwork*, and *Weathering* vs. *Wetspot*.

3.1.1 Finecrack Masks

Most open-source datasets for crack or defect segmentation [11, 22, 23, 24, 25, 26] focus solely on cracks on plain concrete surfaces. These datasets lack important real-world variations, such as wet cracks, efflorescence, graffiti, and weathered backgrounds. Consequently, models trained on these datasets [27, 28] tend to generate numerous false positives, as evidenced by the low precision scores in Table 2.

From a civil engineer's perspective, cracks are not necessarily severe. In concrete structures, they occur where the tensile strength of the concrete is exceeded. Depending on the exposure of the building part, cracks with a width up to 0.4 mm (0.016 in) on non-prestressed building parts may be irrelevant regarding structural integrity and durability. However, on pre-stressed bridges, a crack width of 0.1 mm (0.004 in) can indicate tendon failure and be critical [29, 30]. According to many inspection guidelines, the required measurement accuracy for crack width is 0.1 mm [31, 32, 33]. This emphasizes that, for practical use, the crack defect class requires pixel-accurate segmentation.

To address this issue, we created fine-resolution crack masks for the 496 dacl10k test images that contain *Crack* and *ACrack*. First, the polygon is cropped and contrast-enhanced. Then, it is segmented. For Crack instances, we apply Multi-Otsu thresholding to grayscale images. For *ACracks*, we use a pre-trained crack segmentation model [27] to generate approximations of fine crack masks. The results are fused into a binary mask and manually refined by a civil engineer to ensure pixel-level accuracy. Thus, the defect classes *Crack* and *ACrack* from dacl10k are fused within one binary *Crack* mask.

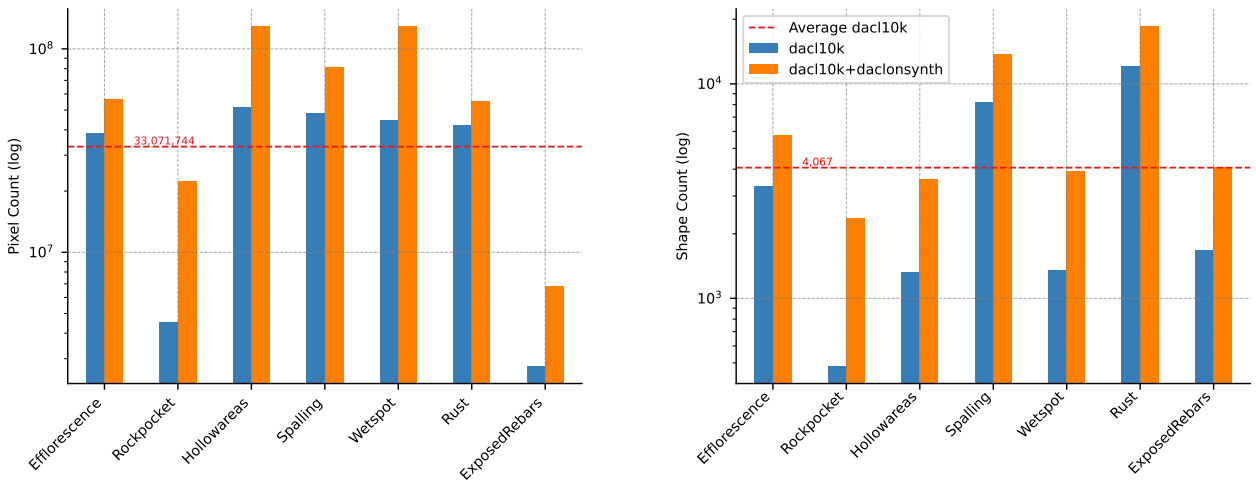


Figure 2: Comparison of classwise pixel count (left) and shape count (right) of dacl10k train set (blue) and the combination of dacl10k and daclonsynth (orange). The red dashed line marks the average pixel, or rather shape count, over the displayed classes from dacl10k. All stats based on the resized data (512×512).

3.2 Synthetic Dataset Extensions

Due to the significant class imbalance and low performance on specific classes in the dacl10k dataset, as well as the high costs associated with data collection and labeling, we explore synthetic data generation methods in the following sections [34]. Details on the class distribution in the original dacl10k-v3 dataset are provided in our supplementary material.

3.2.1 Synthetic Concrete Surfaces

To increase robustness in defect segmentation and overcome class imbalance, we introduce three new dataset extensions based on synthetic concrete surfaces: **daclonsynth**, **synthcrack** and **synthcavity**. To generate these surfaces an extended version of the physics-based rendering (PBR) introduced in Jaziri *et al.* [16] was used. The rendering pipeline consists of two main stages: (1) scene generation, and (2) defect injection. In the first stage, various texture maps are applied to produce diverse concrete surfaces using Blender's Cycles PBR engine. Optional overlays such as moss or dirt simulate *Weathering* effects (see Figure 1). In the second stage, up to two defects are added per scene (see Section 3.2.3 and 3.2.4), and semantic ground-truths including depth, surface normals, and class masks, are generated automatically, including a dedicated map for *Weathering*.

3.2.2 daclonsynth

Building on these synthetic scenes, we generate the **daclonsynth** extension with the specific goal of mitigating class imbalance in the dacl10k training set. As shown in Figure 2, several classes, particularly *Rockpocket*, *Exposed Rebars*, *Hollowareas*, and *Wetspot*, are heavily underrepresented in both pixel count and shape count. For instance, *Rockpocket* is annotated only 354 times and accounts for merely 4.5 million pixels, while the average across classes is around 45 million pixels.

To generate new training samples, we first filter the dacl10k training images to include only those containing at least one instance of a targeted underrepresented class. Then, each annotated shape is cropped from the image along with its corresponding polygonal annotation. This cropped region is then randomly rotated and pasted onto a synthetic concrete surface that is also randomly selected. For each class, half of the synthetic samples include *Weathering* overlays, while the other half remain clean to promote better generalization to both conditions.

In cases where the target defect typically co-occurs with a larger structural issue, as is common for *Exposed Rebars* appearing within *Spalling* or *Rockpocket*, the crop is extended to include the full area of the co-located host defect. Furthermore, to prevent models from exploiting the artificial distinction between real annotations and synthetic backgrounds, the shapes of *Spalling*, *Rockpocket*, *Wetspot*, *Hollowareas*, and *Efflorescence* are dilated using a 30×30 kernel prior to compositing.

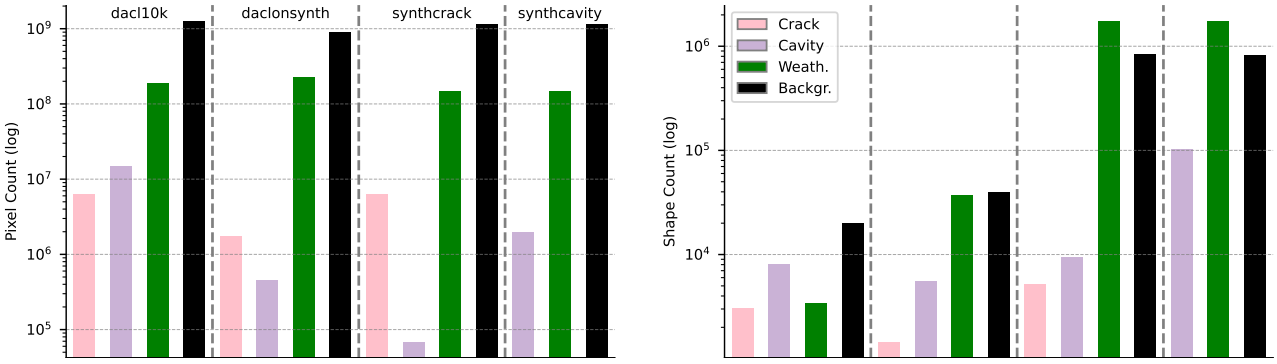


Figure 3: Pixel (left) and shape (right) counts for *Crack*, *Cavity*, *Weathering*, and background in dacl10k and synthetic datasets.

To determine the number of synthetic samples needed per class, we calculate the number of instances required to bring the pixel and shape counts closer to the averages of all underrepresented classes. We then use the mean of these two estimates to define a target sample count per class. The final distribution of the 5,000 synthetic samples in **daclonsynth** is determined proportionally based on these class-wise demands while accounting for the average number of pixels and shapes per sample for each defect. Although complete balance is not possible due to defect co-occurrence and overlaps, the resulting dataset significantly shifts the class distribution towards uniformity. Non-underrepresented classes may also be reproduced due to overlaps, which further contributes to a more diverse and realistic training dataset.

3.2.3 synthcrack

Crack patterns are generated using a fractal model based on [16] and rendered onto synthetic concrete surfaces with corresponding semantic masks. The resulting extension, **synthcrack** matches dacl10k in terms of crack pixel and shape counts. It also contains some incidental *Cavity* shapes, originating from fine-grained surface geometry embedded in the rendering pipeline, but these are not generated explicitly.

3.2.4 synthcavity

To generate cavities, we introduce a dedicated simulation approach based on Perlin noise. Multiple noise layers with varying octaves (2–32), persistence (0.6–0.9), and lacunarity (1.5 or 2) are combined to create irregular cavity maps. After thresholding and filtering out small regions, the resulting masks are used to build geometry-aware PBR textures. These are rendered and overlaid onto synthetic concrete surfaces to form the synthcavity dataset. Compared to dacl10k, synthcavity contains more cavity shapes, but with smaller area per instance, leading to a lower overall pixel count.

All synthetic datasets also contribute additional background and *Weathering* annotations. Overlaps in **daclonsynth** further introduce crack and cavity shapes, which are reflected in the distributions shown in Figure 3.

4 Experiments

In this section, we evaluate the effectiveness of our synthetic dataset extensions by testing eight differently trained models on the dacl10k test set. To assess the contribution of synthetic data to model robustness, the same models are also evaluated on a perturbed version of the test set. Additionally, we conduct two ablation studies: one focusing on the segmentation performance for finely annotated cracks, and another analyzing cross-domain generalization from synthetic to real-world data.

All experiments employ a Feature Pyramid Network (FPN) [35] with a MaxViT-Base Vision Transformer backbone [36]. The primary evaluation metric is Intersection over Union (IoU) as previous work [37, 38, 39] where IoU is set to 1 when the union is zero. This is complemented by F1 score, Precision, and Recall the analysis of the fine-grained classes and the perturbed testing. In general, the metrics are computed per class at the image level and averaged across the dataset. Mean values are reported by averaging the class-level scores. Further details, such as split size and class imbalance, can be found in the supplementary material.

Table 1: **mIoU and classwise IoU on the dacl10k test split.** A check mark in the Training Data column indicates the synthetic data used during training. Bold numbers indicate the highest value in each column, while underlined numbers represent the lowest.

Train data					classwise IoU																	
dacl10k	daclonsynth	synthcrack	synthcavity	**mIoU**	Crack	ACrack	Effloresc.	Rockpocket	WConccor	Hollowareas	Cavity	Spalling	Restformw.	Wetspot	Rust	Graffiti	Weathering	ExposedR.	Bearing	Eloint	Drainage	PEquipment	JTape
✓	–	–	–	42.71	31.40	49.81	42.97	21.26	9.79	54.88	20.07	44.83	31.69	23.52	47.03	66.09	**37.26**	39.89	59.16	56.44	60.17	73.61	41.64
✓	✓	–	–	42.93	**31.42**	**51.76**	**43.53**	**25.26**	10.13	56.07	22.02	45.19	31.68	24.22	48.53	66.01	27.11	40.61	59.47	55.12	**61.62**	73.00	42.85
✓	✓	✓	–	43.12	30.16	49.76	42.73	22.64	**14.25**	55.62	17.58	45.50	32.46	18.43	48.04	66.00	36.89	39.59	**60.98**	**59.96**	60.55	73.20	**44.97**
✓	✓	–	✓	42.85	30.62	47.05	42.58	22.84	13.01	56.41	20.95	45.68	32.28	22.91	48.20	63.63	34.82	40.39	58.44	56.22	61.26	74.23	42.58
✓	✓	✓	✓	**43.37**	29.48	50.55	43.25	21.11	11.99	**56.90**	17.89	45.65	**32.60**	23.87	**49.70**	65.65	35.87	**41.73**	60.92	58.56	60.73	74.10	43.51
✓	–	✓	–	42.76	27.25	48.55	43.23	22.47	11.14	55.18	23.14	**46.08**	31.34	24.62	48.45	66.18	32.17	40.96	60.50	54.86	59.47	**74.70**	42.14
✓	–	–	✓	42.88	29.18	47.93	43.15	22.86	11.49	55.02	**24.63**	45.44	31.74	**24.77**	48.57	**66.51**	33.26	40.03	60.28	55.77	58.58	74.00	41.46
✓	–	✓	✓	41.86	30.68	46.82	42.41	20.21	11.55	54.50	19.06	45.08	32.00	22.61	48.30	64.46	36.98	39.61	54.38	56.30	56.50	73.05	40.84

Table 2: **IoU, F1 Score, Recall and Precision on the finecrack and cavity masks.** Bold numbers indicate the highest value in each column, while underlined numbers represent the lowest. We compare with two open-source baseline methods for crack segmentation (two bottom rows). The train data are: (1) dacl10k, (2) daclonsynth, (3) synthcrack, (4) synthcavity.

Train data				Metrics Finecrack				Metrics Cavity			
(1)	(2)	(3)	(4)	IoU	F1	Rec.	Prec.	IoU	F1	Rec.	Prec.
✓	–	–	–	11.54	20.69	28.87	16.13	20.07	33.43	31.06	36.20
✓	✓	–	–	12.09	21.57	31.19	16.48	22.02	36.09	33.68	38.87
✓	✓	✓	–	**12.88**	**22.82**	26.64	19.96	17.58	29.91	22.34	45.23
✓	✓	–	✓	11.74	21.02	31.99	15.65	20.95	34.65	26.85	48.83
✓	✓	✓	✓	11.97	21.38	23.86	19.37	17.89	30.35	21.46	**51.82**
✓	–	✓	–	12.12	21.38	22.17	**21.11**	23.14	37.59	**36.11**	39.20
✓	–	–	✓	11.34	20.37	**34.11**	14.52	**24.63**	**39.53**	33.86	47.48
✓	–	✓	✓	12.33	21.95	26.50	18.73	19.06	32.02	23.20	51.66
HrSegNet-B64 [27]				3.34	6.47	19.26	3.89	–	–	–	–
CT-CrackSeg [28]				0.59	1.18	84.25	0.60	–	–	–	–

4.1 Results on dacl10k

As shown in Table 1, the inclusion of synthetic data improves overall performance, with the six highest mIoU results being achieved when synthetic data is used during training. At class level, extending dacl10k with the semi-synthetic daclonsynth improves IoU in average for the daclonsynth classes by 1.3%, with *Rockpocket* experiencing a notable +4% increase (see Table 1). This boost likely stems from the fivefold increase in representation of this class provided by daclonsynth. Notably, the top three configurations for crack IoU in Table 1 do not include synthcrack, which might seem contradictory at first. This outcome is explained by the fact that synthcrack employs much finer and more detailed crack annotations than those in dacl10k. As a result, the model learns to predict cracks at a finer resolution than what the dacl10k ground-truth can capture. This hypothesis is supported by the result of the model trained only on synthcrack, presented in Table 4, which confirms best performance on the finecrack masks. The results on the finecrack masks, and on *Cavity* are analyzed in the following Chapter 4.2.

4.2 Ablation: Results on Fine-Grained Classes

For both fine-grained classes, *Crack* in the form of finecrack masks and *Cavity*, we provide additional metrics in Table 2. All models incorporating synthcrack report better results on finecrack masks than the baseline trained on dacl10k only. The best finecrack IoU is achieved by the model utilizing dacl10k, daclonsynth and synthcrack, 1.3 percent points higher than dacl10k. Regarding *Cavity* the highest IoU (24.63%) and F1 score (39.53%) is reported for the model trained on the combination of dacl10k and synthcavity.

4.3 Results on perturbed dacl10k

In real-world bridge inspection applications, factors such as different camera models, varying image acquisition settings and environmental conditions introduce noise that negatively affect model performance. To investigate how this affects our differently trained models, we follow the methodology of Wang et al. [40] and apply 15 different perturbations to the test images. These include different noise functions, blur, brightness changes, and weather effects, which are demonstrated in the supplementary material. The results, averaged over all perturbations, are presented in Table 3.

Apart from the model trained on dacl10k, synthcrack and synthcavity (see Section 5) all models trained on both original and synthetically generated data consistently show the “most robust” to these perturbations. The highest IoU, F1 score,

Table 3: **Comparison of IoU, F1, Recall, and Precision on the test split between raw and perturbed (Pert.) images.** The *Change* column represents the relative performance difference between testing on raw images and perturbed images, highlighting the degradation in performance caused by the perturbations. Bold numbers indicate the highest value in each column, while underlined numbers represent the lowest. The train data are: (1) dacl10k, (2) daclonsynth, (3) synthcrack, (4) synthcavity.

Train data				**mIoU**			**mF1**			**mRecall**			**mPrecision**		
(1)	(2)	(3)	(4)	Raw	Pert.	Change	Raw	Pert.	Change	Raw	Pert.	Change	Raw	Pert.	Change
✓	–	–	–	42.71	30.87	-27.73	57.89	43.34	-25.13	56.02	38.63	-31.04	60.38	60.79	0.67
✓	✓	–	–	42.93	31.03	-27.72	58.12	43.50	-25.16	55.90	39.30	-29.69	62.18	61.56	-0.99
✓	✓	✓	–	43.12	30.89	-28.36	58.23	43.34	-25.57	55.42	38.76	-30.07	63.01	61.95	-1.69
✓	✓	–	✓	42.85	30.87	-27.95	58.16	43.51	-25.19	56.83	39.84	-29.90	60.96	60.61	-0.58
✓	✓	✓	✓	**43.37**	**32.16**	-25.84	**58.47**	**45.26**	-22.59	54.98	40.11	**-27.05**	**64.20**	**63.05**	-1.79
✓	–	✓	–	42.76	31.14	-27.18	57.98	43.84	-24.39	56.00	38.90	-30.52	61.35	61.88	0.86
✓	–	–	✓	42.88	32.00	**-25.36**	58.20	45.06	**-22.58**	**57.79**	**41.35**	-28.45	59.79	60.81	**1.70**
✓	–	✓	✓	41.86	29.96	-28.43	57.16	42.42	-25.80	55.59	38.07	-31.52	60.26	60.71	0.75

Table 4: **Synthetic-only training on three datasets; evaluated on real-world dacl10k test split and finecrack masks.**

Train data				**classwise IoU**									
daclonsynth	synthcrack	synthcavity	**mIoU**	Crack	Finecrack	Effloresc.	Rockpocket	Hollowareas	Cavity	Spalling	Wetspot	Rust	ExposedR.
✓	–	–	**27.42**	–	–	**29.54**	20.37	35.34	–	**31.81**	**3.29**	**37.15**	34.42
✓	✓	–	26.53	21.04	**13.11**	26.82	**21.77**	**38.34**	–	31.22	3.22	35.35	**34.47**
✓	✓	✓	22.59	**21.76**	12.64	27.72	14.03	34.85	0.00	34.71	3.09	34.14	32.99
–	✓	–	–	6.60	8.29	–	–	–	–	–	–	–	–
–	–	✓	–	–	–	–	–	–	0.15	–	–	–	–

and Precision is reported for the model trained on dacl10k combined with all *synth-dacl* extensions. Furthermore, this model shows 1.89% less relative performance loss in mean IoU and 2.54% in mean F1 score (see Table 3).

4.4 Ablation: Domain-Partitioned Evaluation

To evaluate cross-domain generalization from synthetic to real-world data, we perform a domain-partitioned ablation in which models are trained exclusively on synthetic datasets and evaluated on the dacl10k test split, thereby isolating the transferability of synthetic feature representations (see Table 4). Our findings indicate that, with the exception of synthcavity, the *synth-dacl* extensions are representative and consistent to generalize to real-world defect types. This is highlighted by the highest IoU on finecrack masks (13.11%) reported for the model trained on dacl10k and synthcrack. Utilizing synthcavity leads, according to Table 4, to 0% IoU on Cavity from dacl10k test set which is further discussed in Chapter 5.

4.5 Qualitative Evaluation

In some cases, the labels in the dacl10k dataset are overly coarse and not precise, which limits the effectiveness of quantitative evaluation metrics like IoU in reflecting the model's performance. Therefore, we complement our evaluation with qualitative results in Figure 4, providing a clearer assessment of the model's capability in handling the task. The predictions in the second row originate from the baseline trained on "dacl10k" (blue). The predictions in the "dacl10k+synth" row originate from the model trained on dacl10k plus the synthetic split that specifically includes the according class. In this row the three most left columns show predictions by the model trained on dacl10k+daclonsynth (green), followed by the models trained on dacl10k+synthcrack (pink) and dacl10k+synthcavity (turquoise) respectively. The bottom row displays predictions from the network that used all data (grey), or rather the most robust model according to Table 3.

The qualitative examples mostly underline the metrics reported in aforementioned Tables. The predictions on *Wetspot* by the most robust model show no false positives compared to the baseline dacl10k. On *Rockpocket* the prediction gets from top to bottom more accurate indicating that the additional synthetic data raises Precision. The *Crack* prediction proves that the segmentation gets narrower, thus closer to the crack edges but still leaves room for improvement. Regarding the *Cavity* sample, we observe that only the relevant cavities are marked, indicating higher accuracy.

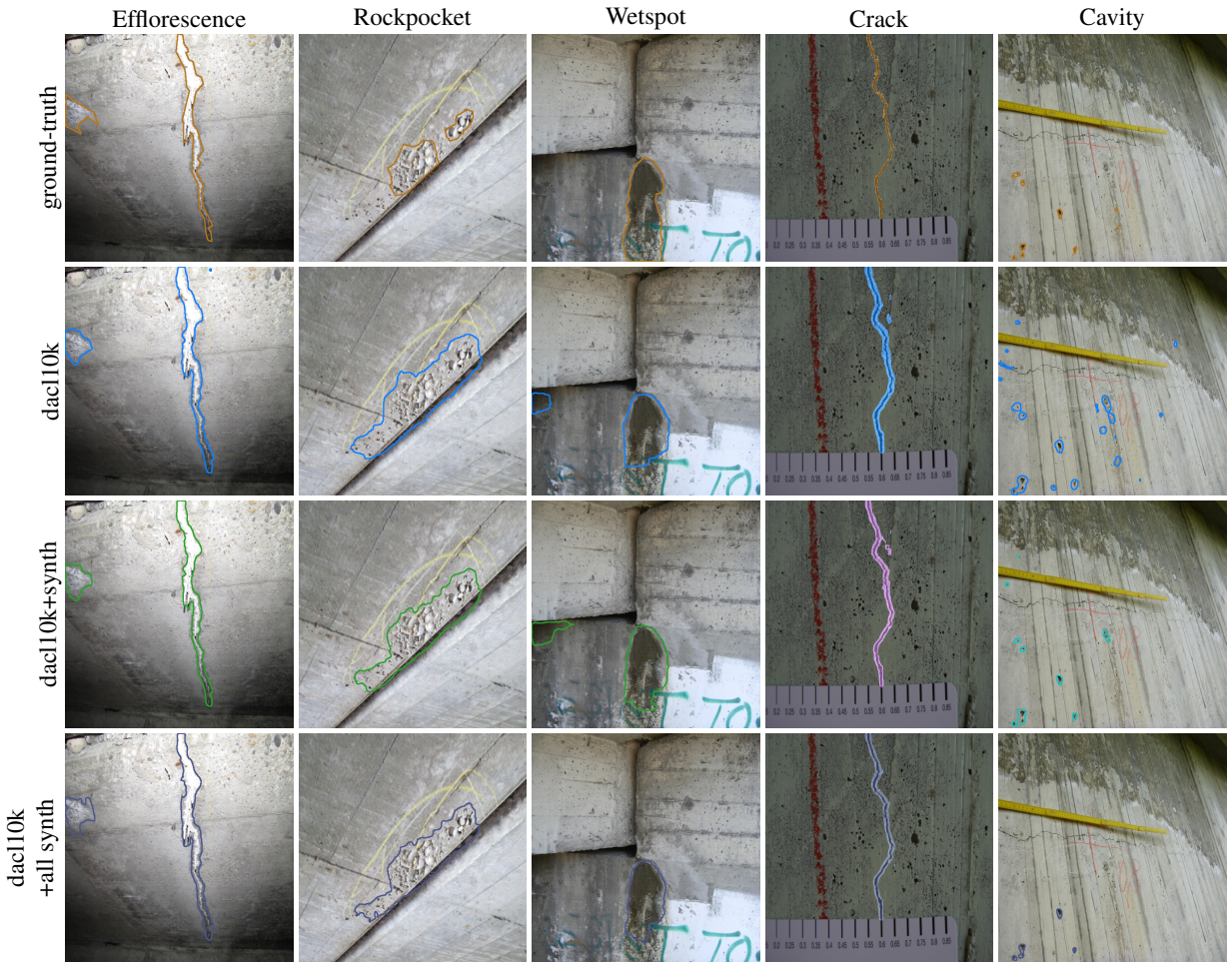


Figure 4: Qualitative results on dacl10k test samples for different damage classes (columns) across different training data setups (rows), where the row dacl10k+synth shows the predictions of the model achieving highest IoU on the given class.

5 Discussion

According to Table 1, introducing synthetic data specific to certain defects, such as *Cracks* (synthcrack) or *Cavities* (synthcavity), in isolation, the model performs better than without additional extensions. However, when both synthetic *Crack* and *Cavity* data are combined, it results in the lowest scores with respect to accuracy and robustness if trained without the balancing from daclonsynth (see Table 1 and 3). This suggests that while synthetic data can effectively address individual class imbalances, combining certain synthetic datasets can create conflicts that negatively impact model performance by introducing imbalances again. This is illustrated in Figure 2 and 3. *E.g.*, the number of pixels and shapes showing *Efflorescence* in dacl10k is 40 million pixels and 3,350 polygons, while synthcrack introduces 147 million pixels and 1.7 million polygons showing *Weathering*.

The best performance on *Weathering* in Table 1 is reported for the model trained exclusively on dacl10k, suggesting that synthetic *Weathering* features may not fully capture the characteristics of its real-world counterpart, which strongly rely on the subjacent concrete. Style transfer techniques may help to bridge this domain gap by enhancing the realism of synthetic textures [41].

Although the model's predictions on finecrack masks appear promising (Figure 4), the achieved IoU of 13% indicates substantial room for improvement in terms of practical applicability. A closer inspection of the predictions by the model trained only on synthcrack (see Supplementary Material) reveals that crack segments are often disrupted when objects such as shadows, Wetspots, or Efflorescence are located adjacent to the crack edges. To improve realism and robustness,

future work should focus on augmenting the synthetic pipeline with additional examples of such crack-bordering artifacts.

The isolated use of synthcavity proves insufficient for real-world Cavity detection, as shown in Table 2. Nonetheless, its combination with real-world data improves performance (Table 1), supporting its role as a supplementary rather than standalone training asset. Qualitative analysis of predictions made by the model trained only on synthcavity (see Supplementary Material) reveals common false positives arising, like in the case of synthcrack, from visually alien elements such as bolts, soil patches, or drainage components. Moreover, while the model successfully identifies small cavities, it consistently misses large ones – especially those with sharp, irregular geometries. This limitation stems from the underrepresentation such *Cavity* features in the synthcavity dataset. In order to generate more realistic synthetic *Cavity* images, future work should focus on incorporating methods such as texture-sensitive preprocessing [42], style transfer methods [43, 44] or domain randomization strategies [45].

6 Conclusion

In this work, we introduce *synth-dacl*, a new set of three synthetic dataset extensions designed to support automated damage recognition in real-world bridge inspections. Alongside these extensions, we provide a cleaned version of the dacl10k dataset and introduce fine-resolution crack masks for the test set, enabling more precise evaluation. Through a series of experiments, we analyze the impact of synthetic data on segmentation performance and model robustness by combining the original dacl10k dataset with our synthetic extensions.

Our findings show that synthetic data can significantly improve performance, though its effectiveness depends on how it is integrated. Training exclusively on the *synthcrack* dataset yields the best performance on fine-resolution crack prediction, as demonstrated in Table 2. We also found that the *synthcavity* extension improves accuracy only when combined with real-world data, particularly in the dacl10k+*synthcavity* setup. Using all three synthetic extensions (*daclonsynth*, *synthcrack*, and *synthcavity*) leads to the most robust models, especially under challenging conditions.

Beyond these empirical results, our study reveals an important insight: although synthetic data is a powerful tool for augmenting training sets and improving model performance, it is not always ready for seamless integration into real-world applications. Further research is necessary to develop methods for generating synthetic data that more accurately capture the visual complexity, texture diversity, and environmental variability present in real-world bridge inspection scenarios. Our work lays a foundation for this future exploration of synthetic data in structural inspection tasks and the development of robust damage recognition systems.

Acknowledgments

We thank the institute for Distributed Intelligent Systems at the University of the Bundeswehr Munich for providing computing time at MonacumOne Cluster.

Supplementary Material

1 Data Availablility

We make all data used in the underlying research openly accessible:

- The updated version of dacl10k-v3 [21].
- The synthetic dataset extensions [34].

2 Implementation Details

We utilize a Feature Pyramid Network (FPN) [46] architecture combined with an ImageNet pretrained MaxViT-base backbone [47]. Training and testing on the dacl10k dataset are performed at a resolution of 512×512. To ensure consistency with previous work [1], we employ a weighted loss function that optimizes two objectives: semantic segmentation and classification loss. The mask loss is a combination of Binary Cross-Entropy (BCE) and Dice Loss. In addition, for the classification task, we use Binary Cross-Entropy to assure that the model does not ignore smaller classes present in the images. This is critical for dealing with imbalances where small classes might otherwise not be predicted. The auxiliary loss is added with a weight of 0.1. During training, we set the learning rate to $1\mathrm{e}{-5}$ and use a cosine learning rate scheduler to gradually reduce the learning rate over 30 epochs. To improve the robustness of the model, we apply basic augmentations during training, specifically random rotations and horizontal/vertical flipping. These augmentations help the model to better generalize to different perspectives and orientations in the dataset.

3 Datasets

Additional statistics and details about the dacl10k-v3 dataset and the synth-dacl extensions are shown below. Table 5 shows the number of images present in dacl10k and the synthetic extensions.

Table 5: Image count and count of images showing Weathering for dacl10k and the synth-dacl extensions.

	Images	Weathered
dacl10k	6,935	3,449
daclonsynth	5,000	2,500
synthcrack	5,000	2,500
synthcavity	5,000	2,500

3.1 dacl10k-v3

The following paragraphs, especially the statistics on classes that are in the main focus of synth-dacl are analyzed, subdivided into the three synthetic extensions (see Table 6).

daclonsynth. *Efflorescence* and *Spalling* emerge as dominant classes in terms of pixel coverage, each contributing over 1 billion pixels (2.59% and 2.61% of the dataset, respectively). This is complemented by a significant number of shapes (5,022 for *Efflorescence* and 12,272 for *Spalling*). Rust has the highest number of shapes in this group (17,679 shapes, 16.31% of total shapes). However, the average size per shape (46,655 pixels) indicates that Rust's polygons are predominantly small. *Wetspot* and *Hollowarea* are moderately represented compared to other classes in synth-dacl, with pixel counts of 537M (1.33%) and 789M (1.95%). The average *Wetspot* contains 271,818 pixels per shape. *Hollowareas* are half the size with 415,234 pixels per shape. *Rockpocket* and *ExposedRebars* are underrepresented, with 60M pixels each (0.15% of total dataset pixels). Despite their lower prevalence, these classes maintain moderate shape complexity, with average pixels per shape of 80,785 and 25,818, respectively.

Crack. The class *Crack* is characterized by a pixel count of 120.7M pixels, which is 0.30% of the dataset, and a moderate number of shapes (4,446, 4.10%). On average, *Crack* polygons have a size of 27,141 pixels/shape.

Cavity. The *Cavity* class has a high number of shapes (11,661, 10.76% of all shapes), making it the most shape-dense class in synth-dacl. However, these shapes are remarkably small, with an average size of 13,458 pixels. Despite the modest total number of pixels (156.9M, 0.39%), the high frequency of *Cavity* shapes (6.77% shapes/image) suggests

Table 6: Overall statistics of the dataset at original resolution regarding pixel count (#pixels), shape/polygon count (#shape), image count (#images), average number of shapes per image (#shape/image), number of pixels per shape (#pixels/shape), number of pixels per image (#pixels/image), share of shapes (%shape) and share of pixels (%pixels). Midrules separate the classes according to their group affiliation.

Class	#pixels	#shape	#images	#shape/image	#pixels/shape	#pixels/image	%shape	%pixels
Crack	120,670,614	4,446	2,464	1.80	27,141	48,973	4.10	0.30
ACrack	497,199,653	523	466	1.12	950,669	1,066,952	0.48	1.23
Efflorescence	1,045,865,294	5,022	2,181	2.30	208,257	479,535	4.63	2.59
Rockpocket	60,669,651	751	446	1.68	80,785	136,031	0.69	0.15
WConccor	225,428,183	492	366	1.34	458,187	615,924	0.45	0.56
Hollowareas	788,944,545	1,900	1,564	1.21	415,234	504,440	1.75	1.95
Cavity	156,930,825	11,661	1,722	6.77	13,458	91,133	10.76	0.39
Spalling	1,055,540,626	12,272	4,705	2.61	86,012	224,344	11.32	2.61
Restformwork	59,095,976	1,282	1,064	1.20	46,097	55,541	1.18	0.15
Wetspot	536,569,266	1,974	1,343	1.47	271,818	399,530	1.82	1.33
Rust	824,807,051	17,679	4,883	3.62	46,655	168,914	16.31	2.04
Graffiti	470,487,206	2,705	1,171	2.31	173,932	401,782	2.50	1.16
Weathering	2,865,923,796	5,021	3,449	1.46	570,787	830,943	4.63	7.09
ExposedRebars	64,287,097	2,490	1,109	2.25	25,818	57,969	2.30	0.16
Bearing	638,064,849	1,506	1,038	1.45	423,682	614,706	1.39	1.58
EJoint	422,636,952	602	538	1.12	702,055	785,571	0.56	1.05
Drainage	466,415,500	2,016	1,470	1.37	231,357	317,289	1.86	1.15
PEquipment	1,482,991,703	2,314	1,888	1.23	640,878	785,483	2.14	3.67
JTape	64,621,780	1,269	1,092	1.16	50,923	59,177	1.17	0.16
Background	30,288,963,831	32,448	9,852	3.29	933,462	3,074,397	29.94	74.91

that when *Cavity* is present, the entire concrete surface is usually affected. Cavities are residual air voids on the concrete surface. They are usually a "cosmetic" defect and do not indicate any deficiencies in structural integrity.

For dacl10k-v3 we use the same partitioning as for dacl10-v2, which is based on K-means clustering (20 clusters), with proportional sampling across training (70%), validation (10%), test-dev (10%), and test-priv (10%) splits.

3.2 Synthetic Addons

3.2.1 daclonsynth

A detailed discussion of the daclonsynth dataset follows in this section. Figure 5 displays the change of pixels, shapes, and images by adding dalonsynth to dacl10k, in percentage. The most significant changes occur in classes such as Exposed Rebars, Hollowarea, *Wetspot*, and *Rockpocket* with large increases in pixel, shape, and image counts. For example, the Rocketpocket class sees a striking jump, especially in image count change, reaching nearly 400% (see Figure 5). Although they are not in the focus of daclonsynth, defects like *Crack*, *ACrack* or *Cavity* are also added. This is due to the overlapping of defects as mentioned in the main article. For most of these "unintentional" defects, a gain in pixel and shape count of about 50% can be observed.

Figure 6 presents the change in pixel, shape and image distribution at the class level.

Comment on daclonsynth class selection. Concrete defects such as *Crack*, *ACrack* and *Cavity* are not part of daclonsynth because they are considered in synthcrack and synthcavity. Annotations of Washouts/Concrete corrosion (WConccor) are ambiguous and need to be cleaned by a domain expert and are therefore neglected in this work. Restformwork would have required synthesized joints between concrete components and the exact fitting of polystyrene panels into the corresponding joint. Due to this complexity, this class was not taken into account. Furthermore, the bridge components Bearing, *Expansion Joint* (EJoint), Drainage, *Protective Equipment* (PEquipment), JTape (JTape) are not in the scope of daclonsynth, too. Objects of this group do not appear on concrete. For instance, *Protective Equipment*, such as railings on the side of the bridge, can be present in front of sky, forest, or water. Similarly, components like *Expansion Joint* and joint tape are usually photographed from a top-down perspective, situated between asphalt surfaces on bridge decks. Moreover, these classes were excluded because the model already performs exceptionally well on them compared to other categories.

4 Perturbations

In Figure 7 all 15 perturbations are demonstrated using one example drawn from dacl10k. The sample displays *Spalling*, *Exposed Reinforcement Bars* (ExposedRebars) that are corroded (*Rust*). Figure 8 shows the IoU drop in percent of the "most robust" model that achieved the best overall performance on the perturbed data. It can be observed that among all classes, the concrete defects *Cavity*, *Rockpocket*, and *WConccorr* are the most affected, compared to others that are

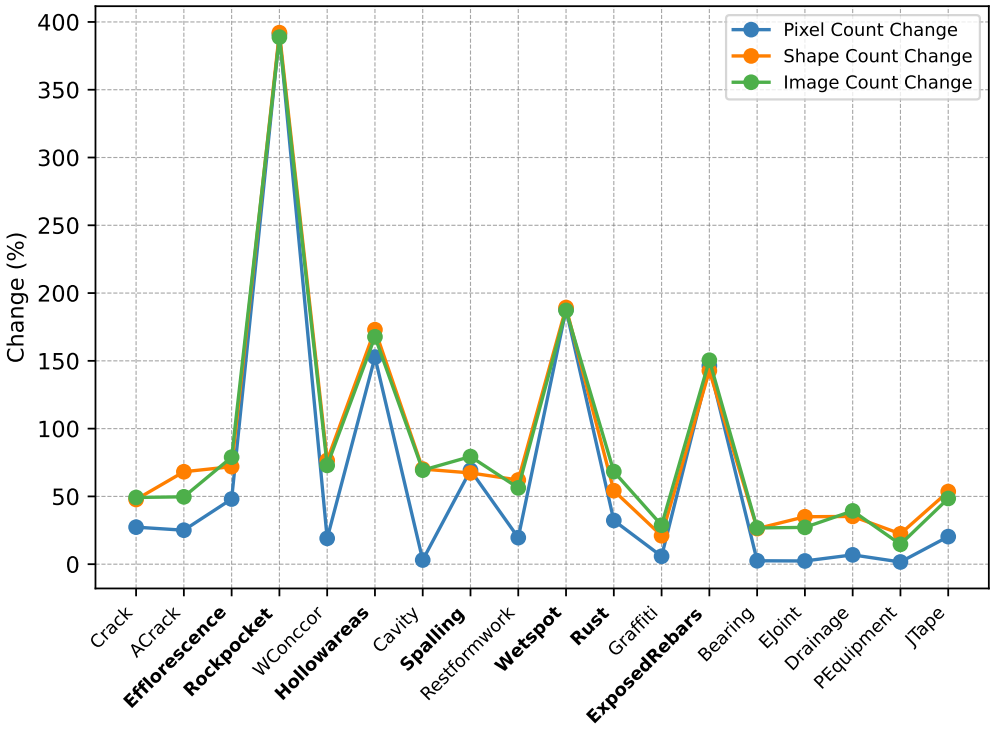


Figure 5: Distribution shift on class level regarding pixel, shape and image count. Classes considered in daclonsynth are in bold.

usually sensitive to only one or two image defects. The most resistant class is Graffiti and the building components such as *Protective Equipment* (PEquipment), *Drainage* or *Expansion Joint* (EJoint). In terms of shape and texture, the components are more unique, as opposed to concrete defects, which often have a very similar visual appearance. Regarding *Crack*, the Elastic Transform significantly disturbs the model, likely due to the distortion of the distinct zigzag pattern and sharp corners characteristic of *Cracks*.

In summary, the listed perturbations strongly affect the model performance (some classes by 100%), compared to other models, *e.g.*, SAM, as analyzed in Wang *et al.* [40], who report a maximum IoU drop of about 30% on the Road Extraction dataset [48].

5 Predictions Domain-Partitioned Evaluation

Figure 10 and Figure 9 present the predictions and ground-truth annotations for models trained exclusively on their respective synthetic extensions. The predictions produced by the synthcrack model demonstrate good performance on cracks that are relatively straight and unbranched. However, the model's accuracy decreases when cracks exhibit complex branching or when wet spots or efflorescence intersect the crack edges, often leading to incomplete or interrupted predictions.

For the synthcavity model, fine-grained and small cavities are partially predicted. In contrast, large cavities with sharp boundaries are often not reliably detected, indicating limitations in generalizing to more geometrically distinct damage patterns.

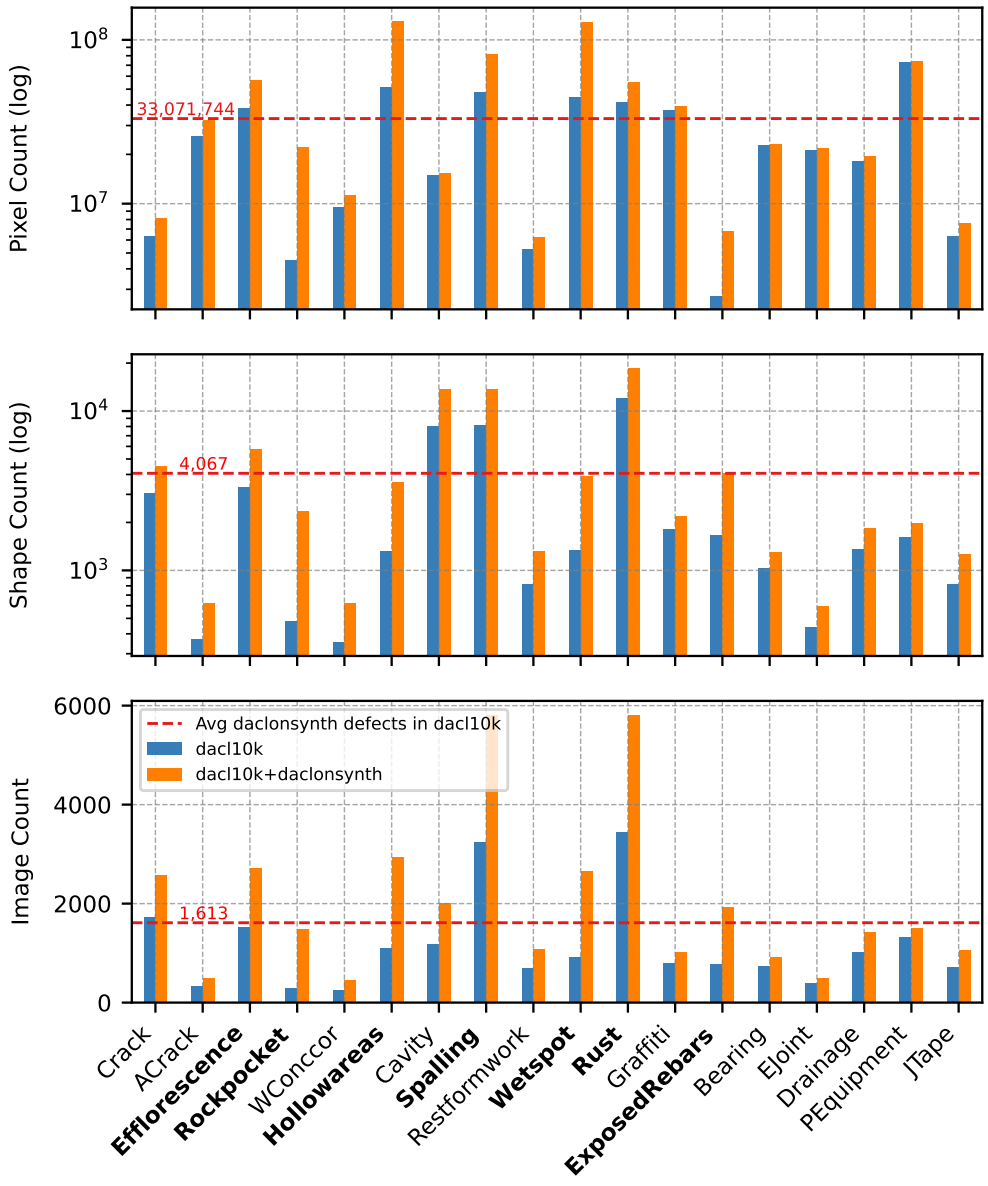


Figure 6: Comparison of classwise pixel count (top) and shape count (bottom) of dacl10k (blue) and the combination of dacl10k and daclonsynth (orange) dataset. The red dashed horizontal line marks the average pixel, or rather shape count, of daclonsynth classes (bold) which was aimed to reach through adding daclonsynth.

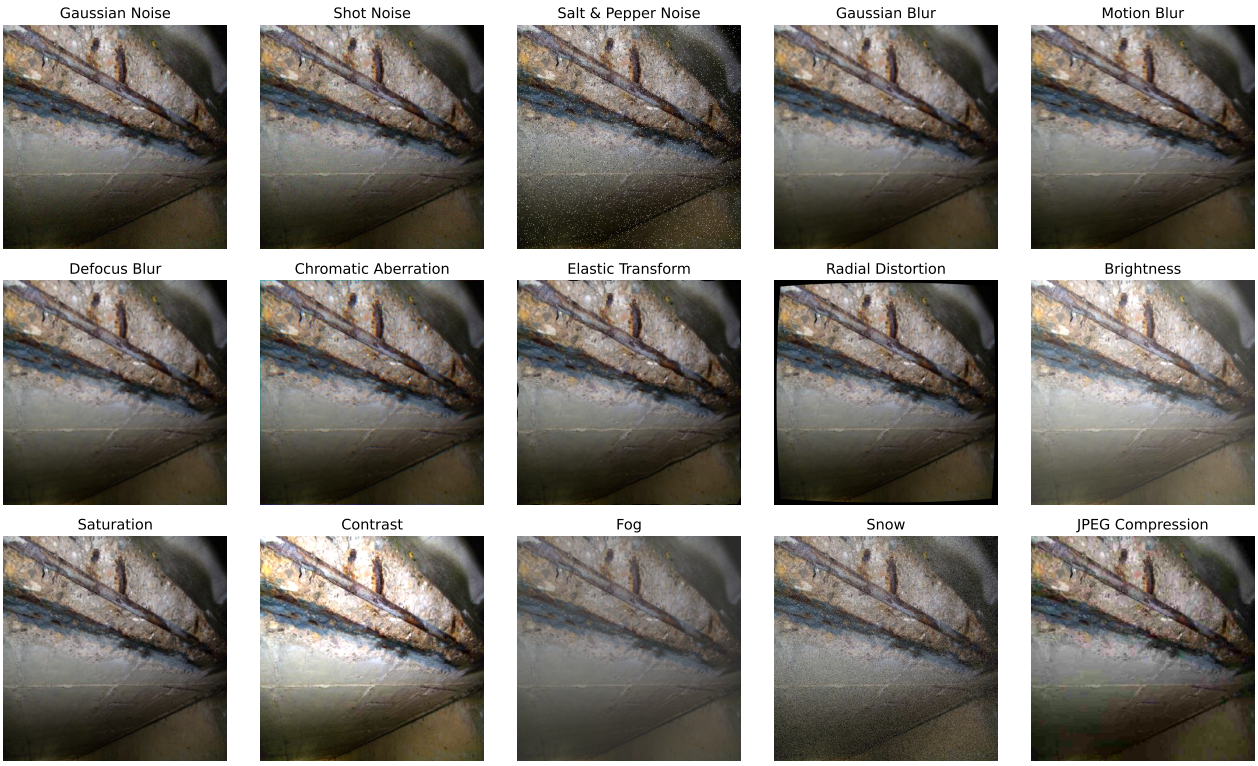


Figure 7: Illustration of the fifteen perturbation types utilized in experiments chapter, following [40].

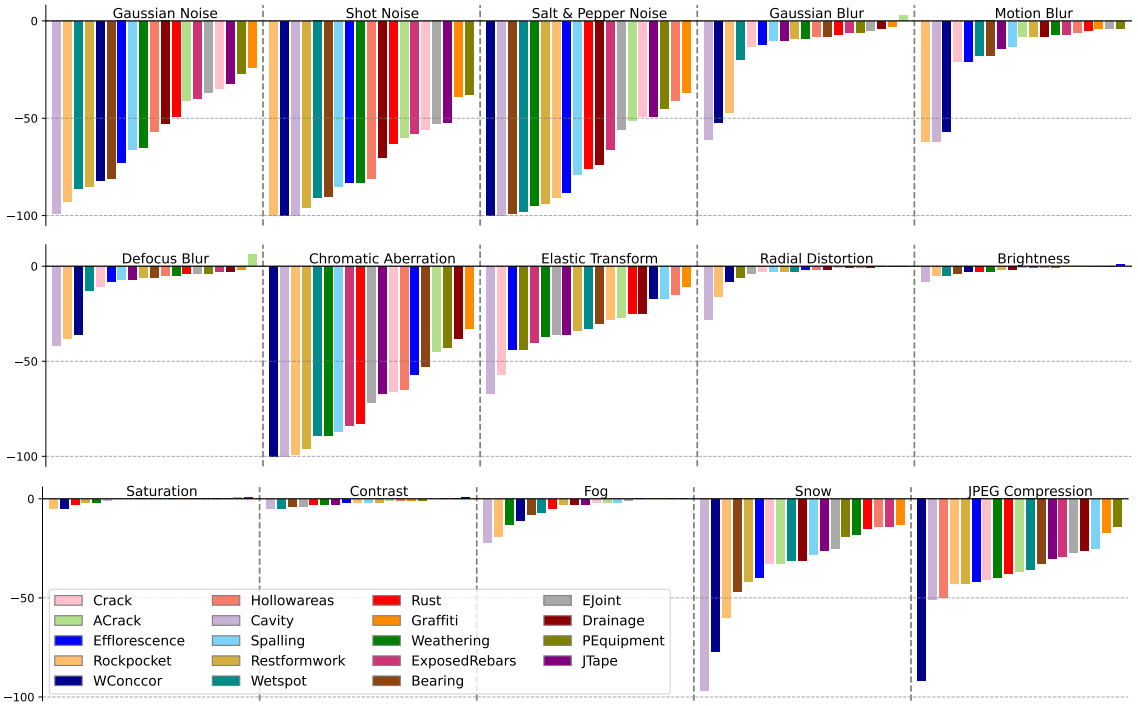


Figure 8: IoU drop in percent of the most robust model, trained on dacl10k and all synthetic extensions, for each perturbation and class in the dacl10k test set.



Figure 9: Rated (column header) predictions of the model trained on synthcrack only.

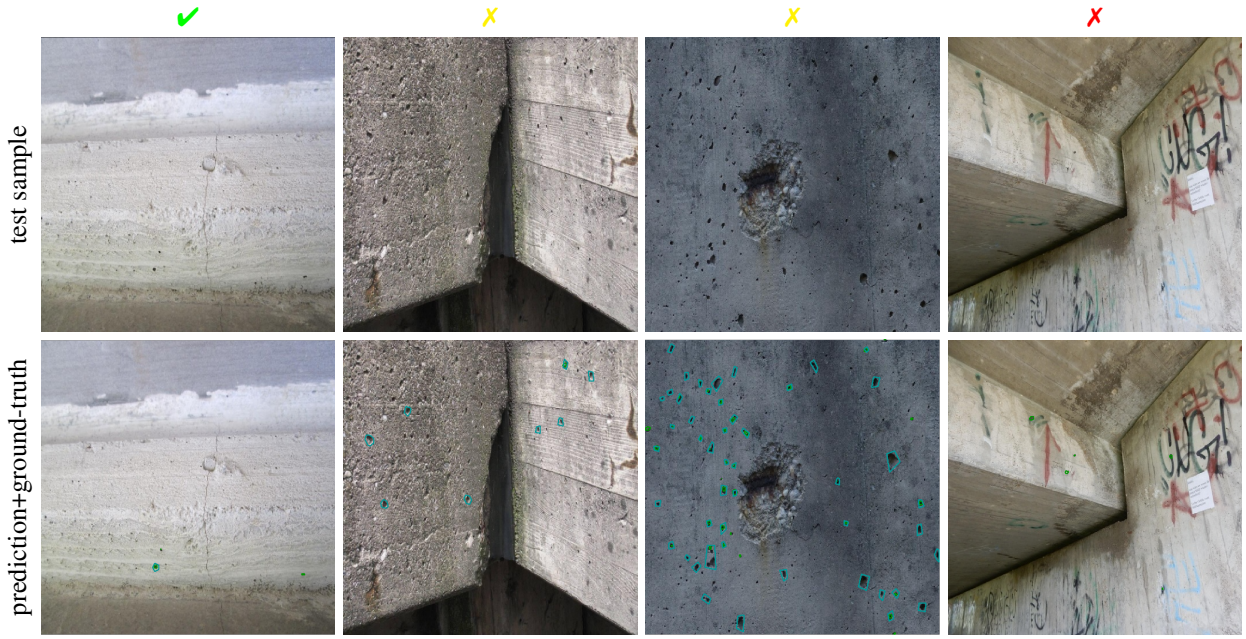


Figure 10: Rated predictions (column header) of the model trained on synthcavity only. Predictions are green and ground-truth is blue.

References

[1] Johannes Flotzinger, Philipp J. Rösch, and Thomas Braml. dacl10k: Benchmark for semantic bridge damage segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 8626–8635, January 2024.

[2] American Road & Transportation Builders Association (ARTBA). ARTBA bridge report, 2024. Accessed: 2024-10-17.

[3] Federal Highway Research Institute (BaSt). Brückenstatistik. `https://www.bast.de/DE/Statistik/Bruecken/Brueckenstatistik.pdf`, 5 2025. [Accessed 15-05-2025].

[4] Brent M. Phares, Glenn A. Washer, Dennis D. Rolander, Benjamin A. Graybeal, and Mark Moore. Routine highway bridge inspection condition documentation accuracy and reliability. *Journal of Bridge Engineering*, 9(4):403–413, July 2004.

[5] Pengkun Liu, Ying Shi, Ruoxin Xiong, and Pingbo Tang. Quantifying the reliability of defects located by bridge inspectors through human observation behavioral analysis. *Developments in the Built Environment*, 14:100167, 2023.

[6] Abdelrahman M. Abdallah. *A Study on Bridge Inspections: Identifying Barriers to New Practices and Providing Strategies for Change*. Phd dissertation, Colorado State University, 2021.

[7] Christian Benz and Volker Rodehorst. Image-based detection of structural defects using hierarchical multi-scale attention. In *DAGM German Conference on Pattern Recognition (GCPR)*, pages 337–353. Springer, 2022.

[8] Zengsheng He, Cheng Su, and Yichuan Deng. A novel mo-yolov4 for segmentation of multi-class bridge damages. *Advanced Engineering Informatics*, 62:102586, 2024.

[9] Johannes Flotzinger, Philipp J. Rösch, Christian Benz, Muneer Ahmad, Murat Cankaya, Helmut Mayer, Volker Rodehorst, Norbert Oswald, and Thomas Braml. Dacl-challenge: Semantic segmentation during visual bridge inspections. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, pages 716–725, January 2024.

[10] Christian Benz and Volker Rodehorst. Omnicrack30k: A benchmark for crack segmentation and the reasonable effectiveness of transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3876–3886, June 2024.

[11] Shreyas Kulkarni, Shreyas Singh, Dhananjay Balakrishnan, Siddharth Sharma, Saipraneeth Devunuri, and Sai Chowdeswara Rao Korlapati. Crackseg9k: a collection and benchmark for crack segmentation datasets and frameworks. In *European Conference on Computer Vision*, pages 179–195. Springer, 2022.

[12] Florentin Poucin, Andrea Kraus, and Martin Simon. Boosting instance segmentation with synthetic data: A study to overcome the limits of real world data sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 945–953, 2021.

[13] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1301–1310, 2017.

[14] Yasmina Al Khalil, Sina Amirrajab, Cristian Lorenz, Jürgen Weese, Josien Pluim, and Marcel Breeuwer. On the usability of synthetic data for improving the robustness of deep learning-based segmentation of cardiac magnetic resonance images. *Medical Image Analysis*, 84:102688, 2023.

[15] Keith Man and Javaan Chahl. A review of synthetic image data and its use in computer vision. *Journal of Imaging*, 8(11):310, 2022.

[16] Achref Jaziri, Martin Mundt, Andres Fernandez, and Visvanathan Ramesh. Designing a hybrid neural system to learn real-world crack segmentation from fractal-based simulation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 8636–8646, January 2024.

[17] Jia Xu, Cheng Yuan, Jiaxuan Gu, Jian Liu, Jiong An, and Qingzhao Kong. Innovative synthetic data augmentation for dam crack detection, segmentation, and quantification. *Structural Health Monitoring*, 22(4):2402–2426, 2023.

[18] Supervisely. Supervisely synthetic crack segmentation, 2023.

[19] Elyas Asadi Shamsabadi, Chang Xu, and Daniel Dias da Costa. Robust crack detection in masonry structures with transformers. *Measurement*, 200:111590, 2022.

[20] Donghan Lee, Jeongho Kim, and Daewoo Lee. Robust Concrete Crack Detection Using Deep Learning-Based Semantic Segmentation. *International Journal of Aeronautical and Space Sciences*, 20(1):287–299, March 2019.

[21] Johannes Flotzinger, Philipp J. Rösch, and Thomas Braml. dacl10k, 2025.

[22] Hongyan Xu, Xiu Su, Yi Wang, Huaiyu Cai, Kerang Cui, and Xiaodong Chen. Automatic bridge crack detection using a convolutional neural network. *Applied Sciences*, 9(14), 2019.

[23] Sattar Dorafshan, Robert J. Thomas, and Marc Maguire. SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks. *Data in Brief*, 21:1664–1668, December 2018.

[24] Qin Zou, Yu Cao, Qingquan Li, Qingzhou Mao, and Song Wang. Cracktree: Automatic crack detection from pavement images. *Pattern Recognition Letters*, 33(3):227–238, 2012.

[25] Myeongsuk Pak and Sanghoon Kim. Crack detection using fully convolutional network in wall-climbing robot. In James J. Park, Simon James Fong, Yi Pan, and Yunsick Sung, editors, *Advances in Computer Science and Ubiquitous Computing*, pages 267–272, Singapore, 2021. Springer Singapore.

[26] Yahui Liu, Jian Yao, Xiaohu Lu, Renping Xie, and Li Li. Deepcrack: A deep hierarchical feature learning architecture for crack segmentation. *Neurocomputing*, 338:139–153, 2019.

[27] Yongshang Li, Ronggui Ma, Han Liu, and Gaoli Cheng. Real-time high-resolution neural network with semantic guidance for crack segmentation. *Automation in Construction*, 156:105112, December 2023.

[28] Huaqi Tao, Bingxi Liu, Jinqiang Cui, and Hong Zhang. A convolutional-transformer network for crack segmentation with boundary awareness. In *2023 IEEE International Conference on Image Processing (ICIP)*, pages 86–90, 2023.

[29] German Institute for Standardization Registered Association (DIN). Eurocode 2: Design of reinforced and prestressed concrete structures - Part 1-1: General design rules and rules for buildings. Standard, DIN German Institute for Standardization, Berlin, GER, April 2013.

[30] ACI Committee 224. *Control of cracking in concrete structures*. American Concrete Institute, Farmington Hills, Mich., 2001.

[31] German Institute for Standardization Registered Association (DIN). Engineering structures in connection with roads - inspection and test. Standard DIN 1076:1999-11, DIN German Institute for Standardization, Berlin, GER, November 1999.

[32] Federal Highway Research Institute (BaSt). Guideline for the uniform acquisition, assessment, recording and evaluation of results of structural inspections (RI-EBW-PRÜF), February 2017.

[33] ACI Committee 201. 201.1r-08: Guide for conducting a visual inspection of concrete in service. *Technical Documents*, 2008.

[34] Johannes Flotzinger, Fabian Deuser, and Achref Jaziri. synth-dacl: Synthetic Extensions for dacl10k Dataset, 2025.

[35] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[36] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 459–479, Cham, 2022. Springer Nature Switzerland.

[37] Mark Everingham, S. M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111:98 – 136, 2014.

[38] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding . In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, Los Alamitos, CA, USA, June 2016. IEEE Computer Society.

[39] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5122–5130, 2017.

[40] Yuqing Wang, Yun Zhao, and Linda Petzold. An empirical study on the robustness of the segment anything model (sam). *Pattern Recognition*, 155:110685, 2024.

[41] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer, 2017.

[42] Timm Hess, Martin Mundt, Iuliia Pliushch, and Visvanathan Ramesh. A procedural world generation framework for systematic evaluation of continual learning, 2021.

[43] Taesung Park, Alexei A. Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive learning for unpaired image-to-image translation, 2020.

[44] Viraj Prabhu, David Acuna, Andrew Liao, Rafid Mahmood, Marc T. Law, Judy Hoffman, Sanja Fidler, and James Lucas. Bridging the sim2real gap with care: Supervised detection adaptation with conditional alignment and reweighting, 2023.

[45] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world, 2017.

[46] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6399–6408, 2019.

[47] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxvit: Multi-axis vision transformer. In *European conference on computer vision*, pages 459–479. Springer, 2022.

[48] Ilke Demir, Krzysztof Koperski, David Lindenbaum, Guan Pang, Jing Huang, Saikat Basu, Forest Hughes, Devis Tuia, and Ramesh Raskar. Deepglobe 2018: A challenge to parse the earth through satellite images. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 172–17209, 2018.

7 Conclusion

7.1 Summary

Each of the three main chapters tackles the problem of *AIDR* for automated bridge inspections based on sequential research questions while utilizing data of different origin: open-source, real-world, and synthetic.

Chapter 4 confronts two key research questions: How to develop strong baseline models trained on existing data for concrete defect classification, and how do these models generalize to real-world data. Together, these questions aim to establish foundational baseline models and assess current dataset limitations in practical scenarios. The first contribution, *bikit* [63], introduced a detailed transfer learning and hyperparameter optimization, establishing strong baselines on the two leading datasets of the time. By bundling datasets, pretrained models, and evaluation metrics into an accessible toolkit, research in *AIDR* was significantly accelerated. The second work [128] in this chapter provides a high-level introduction to automated damage recognition for a civil engineering audience and an in-depth analysis of misclassifications made by the top-performing model from *bikit*. Building on these results, *dacl1k* [41] leveraged insights from *bikit* on effective transfer learning for bridge inspection. It further explored the influence of training configuration – such as data augmentation, image resolution, and multi-label oversampling – on model performance. This refined best practices for real-world defect classification and revealed limitations of existing datasets for practical deployment. Collectively, these works advanced research on *AIDR* through raising awareness, as well as leveraging open-source data, where modest test results underscored the need for developing a novel, more representative dataset.

Chapter 5 examines two central research questions: First, how can a benchmark be designed to support pixel-level classification, measurement, and localization of all visually identifiable defects and components relevant to bridge inspections? Second, what are the limitations of the *dacl10k* dataset, and how can these be collaboratively explored and addressed within the computer vision community? *dacl10k* introduces the first large-scale, multi-label semantic segmentation benchmark for practical visual bridge inspection, along with baseline results and qualitative comparisons to enable comprehensive evaluation for practitioners. It overcomes prior dataset limitations by providing pixel-level annotations, supporting overlapping classes, and covering a strong diversity concerning defect types and real-world conditions. The *dacl-challenge* complements the benchmark by establishing the first public competition and workshop dedicated to semantic segmentation for automated bridge inspection. The challenge revealed key performance bottlenecks, emphasized the importance of training strategies over architecture, and advanced state-of-the-art performance. Together, the *dacl10k* benchmark and the *dacl-challenge* laid the foundation for standardized evaluation and collaborative advancement in automated visual bridge inspection.

Finally, Chapter 6 focuses on the research question posed in the title of the associated publication: Does synthetic defect data enhance segmentation accuracy and robustness for real-world bridge inspections [90]? The core contribution of this work lies in the introduction of *synth-dacl*, a collection of three novel synthetic dataset extensions designed to address class imbalance and limited diversity in concrete damage segmentation, particularly for underrepresented and fine-grained defects such as *Crack* and *Cavity*. By combining physics-based rendering techniques with both cropped real-world defects and fully synthetic defect generation, the study demonstrates a significant boost in robustness and moderate segmentation accuracy.

By prioritizing data as the foundation for progress in *AIDR*, this thesis tackles the challenge of reliable bridge inspection automation through the integration of open-source datasets and the development of novel real-world and synthetic data. These contributions – together with a strong commitment to open-source principles – have significantly advanced research in the field, fostering reproducibility, scalability, and collaboration. Nevertheless, some aspects call for a stronger and more comprehensive understanding, highlighting directions for future research.

7.2 Outlook

The prospects explored in the following arise from insights from the three individual main chapters and one recently emerged research question. In particular, four key directions for future research in automated bridge inspection were identified: (*i*) transfer learning, (*ii*) damage localization, (*iii*) realism in the generation of synthetic data, and, (*iv*) integration of additional modalities for damage detection. These are discussed in the subsequent paragraphs.

(*i*) Given the high cost of manual annotation, leveraging existing datasets presents a promising approach. However, results from the *dacl1k* study [41] suggest that effective utilization of such data requires more sophisticated research in transfer learning. The work delineates three principal domain shifts that must be bridged. The first domain gap lies between the ImageNet-pretrained models and the meta datasets compiled for multi-label damage classification with six classes. Although harmonized at the task level, the fused datasets differ significantly in image characteristics, introducing another level of domain discrepancy. A third challenge arises when models fine-tuned on the aggregated in-domain data must generalize to the data distribution of real-world target datasets. To ensure robust *AIDR* performance during automated bridge inspection, the following research questions are critical:

- Which cross-domain pretraining datasets are most effective for initializing models?
- What currently available in-domain datasets are suitable, and how can they be combined without causing imbalance and distributional divergence that might impair generalization?
- Which advanced fine-tuning strategies are best suited to bridge each of the identified domain gaps?

(*ii*) With the introduction of *dacl10k* [18], the task of *AIDR*, that is tackled, transitioned from multi-label classification to multi-label segmentation. This shift enables pixel-level

damage classification, measurement, and localization on images. In practical use, the latter two must be converted from pixel level to physical units (*e.g.*, spalling area in square meters) and real-world coordinates (*e.g.*, UTM), respectively. Converting damage measurements involves a depth estimation task, whereas localization presents a pose estimation problem. Depth estimation on mobile devices is a well-studied problem, often addressed through structure-from-motion techniques [148]. In contrast, reliable damage localization remains a significant challenge – particularly in principal or rather automatic inspection scenarios where assessments are conducted using standard smartphones operated by walking users. This scenario corresponds to a pedestrian dead reckoning problem, as GPS access is often limited or unavailable, and cost constraints preclude the use of specialized sensors. Thus, a key research question arises: how can defects be reliably localized in GPS-restricted environments using only the standard sensor suite available on smartphones?
(*iii*) The *synth-dacl* extensions (see Chapter 6) were developed to enrich existing real-world data [41] with synthetic samples for semantic segmentation of concrete defects. While the extensions improved robustness, observing the accuracy shows much room for improvement. Several limitations became evident. The current synthesis approach does not fully replicate the complex textures, lighting conditions, and noise patterns present in real-world bridge imagery. This domain gap hinders transferability. Damage instances in the *synth-dacl* often exhibit idealized shapes and contexts that do not reflect the variability and ambiguity encountered in real settings. To enhance the value of synthetic datasets like *synth-dacl*, future work must aim for higher degrees of realism and variability. Promising strategies include:

- Texture-sensitive preprocessing: For instance, adding local binary patterns (LBP) as additional input channels or replacing RGB with LBP features can emphasize local structures and mitigate domain shifts caused by differences in surface texture [149].
- Style transfer methods: Better aligning global and local appearance between domains [150, 151].
- Domain randomization strategies: Offering principled ways to reduce overfitting to synthetic artifacts [152].

(*iv*) *AIDR* for visual bridge inspection has shown promising results, *e.g.* in the Engineer vs. Machine comparison in [18]; however, its performance is challenged by ambiguous damage scenarios. In the *dacl10k* dataset, several defect types – such as *Spalling*, *Washouts/Concrete corrosion*, and honeycombing (*Rockpocket*) – exhibit highly similar visual characteristics in terms of shape and texture, which can lead to misclassification. Another critical factor often neglected is the material context of the damaged area: accurate classification frequently depends on knowing whether the defect appears on concrete, steel, asphalt, or elastomeric components (*e.g.*, joint tapes). This material information further allows for inferring the structural element affected. For instance, the super- and substructure of a concrete bridge are made of concrete, whereas the presence of steel may indicate elements such as railing, bearing, or expansion joint. This is important for compliance with bridge inspection guidelines [13, 14], which require not only the identification of the damage class but also the affected building part. Moreover, purely RGB-image-based systems struggle under bad lighting conditions, *e.g.* during inspections conducted at night or in poorly lit environments, where infrared imaging could offer additional diagnostic value. Hence, integrating complementary sensing

modalities holds considerable potential for improving classification accuracy and robustness, particularly under challenging inspection conditions.

In closing, this thesis has moved *AIDR* for automated bridge inspection significantly closer to reliable, practical deployment. Yet the open questions it raises – mastering the domain gap between open-source and real-world data, localizing defects, refining synthetic outputs to capture the nuances of real-world scenes, and enriching RGB inputs with complementary sensing modalities – remain essential next steps. Engaging with these challenges will further raise efficiency and accuracy in visual bridge assessment.

Bibliography

[1] Federal Highway Research Institute (BaSt). Brückenstatistik. https://www.bast.de/DE/Statistik/Bruecken/Brueckenstatistik.pdf, 5 2025. [Accessed 15-05-2025].

[2] American Society of Civil Engineers (ASCE). ASCE's 2025 american infrastructure report card. https://infrastructurereportcard.org/, March 2025. [Accessed 15-05-2025].

[3] X. Wang, X. Ruan, J. R. Casas, and M. Zhang. Probabilistic model of traffic scenarios for extreme load effects in long-span bridges. *Structural Safety*, 106:102382, 2024. URL: https://www.sciencedirect.com/science/article/pii/S0167473023000693, doi:https://doi.org/10.1016/j.strusafe.2023.102382.

[4] M. Pircher, B. Lechner, O. Mariani, and A. Kammersberger. Damage due to heavy traffic on three rc road bridges. *Engineering Structures*, 33(12):3755–3761, 2011. URL: https://www.sciencedirect.com/science/article/pii/S0141029611003269, doi:https://doi.org/10.1016/j.engstruct.2011.08.012.

[5] Q. Meng, J. Zhu, and T. Wang. Numerical prediction of long-term deformation for prestressed concrete bridges under random heavy traffic loads. *Journal of Bridge Engineering*, 24(11):04019107, 2019. doi:10.1061/(ASCE)BE.1943-5592.0001489.

[6] K. Wardhana and F. C. Hadipriono. Analysis of recent bridge failures in the united states. *Journal of Performance of Constructed Facilities*, 17(3):144–150, 2003. doi:10.1061/(ASCE)0887-3828(2003)17:3(144).

[7] S. M. Ahmadi, S. Balahang, and S. Abolfathi. Predicting the hydraulic response of critical transport infrastructures during extreme flood events. *Engineering Applications of Artificial Intelligence*, 133:108573, 2024. doi:https://doi.org/10.1016/j.engappai.2024.108573.

[8] A. Nasr, I. Björnsson, D. Honfi, O. L. Ivanov, J. Johansson, and E. K. and. A review of the potential impacts of climate change on the safety and performance of bridges. *Sustainable and Resilient Infrastructure*, 6(3-4):192–212, 2021. URL: https://doi.org/10.1080/23789689.2019.1593003, arXiv:https://doi.org/10.1080/23789689.2019.1593003, doi:10.1080/23789689.2019.1593003.

[9] M. Morcillo, B. Chico, L. Mariaca, and E. Otero. Salinity in marine atmospheric corrosion: its dependence on the wind regime existing in the site. *Corrosion Science*, 42(1):91–104, 2000. URL: https://www.sciencedirect.com/science/article/pii/S0010938X99000487, doi:https://doi.org/10.1016/S0010-938X(99)00048-7.

[10] R. Revie and H. Uhlig. *Iron and Steel*, chapter 7, pages 115–148. John Wiley & Sons, Ltd, 2008. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470277270.ch7, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470277270.ch7, doi:https://doi.org/10.1002/9780470277270.ch7.

[11] L. CABAÑAS, B. BENITO, and M. HERRÁIZ. An approach to the measurement of the potential structural damage of earthquake ground motions. *Earthquake Engineering & Structural Dynamics*, 26(1):79–92, January 1997. doi:10.1002/(sici)1096-9845(199701)26:1<79::aid-eqe624>3.0.co;2-y.

[12] Austrian Research Association for Roads, Railways and Transport (RVS). Quality assurance for structural maintenance; Surveillance, checking and assessment of bridges and tunnels; Road bridges. Standard RVS 13.03.11, Vienna, AUT, 2011. URL: http://www.fsv.at/shop/produktdetail.aspx?IDProdukt=b87fe135-a80f-4ab8-8689-51a6922a719d.

[13] G. I. for Standardization Registered Association (DIN). Engineering structures in connection with roads - inspection and test. Standard DIN 1076:1999-11, Berlin, GER, 1999. URL: https://www.beuth.de/de/norm/din-1076/23474630.

[14] F. H. R. I. (BaSt). Guideline for the uniform acquisition, assessment, recording and evaluation of results of structural inspections (RI-EBW-PRÜF), February 2017.

[15] Inspection of highway structures. Standard, National Highways, Bristol, UK, April 2021.

[16] G. Hearn. *Bridge Inspection Practices*. Transportation Research Board, March 2007. URL: http://dx.doi.org/10.17226/14127, doi:10.17226/14127.

[17] B. M. Phares, G. A. Washer, D. D. Rolander, B. A. Graybeal, and M. Moore. Routine highway bridge inspection condition documentation accuracy and reliability. *Journal of Bridge Engineering*, 9(4):403–413, July 2004. URL: https://doi.org/10.1061/(asce)1084-0702(2004)9:4(403), doi:10.1061/(asce)1084-0702(2004)9:4(403).

[18] J. Flotzinger, P. J. Rösch, and T. Braml. dacl10k: Benchmark for semantic bridge damage segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 8626–8635, January 2024.

[19] Papers with Code - ImageNet Benchmark (Image Classification) — paperswithcode.com. https://paperswithcode.com/sota/image-classification-on-imagenet?metric=Top%201%20Accuracy&tag_filter=171. [Accessed 20-05-2025].

[20] Papers with Code - PASCAL VOC 2012 test Benchmark (Semantic Segmentation) — paperswithcode.com. https://paperswithcode.com/sota/semantic-segmentation-on-pascal-voc-2012. [Accessed 20-05-2025].

[21] Papers with Code - Cityscapes test Benchmark (Semantic Segmentation) — paperswithcode.com. https://paperswithcode.com/sota/semantic-segmentation-on-cityscapes. [Accessed 12-06-2025].

[22] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[23] A. L. Simpson, M. Antonelli, S. Bakas, M. Bilello, K. Farahani, B. van Ginneken, A. Kopp-Schneider, B. A. Landman, G. Litjens, B. Menze, O. Ronneberger, R. M. Summers, P. Bilic, P. F. Christ, R. K. G. Do, M. Gollub, J. Golia-Pernicka, S. H. Heckers, W. R. Jarnagin, M. K. McHugo, S. Napel, E. Vorontsov, L. Maier-Hein, and M. J. Cardoso. A large annotated medical image dataset for the development and evaluation of segmentation algorithms, 2019. arXiv:1902.09063.

[24] C. Benz and V. Rodehorst. Image-based detection of structural defects using hierarchical multi-scale attention. In *DAGM German Conference on Pattern Recognition (GCPR)*, pages 337–353. Springer, 2022.

[25] C. Benz and V. Rodehorst. Omnicrack30k: A benchmark for crack segmentation and the reasonable effectiveness of transfer learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2024.

[26] S. Kulkarni, S. Singh, D. Balakrishnan, S. Sharma, S. Devunuri, and S. C. R. Korlapati. Crackseg9k: A collection and benchmark for crack segmentation datasets and frameworks. In L. Karlinsky, T. Michaeli, and K. Nishino, editors, *Computer Vision – ECCV 2022 Workshops*, pages 179–195, Cham, 2023. Springer Nature Switzerland.

[27] M. Mundt, S. Majumder, S. Murali, P. Panetsos, and V. Ramesh. Meta-learning convolutional neural architectures for multi-target concrete defect classification with the concrete defect bridge image dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[28] G. Wu, Z. Chen, and J. Dang. *Bridge Inspection Automation*, pages 107–162. Springer Nature Singapore, Singapore, 2024. URL: https://doi.org/10.1007/978-981-97-3827-4_4, doi:10.1007/978-981-97-3827-4_4.

[29] J. E. Fischer, C. Greenhalgh, W. Jiang, S. D. Ramchurn, F. Wu, and T. Rodden. In-the-loop or on-the-loop? interactional arrangements to support team coordination with a planning agent. *Concurrency and Computation: Practice and Experience*, 33(8), March 2017. URL: http://dx.doi.org/10.1002/cpe.4082, doi:10.1002/cpe.4082.

[30] T. Panigati, M. Zini, D. Striccoli, P. F. Giordano, D. Tonelli, M. P. Limongelli, and D. Zonta. Drone-based bridge inspections: Current practices and future directions. *Automation in Construction*, 173:106101, 2025. URL: https://www.sciencedirect.com/science/article/pii/S0926580525001414, doi:https://doi.org/10.1016/j.autcon.2025.106101.

[31] I. Gohar, W. K. Yew, A. Halimi, and J. See. Review of state-of-the-art surface defect detection on wind turbine blades through aerial imagery: Challenges and recommendations. *Engineering Applications of Artificial Intelligence*, 144:109970, 2025. URL: https://www.sciencedirect.com/science/article/pii/S0952197624021298, doi:https://doi.org/10.1016/j.engappai.2024.109970.

[32] X. Huang, Y. Liu, L. Huang, S. Stikbakke, and E. Onstein. Bim-supported drone path planning for building exterior surface inspection. *Computers in Industry*, 153:104019, 2023. URL: https://www.sciencedirect.com/science/article/pii/S0166361523001690, doi:https://doi.org/10.1016/j.compind.2023.104019.

[33] Y. Tan, G. Li, R. Cai, J. Ma, and M. Wang. Mapping and modelling defect data from uav captured images to bim for building external wall inspection. *Automation in Construction*, 139:104284, 2022. URL: https://www.sciencedirect.com/science/article/pii/S0926580522001571, doi:https://doi.org/10.1016/j.autcon.2022.104284.

[34] W. Ma, B. Li, L. Jiang, Y. Wu, R. Bai, and G. Chen. A soft, centimeter-scaled, thin-cable-crawling robot for narrow space inspection. *Adv. Intell. Syst.*, 6(6), June 2024.

[35] A. J. Lee, W. Song, B. Yu, D. Choi, C. Tirtawardhana, and H. Myung. Survey of robotics technologies for civil infrastructure inspection. *Journal of Infrastructure Intelligence and Resilience*, 2(1):100018, 2023. URL: https://www.sciencedirect.com/science/article/pii/S2772991522000184, doi:https://doi.org/10.1016/j.iintel.2022.100018.

[36] S. T. Nguyen and H. M. La. A climbing robot for steel bridge inspection. *J. Intell. Robot. Syst.*, 102(4), August 2021.

[37] J. Feng, B. Shang, E. Hoxha, C. Hernández, Y. He, W. Wang, and J. Xiao. Robotic inspection and data analytics to localize and visualize the structural defects of concrete infrastructure. *IEEE Transactions on Automation Science and Engineering*, pages 1–1, 2025. doi:10.1109/TASE.2025.3535227.

[38] H. Liu, J. Yuan, Q. Ren, M. Li, Z. Qi, and X. Deng. Remotely operated vehicle (rov) underwater vision-based micro-crack inspection for concrete dams using a customizable cnn framework. *Automation in Construction*, 173:106102, 2025. URL: https://www.sciencedirect.com/science/article/pii/S0926580525001426, doi:https://doi.org/10.1016/j.autcon.2025.106102.

[39] T. Ni, R. Zhou, C. Gu, and Y. Yang. Measurement of concrete crack feature with android smartphone app based on digital image processing techniques. *Measurement*, 150:107093, 2020. URL: https://www.sciencedirect.com/science/article/pii/S0263224119309595, doi:https://doi.org/10.1016/j.measurement.2019.107093.

[40] J. Flotzinger, P. J. Rösch, F. Deuser, T. Braml, and B. Maradni. Applying automated damage classification during digital inspection of structures. In *2022 Conference on*

Structural Engineering, Mechanics and Computation (SEMC), pages 1863–1869, 2022. doi:https://doi.org/10.1201/9781003348443.

[41] J. Flotzinger, P. J. Rösch, N. Oswald, and T. Braml. dacl1k: Real-world bridge damage dataset putting open-source data to the test. *Engineering Applications of Artificial Intelligence*, 137:109106, 2024. URL: https://www.sciencedirect.com/science/article/pii/S0952197624012648, doi:https://doi.org/10.1016/j.engappai.2024.109106.

[42] C. M. Eastman. *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*. John Wiley & Sons, 2011.

[43] A. Bochkovskii, A. Delaunoy, H. Germain, M. Santos, Y. Zhou, S. R. Richter, and V. Koltun. Depth pro: Sharp monocular metric depth in less than a second, 2025. URL: https://arxiv.org/abs/2410.02073, arXiv:2410.02073.

[44] A. Ganj, H. Su, and T. Guo. Hybriddepth: Robust metric depth fusion by leveraging depth from focus and single-image priors. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pages 973–982, February 2025.

[45] N. Zhang, F. Nex, G. Vosselman, and N. Kerle. Lite-mono: A lightweight cnn and transformer architecture for self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18537–18546, June 2023.

[46] X. Peng, Z. Liu, W. Li, P. Tan, S. Y. Cho, and Q. Wang. Dvi-slam: A dual visual inertial slam network. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12020–12026, 2024. doi:10.1109/ICRA57147.2024.10610042.

[47] Y. Pan, W. Zhou, Y. Cao, and H. Zha. Adaptive vio: Deep visual-inertial odometry with online continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18019–18028, June 2024.

[48] M. Yang, Y. Chen, and H.-S. Kim. Efficient deep visual and inertial odometry with adaptive visual modality selection. In S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, editors, *Computer Vision – ECCV 2022*, pages 233–250, Cham, 2022. Springer Nature Switzerland.

[49] F. H. R. I. (BaSt). Guideline for the uniform acquisition, assessment, recording and evaluation of results of structural inspections defect examples (RI-EBW-PRÜF Schadensbeispiele), February 2017.

[50] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn. Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, 51(11):1016–1022, 2018. 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018. URL: https://www.sciencedirect.com/science/article/pii/S2405896318316021, doi:https://doi.org/10.1016/j.ifacol.2018.08.474.

[51] S. Dorafshan, R. J. Thomas, and M. Maguire. SDNET2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks. *Data in Brief*, 21:1664–1668, December 2018. doi:10.1016/j.dib.2018.11.015.

[52] P. Hüthwohl and I. Brilakis. Detecting healthy concrete surfaces. *Advanced Engineering Informatics*, 37(April):150–162, 2018. URL: https://doi.org/10.1016/j.aei.2018.05.004, doi:10.1016/j.aei.2018.05.004.

[53] H. Xu, X. Su, H. Xu, and H. Li. Autonomous bridge crack detection using deep convolutional neural networks. In *3rd Int. Conf. on Computer Engineering, Information Science & Application Technology*, pages 274–284. Atlantis Press, 2019.

[54] P. Hüthwohl, R. Lu, and I. Brilakis. Multi-classifier for reinforced concrete bridge defects. *Automation in Construction*, 105, Sep 2019. doi:10.1016/j.autcon.2019.04.019.

[55] S. Kulkarni, S. Singh, D. Balakrishnan, S. Sharma, S. Devunuri, and S. C. R. Korlapati. Crackseg9k: a collection and benchmark for crack segmentation datasets and frameworks. In *European Conference on Computer Vision*, pages 179–195. Springer, 2022.

[56] F. undefinedelik and M. König. A sigmoid-optimized encoder–decoder network for crack segmentation with copy-edit-paste transfer learning. *Computer-Aided Civil and Infrastructure Engineering*, 37(14):1875–1890, April 2022. URL: http://dx.doi.org/10.1111/mice.12844, doi:10.1111/mice.12844.

[57] J. Flotzinger, P. J. Rösch, N. Oswald, and T. Braml. dacl1k: Real-world bridge damage dataset putting open-source data to the test, 2023. URL: https://arxiv.org/abs/2309.03763, arXiv:2309.03763.

[58] M. Eliseev. Supervisely synthetic crack segmentation dataset, 2023. URL: https://supervisely.com/blog/introducing-supervisely-synthetic-crack-segmentation-dataset/.

[59] J. Flotzinger, P. J. Rösch, and T. Braml. dacl10k: Benchmark for semantic bridge damage segmentation, 2023. URL: https://arxiv.org/abs/2309.00460, arXiv:2309.00460.

[60] Z. He, C. Su, and Y. Deng. A novel mo-yolov4 for segmentation of multi-class bridge damages. *Advanced Engineering Informatics*, 62:102586, 2024. URL: https://www.sciencedirect.com/science/article/pii/S1474034624002349, doi:https://doi.org/10.1016/j.aei.2024.102586.

[61] A. Jaziri, M. Mundt, A. Fernandez, and V. Ramesh. Designing a hybrid neural system to learn real-world crack segmentation from fractal-based simulation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 8636–8646, January 2024.

[62] P. J. Rösch and J. Flotzinger. Building inspection toolkit. `https://github.com/phiyodr/building-inspection-toolkit`, 2022.

[63] J. Flotzinger, P. J. Rösch, N. Oswald, and T. Braml. Building inspection toolkit: Unified evaluation and strong baselines for bridge damage recognition. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 1221–1225, 2022. `doi:10.1109/ICIP46576.2022.9897743`.

[64] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3234–3243, 2016. `doi:10.1109/CVPR.2016.352`.

[65] M. Roberts, J. Ramapuram, A. Ranjan, A. Kumar, M. A. Bautista, N. Paczan, R. Webb, and J. M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *International Conference on Computer Vision (ICCV) 2021*, 2021.

[66] J. Tremblay, T. To, and S. Birchfield. Falling Things: A Synthetic Dataset for 3D Object Detection and Pose Estimation . In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2119–21193, Los Alamitos, CA, USA, June 2018. IEEE Computer Society. URL: `https://doi.ieeecomputersociety.org/10.1109/CVPRW.2018.00275`, `doi:10.1109/CVPRW.2018.00275`.

[67] F. Xia, J. Liu, H. Nie, Y. Fu, L. Wan, and X. Kong. Random walks: A review of algorithms and applications. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 4(2):95–107, 2019.

[68] B. Li and B. Chen. An adaptive rapidly-exploring random tree. *IEEE/CAA Journal of Automatica Sinica*, 9(2):283–294, 2021.

[69] J. McCormack et al. Interactive evolution of l-system grammars for computer graphics modelling. *Complex Systems: from biology to computation*, 2, 1993.

[70] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam, 2025. URL: `http://www.blender.org`.

[71] M. Ali, M. A. Gennert, and T. G. Clarkson. Analysis, generation and compression of pavement distress images using fractals. In A. J. Crilly, R. A. Earnshaw, and H. Jones, editors, *Applications of Fractals and Chaos*, pages 147–169, Berlin, Heidelberg, 1993. Springer Berlin Heidelberg.

[72] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. `doi:10.1109/5.726791`.

[73] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. `doi:10.1109/CVPR.2009.5206848`.

[74] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[75] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi:10.1007/s11263-015-0816-y.

[76] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. URL: https://arxiv.org/abs/1409.1556, arXiv:1409.1556.

[77] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi:10.1109/CVPR.2016.90.

[78] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994. doi:10.1109/72.279181.

[79] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Y. W. Teh and M. Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL: https://proceedings.mlr.press/v9/glorot10a.html.

[80] R. Wightman, H. Touvron, and H. Jégou. Resnet strikes back: An improved training procedure in timm, 2021. URL: https://arxiv.org/abs/2110.00476, arXiv:2110.00476.

[81] X. Dong, J. Bao, T. Zhang, D. Chen, W. Zhang, L. Yuan, D. Chen, F. Wen, N. Yu, and B. Guo. Peco: Perceptual codebook for bert pre-training of vision transformers, 2022. URL: https://arxiv.org/abs/2111.12710, arXiv:2111.12710.

[82] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019. URL: https://arxiv.org/abs/1909.13719, arXiv:1909.13719.

[83] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features, 2019. URL: https://arxiv.org/abs/1905.04899, arXiv:1905.04899.

[84] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization, 2018. URL: https://arxiv.org/abs/1710.09412, arXiv:1710.09412.

[85] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[86] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL: https://arxiv.org/abs/2010.11929, arXiv:2010.11929.

[87] Z. Tu, H. Talebi, H. Zhang, F. Yang, P. Milanfar, A. Bovik, and Y. Li. Maxvit: Multi-axis vision transformer. In S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, editors, *Computer Vision – ECCV 2022*, pages 459–479, Cham, 2022. Springer Nature Switzerland.

[88] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection, 2017. URL: https://arxiv.org/abs/1612.03144, arXiv:1612.03144.

[89] J. Flotzinger, P. J. Rösch, C. Benz, M. Ahmad, M. Cankaya, H. Mayer, V. Rodehorst, N. Oswald, and T. Braml. Dacl-challenge: Semantic segmentation during visual bridge inspections. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, pages 716–725, January 2024.

[90] J. Flotzinger, F. Deuser, A. Jaziri, H. Neumann, N. Oswald, V. Ramesh, and T. Braml. synth-dacl: Does synthetic defect data enhance segmentation accuracy and robustness for real-world bridge inspections?, 2025. URL: https://arxiv.org/abs/2506.14255, arXiv:2506.14255.

[91] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders, 2023. URL: https://arxiv.org/abs/2301.00808, arXiv:2301.00808.

[92] M. Tan and Q. V. Le. Efficientnetv2: Smaller models and faster training, 2021. URL: https://arxiv.org/abs/2104.00298, arXiv:2104.00298.

[93] X. Dong, J. Bao, D. Chen, W. Zhang, N. Yu, L. Yuan, D. Chen, and B. Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows, 2022. URL: https://arxiv.org/abs/2107.00652, arXiv:2107.00652.

[94] H. Chen, Y. Wang, J. Guo, and D. Tao. Vanillanet: the power of minimalism in deep learning, 2023. URL: https://arxiv.org/abs/2305.12972, arXiv:2305.12972.

[95] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013. doi:10.1109/TPAMI.2012.231.

[96] P. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In E. P. Xing and T. Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 82–90, Bejing, China, 22–24 Jun 2014. PMLR. URL: https://proceedings.mlr.press/v32/pinheiro14.html.

[97] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation, 2015. URL: https://arxiv.org/abs/1411.4038, arXiv:1411.4038.

[98] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, June 2014. URL: http://dx.doi.org/10.1007/s11263-014-0733-5, doi:10.1007/s11263-014-0733-5.

[99] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context, 2015. URL: https://arxiv.org/abs/1405.0312, arXiv:1405.0312.

[100] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019.

[101] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs, 2016. URL: https://arxiv.org/abs/1412.7062, arXiv:1412.7062.

[102] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017. URL: https://arxiv.org/abs/1606.00915, arXiv:1606.00915.

[103] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation, 2017. URL: https://arxiv.org/abs/1706.05587, arXiv:1706.05587.

[104] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018. URL: https://arxiv.org/abs/1802.02611, arXiv:1802.02611.

[105] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017. URL: https://arxiv.org/abs/1412.6980, arXiv:1412.6980.

[106] I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2019. URL: https://arxiv.org/abs/1711.05101, arXiv:1711.05101.

[107] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[108] F. Hutter, L. Kotthoff, and J. Vanschoren. *Automated Machine Learning: Methods, Systems, Challenges*. Springer Publishing Company, Incorporated, 1st edition, 2019.

[109] H. Li, P. Chaudhari, H. Yang, M. Lam, A. Ravichandran, R. Bhotika, and S. Soatto. Rethinking the hyperparameters for fine-tuning, 2020. URL: https://arxiv.org/abs/2002.11770, arXiv:2002.11770.

[110] F. Chollet. *Deep learning with python*. Manning Publications, New York, NY, October 2017.

[111] K. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big Data*, 3(1), May 2016. URL: http://dx.doi.org/10.1186/s40537-016-0043-6, doi:10.1186/s40537-016-0043-6.

[112] C. Matsoukas, J. F. Haslum, M. Sorkhei, M. Söderberg, and K. Smith. What makes transfer learning work for medical images: Feature reuse & other factors, 2022. URL: https://arxiv.org/abs/2203.01825, arXiv:2203.01825.

[113] S. P. Mohanty, D. P. Hughes, and M. Salathé. Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, Volume 7 - 2016, 2016. URL: https://www.frontiersin.org/journals/plant-science/articles/10.3389/fpls.2016.01419, doi:10.3389/fpls.2016.01419.

[114] D. Lee, J. Lee, and E. Park. Automated vehicle damage classification using the three-quarter view car damage dataset and deep learning approaches. *Heliyon*, 10(14):e34016, 2024. URL: https://www.sciencedirect.com/science/article/pii/S2405844024100473, doi:https://doi.org/10.1016/j.heliyon.2024.e34016.

[115] Z. A. Bukhsh, N. Jansen, and A. Saeed. Damage detection using in-domain and cross-domain transfer learning. *Neural Comput. Appl.*, 33(24):16921–16936, December 2021.

[116] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015. doi:10.1109/ICCV.2015.123.

[117] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks?, 2014. URL: https://arxiv.org/abs/1411.1792, arXiv:1411.1792.

[118] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition, 2014. URL: https://arxiv.org/abs/1403.6382, arXiv:1403.6382.

[119] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets, 2014. URL: https://arxiv.org/abs/1405.3531, arXiv:1405.3531.

[120] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[121] E. D. Cubuk, B. Zoph, D. Mané, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation strategies from data. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 113–123, 2019. doi:10.1109/CVPR.2019.00020.

[122] S. G. Müller and F. Hutter. Trivialaugment: Tuning-free yet state-of-the-art data augmentation, 2021. URL: https://arxiv.org/abs/2103.10158, arXiv:2103.10158.

[123] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim. Fast autoaugment, 2019. URL: https://arxiv.org/abs/1905.00397, arXiv:1905.00397.

[124] D. Dwibedi, I. Misra, and M. Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1301–1310, 2017.

[125] Y. Al Khalil, S. Amirrajab, C. Lorenz, J. Weese, J. Pluim, and M. Breeuwer. On the usability of synthetic data for improving the robustness of deep learning-based segmentation of cardiac magnetic resonance images. *Medical Image Analysis*, 84:102688, 2023.

[126] Q. Lhoest, A. V. del Moral, Y. Jernite, A. Thakur, P. von Platen, S. Patil, J. Chaumond, M. Drame, J. Plu, L. Tunstall, J. Davison, M. Šaško, G. Chhablani, B. Malik, S. Brandeis, T. L. Scao, V. Sanh, C. Xu, N. Patry, A. McMillan-Major, P. Schmid, S. Gugger, C. Delangue, T. Matussière, L. Debut, S. Bekman, P. Cistac, T. Goehringer, V. Mustar, F. Lagunas, A. M. Rush, and T. Wolf. Datasets: A community library for natural language processing, 2021. arXiv:2109.02846.

[127] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo. OpenML: Networked Science in Machine Learning. *SIGKDD Explorations*, 15(2):49–60, 2013. URL: http://doi.acm.org/10.1145/2641190.2641198, doi:10.1145/2641190.2641198.

[128] J. Flotzinger and T. Braml. Automatisierte Klassifizierung von Schäden an Massivbrücken mittels Neuronaler Netze [Automated Damage Classification on Concrete Bridges using Convolutional Neural Networks]. *Beton Stahlbetonbau*, 117(10):786–794, October 2022.

[129] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[130] E. Triantafillou, T. Zhu, V. Dumoulin, P. Lamblin, U. Evci, K. Xu, R. Goroshin, C. Gelada, K. Swersky, P.-A. Manzagol, and H. Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*, 2020. URL: https://openreview.net/forum?id=rkgAGAVKPr.

[131] I. Ullah, D. Carrión-Ojeda, S. Escalera, I. Guyon, M. Huisman, F. Mohr, J. N. van Rijn, H. Sun, J. Vanschoren, and P. A. Vu. Meta-album: Multi-domain meta-dataset for few-shot image classification. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave,

K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 3232–3247. Curran Associates, Inc., 2022.

[132] Z. Xue, F. Yang, S. Rajaraman, G. Zamzmi, and S. Antani. Cross dataset analysis of domain shift in cxr lung region detection. *Diagnostics*, 13(6), 2023. URL: https://www.mdpi.com/2075-4418/13/6/1068, doi:10.3390/diagnostics13061068.

[133] L. van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

[134] J. Flotzinger, P. J. Rösch, and T. Braml. dacl10k, 2025. URL: https://doi.org/10.60776/RQUOYN, doi:10.60776/RQUOYN.

[135] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[136] G. J. Brostow, J. Fauqueur, and R. Cipolla. Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97, 2009. Video-based Object and Event Analysis. doi:https://doi.org/10.1016/j.patrec.2008.04.005.

[137] T. Scharwächter, M. Enzweiler, U. Franke, and S. Roth. Efficient multi-cue scene segmentation. In J. Weickert, M. Hein, and B. Schiele, editors, *Pattern Recognition*, pages 435–445, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[138] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2633–2642, 2020. doi:10.1109/CVPR42600.2020.00271.

[139] P. F. Raudaschl, P. Zaffino, G. C. Sharp, M. F. Spadea, A. Chen, B. M. Dawant, T. Albrecht, T. Gass, C. Langguth, M. Lüthi, F. Jung, O. Knapp, S. Wesarg, R. Mannion-Haworth, M. Bowes, A. Ashman, G. Guillard, A. Brett, G. Vincent, M. Orbes-Arteaga, D. Cárdenas-Peña, G. Castellanos-Dominguez, N. Aghdasi, Y. Li, A. Berens, K. Moe, B. Hannaford, R. Schubert, and K. D. Fritscher. Evaluation of segmentation methods on head and neck ct: Auto-segmentation challenge 2015. *Medical Physics*, 44(5):2020–2036, 2017. arXiv:https://aapm.onlinelibrary.wiley.com/doi/pdf/10.1002/mp.12197, doi:https://doi.org/10.1002/mp.12197.

[140] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. arXiv:1905.11946.

[141] J. B. Haurum, M. Wang, A. Mian, and T. B. Moeslund. Computer Vision for Civil and Infrastructure Engineering Workshop (CVCIE @ ECCV2022). https://vap.aau.dk/cvcie/. Accessed: 2023-11-15.

[142] T. Frick, D. Antognini, M. Rigotti, I. Giurgiu, B. Grewe, and C. Malossi. Active learning for imbalanced civil infrastructure data. In L. Karlinsky, T. Michaeli, and K. Nishino, editors, *Computer Vision – ECCV 2022 Workshops*, pages 283–298, Cham, 2023. Springer Nature Switzerland.

[143] K. Janouskova, M. Rigotti, I. Giurgiu, and C. Malossi. Model-assisted labeling via explainability for visual inspection of civil infrastructures. In L. Karlinsky, T. Michaeli, and K. Nishino, editors, *Computer Vision – ECCV 2022 Workshops*, pages 244–257, Cham, 2023. Springer Nature Switzerland.

[144] H. Bai, G. Cinbis, M. Cao, T. Likhomanenko, and S. M. O. Tuzel. 1st workshop on Vision-based InduStrial InspectiON (VISION@CVPR2023). `https://vision-based-industrial-inspection.github.io/cvpr-2023/`. Accessed: 2023-11-15.

[145] Open Images Challenge. `https://storage.googleapis.com/openimages/web/challenge_overview.html`. [Accessed 29-05-2025].

[146] J. Flotzinger, F. Deuser, and A. Jaziri. synth-dacl: Synthetic Extensions for dacl10k Dataset, 2025. URL: `https://doi.org/10.60776/9D6E4M`, `doi:10.60776/9D6E4M`.

[147] Y. Wang, Y. Zhao, and L. Petzold. An empirical study on the robustness of the segment anything model (sam). *Pattern Recognition*, 155:110685, 2024. URL: `https://www.sciencedirect.com/science/article/pii/S0031320324004369`, `doi:https://doi.org/10.1016/j.patcog.2024.110685`.

[148] J. Valentin, A. Kowdle, J. T. Barron, N. Wadhwa, M. Dzitsiuk, M. J. Schoenberg, V. Verma, A. Csaszar, E. L. Turner, I. Dryanovski, J. Afonso, J. Pascoal, K. N. J. Tsotsos, M. A. Leung, M. Schmidt, O. G. Guleryuz, S. Khamis, V. Tankovich, S. Fanello, S. Izadi, and C. Rhemann. Depth from motion for smartphone ar. *ACM Transactions on Graphics*, 2018. URL: `https://dl.acm.org/citation.cfm?id=3275041`.

[149] T. Hess, M. Mundt, I. Pliushch, and V. Ramesh. A procedural world generation framework for systematic evaluation of continual learning, 2021. URL: `https://arxiv.org/abs/2106.02585`, `arXiv:2106.02585`.

[150] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu. Contrastive learning for unpaired image-to-image translation, 2020. URL: `https://arxiv.org/abs/2007.15651`, `arXiv:2007.15651`.

[151] V. Prabhu, D. Acuna, A. Liao, R. Mahmood, M. T. Law, J. Hoffman, S. Fidler, and J. Lucas. Bridging the sim2real gap with care: Supervised detection adaptation with conditional alignment and reweighting, 2023. URL: `https://arxiv.org/abs/2302.04832`, `arXiv:2302.04832`.

[152] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world, 2017. URL: `https://arxiv.org/abs/1703.06907`, `arXiv:1703.06907`.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Prof. Dr.-Ing. Thomas Braml. His unwavering support and insightful guidance – both in the realm of research and across various applied projects – have far exceeded the expectations of a supervisor. Drawing from his structural engineering expertise and his broad professional network, he played a pivotal role in shaping my work. His steadfast encouragement, coupled with a liberal and open-minded mentorship style, empowered me to step beyond my comfort zone and pursue ambitious application-driven research. I am especially grateful that he entrusted me with an application-driven, deep learning–focused project at a time when I had no prior programming experience.

I am also sincerely grateful to Prof. Dr. Heiko Neumann for his insightful feedback, stimulating discussions, and for co-authoring with me.

I am thankful to all my past and present colleagues for their intellectual companionship. A special thank-you goes to my collaborators, whose constructive input and support played a vital role in refining and strengthening this work. In particular, I am deeply indebted to Fabian Deuser, Philipp J. Rösch, Johannes Kreutz, and Vinko Novak for their invaluable contributions and inspiring collaboration.

Lastly, my deepest gratitude goes to my partner, Lisa, who has stood by me with unwavering support and belief, especially in moments of doubt. She has been my anchor throughout this journey.