Meta-learning Convolutional Neural Architectures for Multi-target Concrete Defect Classification with the COncrete DEfect BRidge IMage Dataset: Supplementary Material

Martin Mundt[1*], Sagnik Majumder[1], Sreenivas Murali[1*], Panagiotis Panetsos[2], Visvanathan Ramesh[1*]
1. Goethe University 2. Egnatia Odos A. E.
{mmundt, vramesh}@em.uni-frankfurt.de {majumder, murali}@ccc.cs.uni-frankfurt.de
ppane@egnatia.gr

A. Content overview

The supplementary material contains further details for material presented in the main body.

We start with an extended description of the CODEBRIM dataset in section B. Here, we provide the specific settings for the employed cameras for dataset image acquisition. In addition to the histogram presented in the main body that shows number of different defects per bounding box, we further provide a histogram with amount of bounding box annotations per image. Additional material reveals specifics of the main body's figure depicting the large variations in distribution of bounding boxes by illustrating the individual nuances of this distribution per defect class. The supplementary dataset material is concluded with a brief discussion on background patch generation.

In supplementary section C we provide a brief discussion on why multi-target accuracy is a better reward metric than naive single-class accuracies and show what multi-target accuracies would translate to in terms of a naive average single-class accuracy. We give detailed descriptions and graphs of the six meta-learned architectures for the top three models obtained through MetaQNN and ENAS. Although it isn't an immediate extension to the main body, but rather additional content, we provide a compact section on transfer learning with experiments conducted with models pre-trained on the ImageNet and MINC datasets. We have decided to move these experiments to the the supplementary material for the interested reader as they do not show any improvements over the content presented in the main body. We conclude the supplementary material with examples for images that are commonly classified correctly as well as showing some typical false multi-target classifications to give the reader a better qualitative understanding of the dataset complexity and challenges.

* work conducted while at Frankfurt Institute for Advanced Studies

B. CODEBRIM dataset

B.1. Delamination as a defect class

Some of the CODEBRIM dataset features images that have a defect that is typically referred to as delimation. It is a stage where areas start to detach from the surface. Delamination can thus be recognized by a depth offset of a layer from the main surface body. However, in images acquired by a single camera, especially if the images were acquired using a camera view direction that is orthogonal to the surface, these boundaries are often visually not distinguishable from cracks. Without further information, even a civil engineering expert faces major difficulty in such a distinction between these categories. We have thus decided to label eventual occurrences of delamination together with the crack category.

B.2. Cameras

We show the four cameras used for acquisition of dataset images in table 1. All chosen cameras have a resolution above Full-HD, with the highest resolution going up to 6000×4000 pixels. For two cameras we have used a lens with varying focal length, whereas two cameras had a lens with fixed focal length of 50 and $55\,\mathrm{mm}$ respectively. We have further systematically varied aperture in conjunction with the use of diffused flash modules to homogeneously illuminate dark bridge areas, while also adjusting for changing global illumination (avoiding heavy over or under-exposure). A different very crucial aspect was the employed exposure time. Pictures acquired by UAV were generally captured with a much shorter exposure time to avoid blurring of the image due to out of focus acquisition or inherent vibration and movement of the UAV. One of our cameras, Sony α-6000 has thus exclusively been used in the context of UAV based image acquisition with an exposure time of $1/1000$ seconds.

We show how the CODEBRIM dataset is practically

Camera	Resolution [pixels]	Exposure [s]	f [mm]	F-value [f/]	ISO	Flash
Canon IXUS 870 IS	2592×1944	flexible	$5 - 20$	$2.8 - 5.8$	$100 - 800$	none
Panasonic DMC-FZ72	4608×3456	$1/250$	$4 - 42$	5.6	400	built-in
Nikon D5200	6000×4000	$1/200$	55	11.0	200	built-in
Sony α-6000	6000×3376	$1/1000$	50	$2.0 - 5.6$	$50 - 400$	HVL-F43M

Table 1: Description of cameras, including resolution, exposure time in fraction of a second, focal length f in mm, the aperture or F-value in terms of focal length, ISO speed rating and information on potentially used flash.

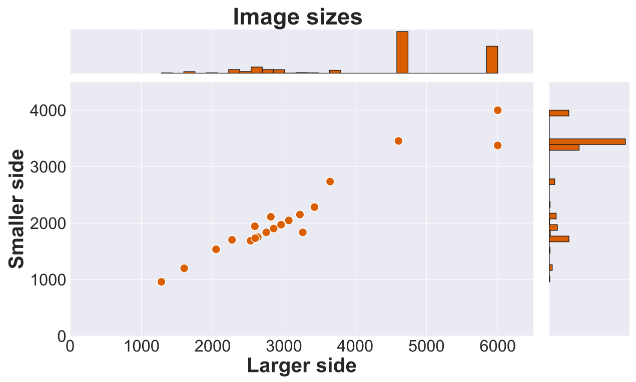


Figure 1: Distribution of image resolutions. Smaller and larger side refer to the image's larger and smaller dimension.

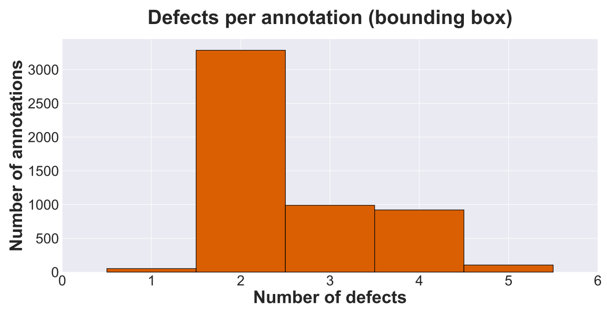


Figure 2: Histogram of number of simultaneously occurring defect classes per annotated bounding box.

comprised of the varying resolutions resulting from use with different cameras and settings in figure 1. We can observe that the aspect ratio is almost constant with changes in absolute resolution and that the large majority of images has been acquired at very high resolutions.

B.3. Annotation process

After curating acquired images by excluding the majority of images that do not have defects, we employed a multi-stage annotation process to create a multi-class multi-target classification dataset using the annotation tool LabelImg [2]

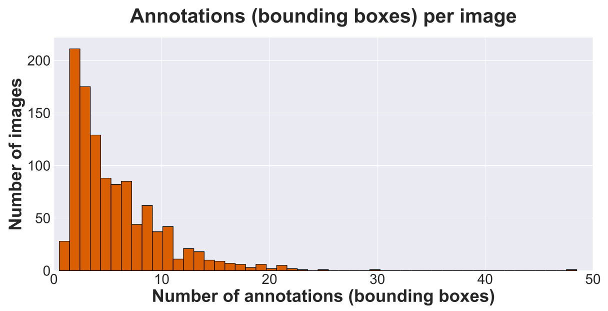


Figure 3: Distribution of number of bounding box annotations per image.

in consultation with civil engineering experts:

1. We first annotated bounding boxes for areas containing defects in the Pascal format [1].
2. Each individual bounding box was analyzed with respect to one defect class and a corresponding label was set if the defect is present.
3. After finishing the entire set of bounding boxes for one class, we repeated step 2 for the remaining classes and arrived at a multi-class multi-target annotation.
4. In the last stage, we sampled bounding boxes containing background (concrete without defects as well as non-concrete) according to absolute count, aspect ratios and size of annotated defect bounding boxes.

The reason for staging the process is that we found the annotation process to be less error prone if annotators had to concentrate on the presence of one defect at a time.

B.4. Further dataset statistics

We show additional information and statistics of the dataset. In figure 2 we show a histogram that demonstrates how one bounding box annotation typically contains more than one defect class at a time. In figure 3 the complementary histogram of the number of annotated bounding boxes per image can be found. Here, we can further observe

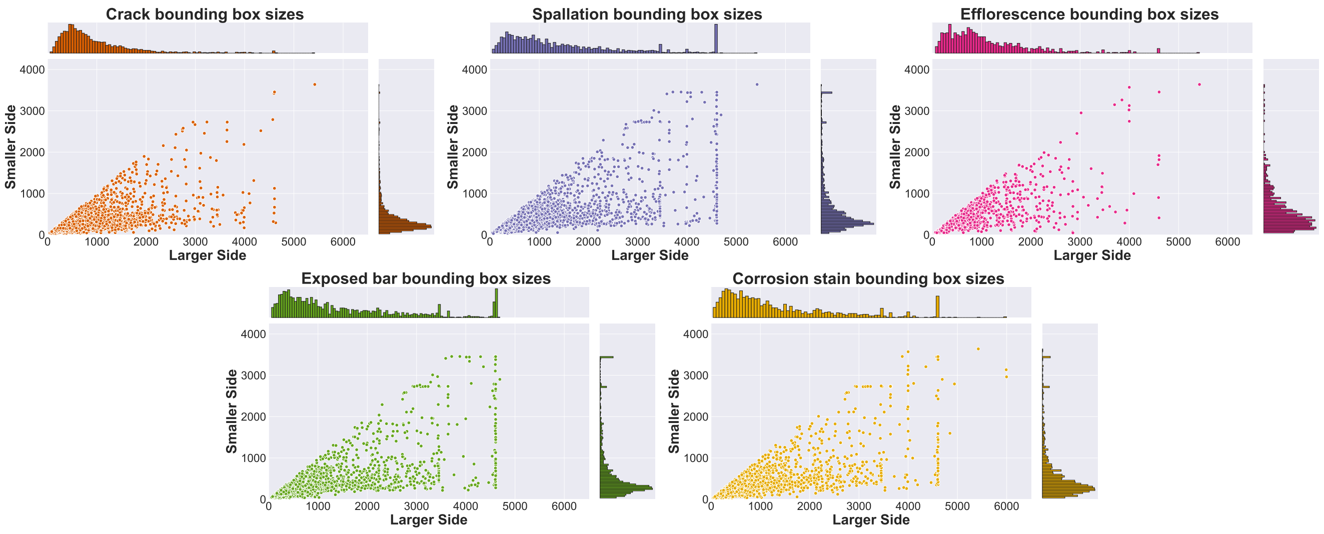


Figure 4: Individual distributions of annotated bounding box sizes for each of the 5 defect classes.

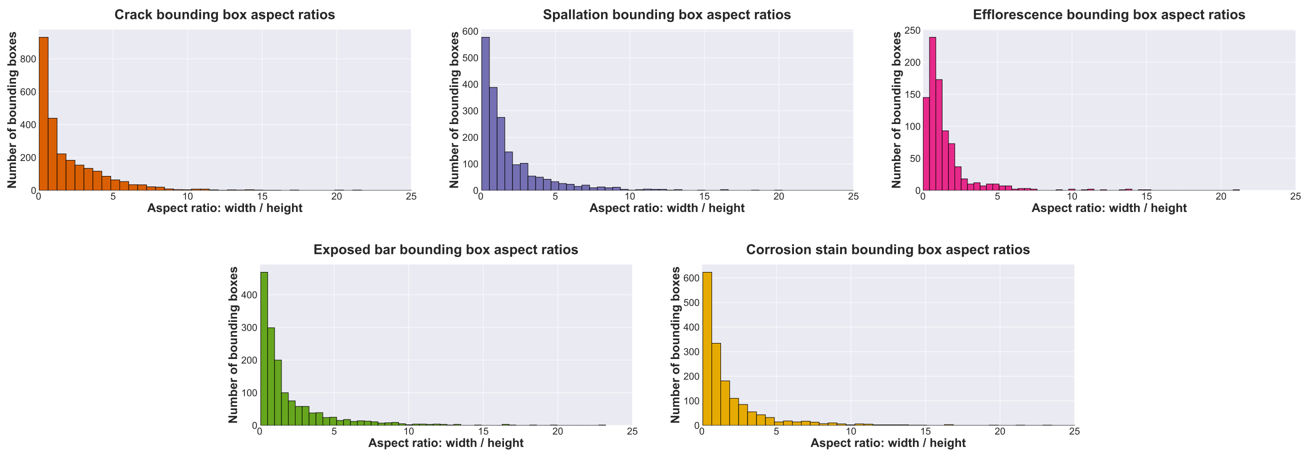


Figure 5: Individual distributions of number of bounding box annotations for different aspect ratios for each of the 5 defect classes.

that our choice of bridges led to image acquisition scenarios where one acquired image generally contains multiple different defect locations. While this is not impacting our classification task, we believe it is a crucial precursor for future extensions to a realistic semantic segmentation scenario.

In addition to the distribution of the annotated bounding box sizes for background and for all the defects combined as shown in the main body, the reader might be interested in the specific distribution per defect class. In figure 4 the corresponding distribution of annotated bounding boxes per-class is shown. Similarly, figure 5 contrasts the aspect ratio distributions for the individual defects. It is to be noted that these per-class distributions are not mutually exclusive because of multi-target overlap in the bounding box annotations. All individual classes have a similarly distributed bounding box size per defect including a long tail towards large resolutions. A major difference for individual classes can be found at high resolutions between the crack and efflorescence classes and the spallation, exposed bar and corrosion stain classes. The latter sometimes span an entire image. While this of course depends on the acquisition distance, we point out that in images acquired at a similar distance spalled and corroded areas including bar exposition are larger on average.

B.5. Random generation of background bounding boxes

We emphasize that the CODEBRIM dataset has many factors that add to the complexity. Acquired images have

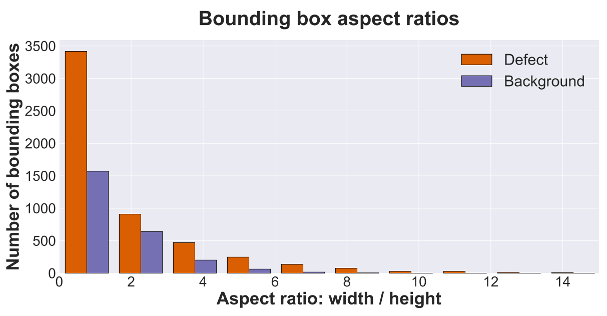


Figure 6: Distribution of number of defect and background bounding box annotations for different aspect ratios.

large variations depending on the target geometry, types of defects and their overlapping behavior, the camera pose relative to the photographed surface (particularly if captured by UAV), as well as global scene properties such as illumination. From a machine learning classification dataset point of view it is thus interesting to capture this complexity in the generation of image patches for the background class.

We therefore devote this supplementary section to provide further details to the reader on generation of background bounding boxes. Before administering the final dataset, the last dataset creation step of sampling areas containing background has been validated. Specifically, we have checked whether the distribution of sizes (shown in the main body) as well as the distribution of sampled areas' aspect ratios approximately follow those of the human annotated defects. In figure 6 we show the aspect ratios for the annotated defects together with the sampled bounding box aspect ratios for background. Whereas the overall count for background is less than the integrated total amount of defects (number of background samples roughly corresponds to occurrence of each individual defect class), the distribution of aspect ratios is confirmed to have the same trend. We have further made sure that none of the background bounding boxes have any overlap with bounding boxes annotated for defects and that bounding boxes for background are evenly distributed among images. In summary, this methodology captures the complexity of surface variations, target geometry, global illumination and makes sure that image patch resolution and sizes reflect those of defect annotations.

C. Deep convolutional neural networks for multi-target defect classification

C.1. Per-class and multi-target CNN accuracies

As mentioned in the main body of this work, most image classification tasks focus on the single target scenario and an easy pitfall would be to treat our task in a similar fashion. This would imply reporting classification accuracies independently per class and not treating the task in the multi-target fashion. We remind the reader that this would not represent the real-world scenario appropriately, where one is interested in the severity of the degradation of the inspected concrete structure. This severity is magnified if two or more different defect classes are mutually occurring and overlapping. Nevertheless, one idea could be to design the reward for the meta-learning algorithms based on such individual class accuracies or the corresponding average. We report the validation accuracy per class (background and five defects) and their respective average for the CNN literature baselines, together with the multi-target best validation accuracy and the corresponding test accuracy in table 2. Note that we do this only for the sake of completeness as this thought could occur to other researchers and to show researchers the relationship between accuracy values. Initially, a multi-target accuracy of 65% might not look like a large value, but it practically translates to around 90% classification accuracy had each class been treated independently in our task. Apart from the above stated obvious argument of resemblance to real-world application, the table further indicates why the multi-target accuracy is a better metric to employ in meta-learning reward design. Although each of the baseline models learns to recognize individual defects in the image with high precision, they are not equally competent at recognizing all the defects together in the multi-target scenario. The individual class accuracies do not have clear trends as they fluctuate individually, are difficult to interpret from one model to the next and do not intuitively correlate with multi-target values. It is thus a bad idea to base evaluation and model comparison on single-target values and then later report multi-target accuracies as the former does not linearly correlate with the latter. We have noticed that rewards designed on the average per-class instead of multi-target accuracies lead to models that learn to predict only a subset of classes correctly and neglect overlaps as there is no reward for higher recognition rate of these overlaps.

C.2. Meta learned architecture definitions

We show the detailed configurations of the top three MetaQNN and ENAS neural architectures for which accuracies are shown in the main body.

Table 3 shows the definitions for the top three meta-learned models from MetaQNN on our task. Each convolutional layer is expressed through quadratic filter size and number of filters, followed by an optional specification of padding or stride. If a skip connection/convolution has been added it is added as an additional operation on the same level and we specify the layer to which it skips to. The SPP layer is characterized by the number of scales at which it pools its feature input. As an example, scales = 4 indicates

Architecture	Accuracy [%]								
	mt best val	mt bv-test	bv-bg	bv-cr	bv-sp	bv-ef	bv-eb	bv-cs	bv-avg
Alexnet	63.05	66.98	89.30	89.30	89.93	90.72	93.71	88.05	90.16
T-CNN	64.30	67.93	90.09	87.89	89.62	88.99	94.49	87.57	89.77
VGG-A	64.93	70.45	91.35	90.25	89.93	90.56	93.55	86.47	90.35
VGG-D	64.00	70.61	90.72	91.82	89.93	89.30	93.71	87.42	90.48
WRN-28-4	52.51	57.19	87.89	84.11	85.53	84.43	89.15	80.34	85.24
DenseNet-121	65.56	70.77	91.51	89.62	87.75	89.10	94.49	87.73	90.03

Table 2: Best validation model's accuracies for each individual class (bg - background, cr - crack, sp - spallation, ef - efflorescence, eb - exposed bars, cs - corrosion stain) and their average (avg) shown together with the multi-target accuracy (mt best val) and the corresponding multi-target test accuracy (mt bv-test).

Layer type	MetaQNN-1	MetaQNN-2	MetaQNN-3
conv 1	9×9 - 256, s = 2	5×5 - 128	3×3 - 128, p = 1; 1×1 - 128 (skip to 3)
conv 2	3×3 - 32, p = 1	7×7 - 32, s = 2	3×3 - 128, p = 1
conv 3	5×5 - 256	3×3 - 256, p = 1; 1×1 - 256 (skip to 5)	9×9 - 128, s = 2
conv 4	7×7 - 256, s = 2	3×3 - 256, p = 1	3×3 - 256, p = 1; 1×1 - 256 (skip to SPP)
conv 5		3×3 - 32	3×3 - 256, p = 1
conv 6		9×9 - 128, s = 2	
SPP	scales = 4	scales = 3	scales = 4
FC 1	128	128	64
classifier	linear - 6, sigmoid	linear - 6, sigmoid	linear - 6, sigmoid

Table 3: Top three neural architectures of MetaQNN for our task. Convolutional layers (conv) are parametrized by a quadratic filter size followed by the amount of filters. p and s represent padding and stride respectively. If no padding or stride is specified then $p = 0$ and $s = 1$. Skip connections are an additional operation at a layer, with the layer where the connection is attached to specified in brackets. A spatial pyramidal pooling (SPP) layer connects the convolutional feature extractor part to the classifier. Every convolutional and FC layer are followed by a batch-normalization and a ReLU and each model ends with a linear transformation with a Sigmoid function for multi-target classification.

four adaptive pooling operations such that the output width times height corresponds to $1 \times 1, 2 \times 2, \ldots 4 \times 4$. The fully-connected (FC) layer is defined by the number of feature outputs it produces. All convolutional and FC layers are followed by a batch-normalization and a ReLU layer.

Figure 7 shows graphical representations of the top three neural models of ENAS for our task. All of the ENAS architectures have seven convolutional layers followed by a linear transformation as defined prior to the search. We have chosen a visual representation instead of a table because the neural architectures (acyclic graphs) contain many skip connections that are easier to perceive this way. All convolutions have quadratic filter size and a base amount of 64 features that is doubled after the second and forth layer as defined by a DenseNet growth strategy with $k = 2$.

C.3. Transferring ImageNet and MINC features

We investigate transfer learning with features pre-trained on the ImageNet and MINC datasets for a variety of neu-

Transfer learning

Architecture	Source	Accuracy [%]	
		best val	bv-test
Alexnet	ImageNet	60.53	62.87
VGG-A	ImageNet	60.22	66.35
VGG-D	ImageNet	56.13	65.56
Densenet-121	ImageNet	54.71	57.66
Alexnet	MINC	60.06	66.50
VGG-D	MINC	61.47	67.14

Table 4: Multi-target best validation and best validation model's test accuracy for fine-tuned CNNs with convolutional feature transfer from models pre-trained on the MINC and ImageNet datasets.

ral architectures by using pre-trained weights provided by corresponding original authors. We fine-tune these models by keeping the convolutional features constant and only training the classification stage for 70 epochs with a cycled learning rate and other hyper-parameters as specified in the main body. Best multi-target validation and associated test values are reported in table 4. Although the pre-trained networks initially train much faster, we observe that transferring features from the unrelated ImageNet and MINC tasks does not help, it in fact hinders the multi-target defect classification task. We postulate that this could be due to a variety of factors like the task being too unrelated with respect to the combination of object and texture recognition demanded by our task. This observation matches previous work investigating transfer learning of object related features to texture recognition problems. In such a scenario, the authors of [3] find the need to evaluate feature importance and selectively integrate only a subset of relevant ImageNet object features to yield performance benefits for texture recognition and prevent performance degradation. We further hypothesize another possibility that the multi-target property of the task could require a different abstraction of features from those already present in the convolutional feature encoder of the pre-trained models. Further investigation of transfer learning should thus consider an approach that doesn't include all pre-trained features, selects a subset of pre-trained weights or employs different fine-tuning strategies.

C.4. Classification examples

In addition to the accuracy values reported in the main body, we show qualitative example multi-target classifications as predicted by our trained MetaQNN-1 model. We do this to give the reader a more comprehensive qualitative understanding of the complexity and challenges of our multi-target dataset. In order to better outline these challenges, we separate these examples into the following three categories:

1. Correct multi-target classification examples where all labels are predicted correctly.

2. False multi-target classification examples where at least one present defect class is recognized correctly, but one or more defect classes is missed or falsely predicted in addition.

3. False multi-target classification examples where none of the present defect classes is recognized correctly.

Corresponding images, together with ground-truth labels and the model's predictions are illustrated in the respective parts of figure 8. The few shown examples were picked to show the variety of different defect types and their combinations. Overall, the images show the challenging nature of the multi-target task. Whereas the majority of multi-target predictions are correct, the trained models face a number of different factors that make classification difficult. Particularly, partially visible defect classes, amount of overlap, variations in the surface, different exposure and illumination can lead to the model making false multi-target predictions, where only a subset of targets is predicted correctly.

C.5. Alternative dataset splits

Architecture	Multi-target accuracy [%]		Params [M]	Layers
	val	test		
Alexnet	63.50	62.94	57.02	8
T-CNN	63.87	63.00	58.60	8
VGG-A	65.33	61.93	128.79	11
VGG-D	63.76	62.50	134.28	16
WRN-28-4	59.75	55.56	5.84	28
Densenet-121	66.54	65.93	11.50	121
ENAS-1	67.71	66.31	3.41	8
ENAS-2	66.50	64.37	2.71	8
ENAS-3	65.66	65.81	1.70	8
MetaQNN-1	66.70	65.91	4.53	6
MetaQNN-2	65.25	64.82	1.22	8
MetaQNN-3	70.95	67.56	2.88	7

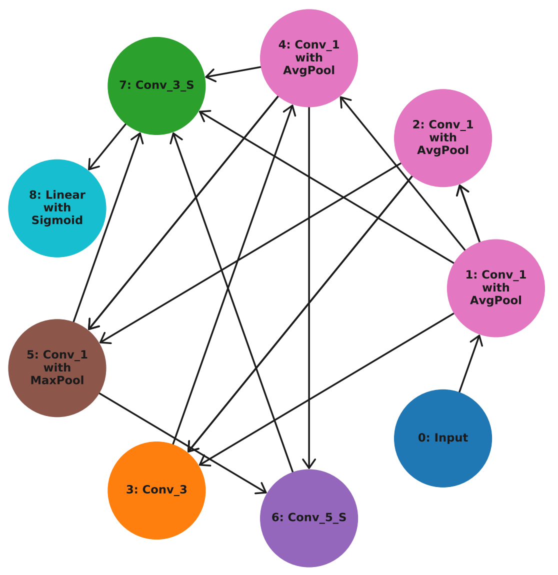
Table 5: Evaluation in analogy to table 2 of the main body, but on alternative dataset splits based on a per-bridge separation.

The final dataset presented in the main portion of the paper has been chosen to contain a random set of 150 unique defect examples per class for validation and test sets respectively. To avoid over-fitting we have further added the constraint that all crops stemming from bounding boxes from one image must be contained in only one of the dataset splits. The rationale behind this choice is to ensure a non-overlapping balanced test and validation set in order to avoid biased training that favors certain classes and report skewed loss and accuracy metrics.

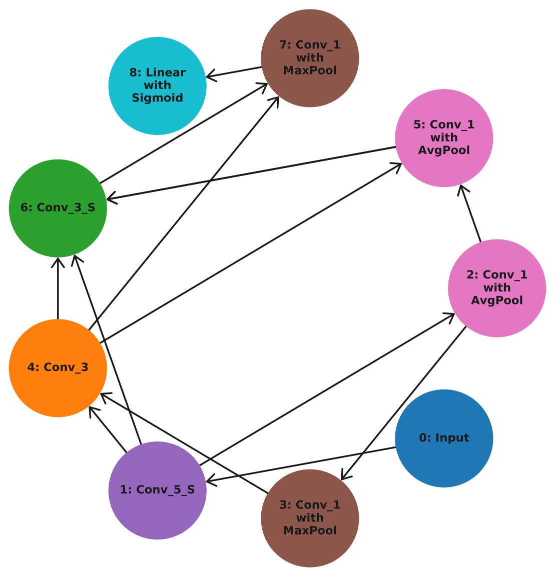
A different alternative way of conducting such a validation and test split is to split the data based on unique bridges. Such an approach however features multiple challenges that make it infeasible to apply in practice. In particular, not every bridge has the same amount of defects and not every bridge has the same amount of defects per class. Typically also defects of varying severity and overlap are featured (e.g. some have more early-stage cracks than exposed bars). The main challenges thus are:

1. Only a certain combination of unique bridges can yield an even approximately balanced dataset split in terms of class presence.

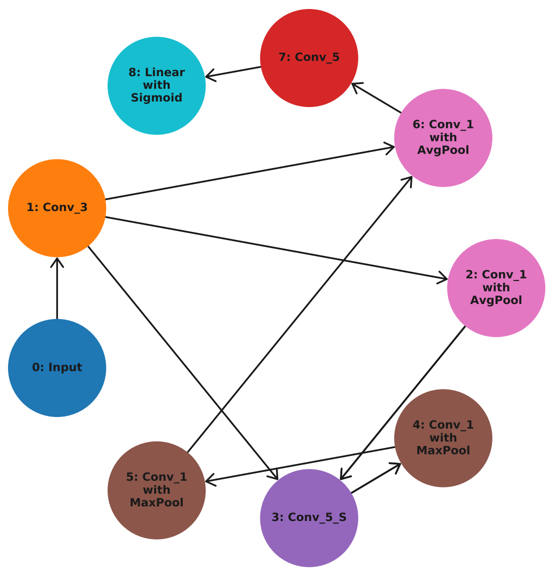
2. Creation of class-balanced splits relies on either excluding some of the highly occurring defects or leaving



(a) ENAS-1



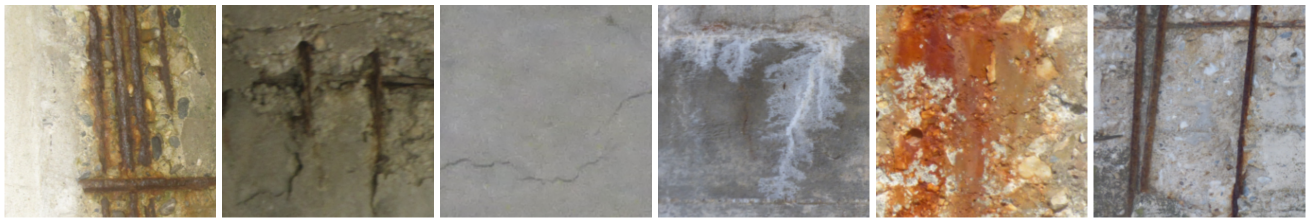
(b) ENAS-2



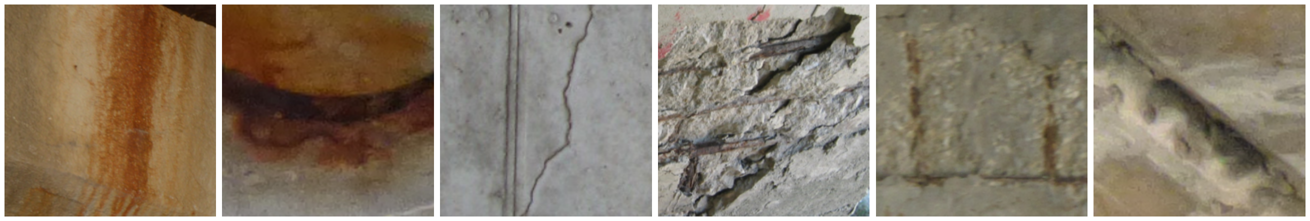
(c) ENAS-3

Figure 7: Top three neural architectures of ENAS for our task. Convolutions (conv) are denoted with quadratic filter size and a post-fix "S" for depth-wise separability. MaxPool and AvgPool are max and average pooling stages with 3×3 windows. ENAS uses a pre-determined amount of features per convolutional layer during the search and during final training uses a growth strategy of $k = 2$ similar to DenseNets. The amount of features per convolution is 64, doubled by the growth strategy after layers 2 and 4. The graph is acyclic and all connections between layers are indicated by directed arrows.

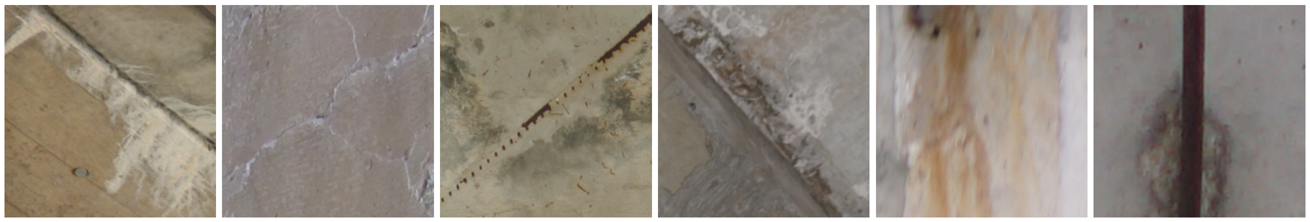
the dataset split only approximately balanced. The latter could result in training a model that favors a particular class and skewed average metrics being reported. The former can result in omitting particularly challenging or easy instances from the validation or test set and accordingly distorting the interpretation of the



(a) Correct multi-target classification examples from the validation set. From left to right: 1.) exposed bar, corrosion, spalling. 2.) spallation, exposed bars, corrosion and cracks. 3.) crack. 4.) efflorescence 5.) spallation and corrosion. 6.) spallation with exposed bars.



(b) False multi-target classification examples from the validation set where at least one present defect class is recognized correctly. From left to right: 1.) corrosion (predicted corrosion and efflorescence). 2.) corrosion (predicted corrosion, spallation and exposed bar). 3.) crack (predicted crack and efflorescence). 4.) spallation, exposed bar, corrosion (predicted spallation and corrosion). 5.) spallation, exposed bar, corrosion (predicted crack and corrosion). 6.) efflorescence (predicted efflorescence and crack).



(c) False multi-target classification examples from the validation set where none of the present defect categories is recognized correctly. From left to right: 1.) efflorescence (predicted background). 2.) crack (predicted background). 3.) exposed bar with corrosion (predicted background). 4.) efflorescence (predicted background). 5.) corrosion (predicted spallation). 6.) exposed bar (predicted crack).

Figure 8: Multi-target classification examples from the validation set using the trained MetaQNN-1 model.

model's accuracy.

3. Even when balancing the classes approximately by choosing complementary bridges, the severity of defects is not necessarily well sampled or balanced.

On the other hand, a bridge-based dataset split provides more insights with respect to over-fitting concrete properties such as surface roughness, color, context or, given that images at different bridges were acquired at different points in time with variations in global scene conditions. We therefore nevertheless investigate an alternative bridge-based dataset split that is based on three bridges for validation and test set respectively. The bridges have been chosen such that the resulting splits are approximately balanced in terms of class occurrence, albeit with the crack category being more present and the efflorescence class being under-sampled. The resulting accuracies should thus be considered with caution in direct comparison to the main paper.

Using this alternate dataset split we retrain all neural architectures presented in the main paper. We note that we have not repeated the previous hyper-parameter grid-search and simply use the previously obtained best set of hyper-parameters. In analogy, the meta-learning architecture search algorithms have not been used to sample new architectures specific to this dataset variant. The obtained final validation and test accuracies are reported in table 5. We re-iterate, that although we have coined the splits validation and test set, the sets can be used interchangeably here as no hyper-parameter tuning has been conducted on the validation set.

Obtained accuracies are similar to the experimental results presented in the paper's main body. We can observe that meta-learned architectures are not in exact previous order, e.g. MetaQNN3 outperforms MetaQNN1. However, meta-learned architectures still outperform the baselines and previous conclusions therefore hold. Due to the previously presented challenges in creation of an unbiased bridge-based dataset we therefore believe our dataset splits presented in the main body to be more meaningful to assess the models' generalization capabilities.

References

[1] Mark Everingham, S. M.Ali Ali Eslami, Luc Van Gool, Christopher K.I. I Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision (IJCV)*, 111(1):98–136, 2014. 2

[2] Tzutalin. LabelImg. *https://github.com/tzutalin/labelImg*, 2015. 2

[3] Yan Zhang, Mete Ozay, Xing Liu, and Takayuki Okatani. Integrating deep features for material recognition. In *International Conference on Pattern Recognition (ICPR)*, 2016. 6