MICROSCOPIC IMAGE SEGMENTATION FOR AUTOMATED INSPECTION OF SATELLITE COMPONENTS

David Bohlig*, Florian Leutert*, Florian Kempf*, David Möschwitzer*, Klaus Schilling*
* Zentrum für Telematik e.V., Magdalene-Schoch-Straße 5, Würzburg, Germany

Abstract

In the move towards image processing for Automated Integration and Testing (AIT) we explored defect detection on reflowed satellite PCBs using deep learning. To this end we utilized a convolutional neural network for semantic segmentation and subsequent instance segmentation for the detection of surface mounted devices, pollution and defects. A dataset of 16k labeled instances including devices, solder connections, solderballs, bridges and tombstoned components was created from our satellite fleet PCBs to train the network. The images were recorded using microscopes and industrial cameras and labeled using an active learning approach with human experts annotating the initial data. Then, the partially trained network labeled additional data with experts supervising the process and correcting predictions where necessary.We explored k-fold cross validation as well as dropout based uncertainty estimation for the prediction of samples that meaningfully extend our training data. Further we evaluated the benefits of the implemented procedures and the annotation speedup from the network assisted annotation. The resulting inspection system was successfully integrated into a human-robot collaborative workspace to increase its production efficiency.

Keywords

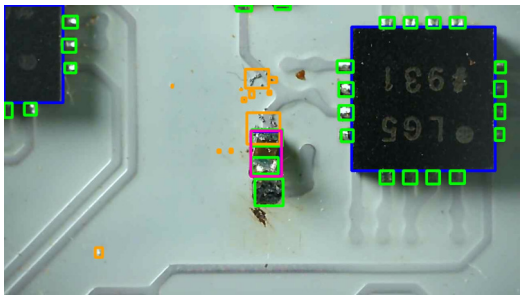
Deep Learning; Computer Vision; Optical inspection; Satellite assembly

1. INTRODUCTION

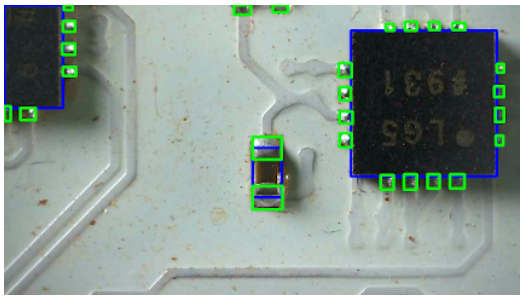
The system integration process of small satellite systems is challenging in respect to its requirements for manufacturing precision, fault tolerance and cleanness. As most satellites used to be built in small batch sizes down to a hand full of engineering models and a single flight model, the effort of manual inspection of system components was considerably lower than the engineering costs, and the expenditure of time small compared to the system integration process duration. With the rise of modern small satellite systems which are produced in larger batches manual inspection quickly becomes a considerable cost factor. Thus we developed an AI aided inspection module integrated into a human-robot-collaboration workspace that automatically detects and labels PCB faults on satellite subsystems. Our system was trained on a large dataset of PCB components and defects from different variants of our satellite subsystem models.

2. RELATED WORK

Various methods for component detection on PCBs include the detection of defects in copper lanes below silk screen [1], the detection of components through depth images [2], the detection of components with hand-crafted features [3] and object detection of various components in high-resolution wide-angle images [4]. Especially the FICS-PCB Dataset [5]



(a)



(b)

FIG 1. Automatic detections of two ICs (blue) with their solder connections (green) and two capacitors on the same PCB version. (a) This capacitor is tombstoned (pink) with surrounding solderballs (orange). (b) Correctly reflowed capacitor with surrounding flux and solder connections.

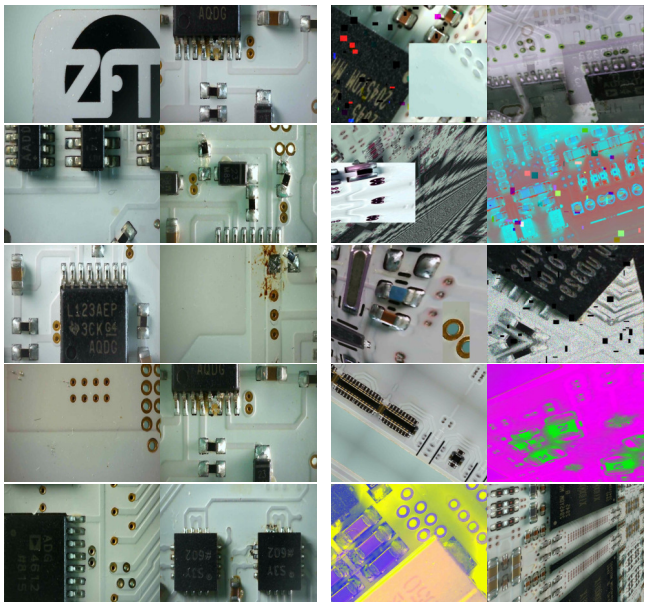
doi: 10.25967/570347

is tailored towards high-level inspection of PCBs with object detection methods for identifying different component types, but contains only images of successfully reflowed boards and no information about defects or solder joints. This enables a system to identify missing or additional components for fraud prevention and hardware security evaluation, but does not enable fault detection during AIT. While there are many semantic segmentation approaches with language models and end-to-end instance segmentation models [6] [7] [8] [9] [10] [11], we decided to use a modernized convolutional architecture [12] as they are a mature architecture with a well established design and training process. ConvNets have the additional advantage of being well suited for the detection of small scale objects like solderballs that often only have a visible size of a few pixels.

3. SYSTEM DESCRIPTION

We developed a visual inspection system using a semantic segmentation network to create instance segmentations with a classical computer vision algorithm and corresponding bounding box predictions from the detected instances. The deep learning step of our pipeline utilizes ConvNeXt layers in a UPerNet [13] configuration with the layer and stage ratios differing slightly from the original approach. We initially tested a variety of different architectures including UNet [14], and variations of it with ConvNeXt layers making up its backbone and additional skip-layers following the approach in [15], but found them to generate less precise predictions or not generalize as well as the UPerNet described in the original ConvNeXt paper. Our System is trained on a single RTX 2070 SUPER. As we were unable to identify a viable dataset we created one consisting of 744 images of our satellite PCBs recorded with a USB microscope and FLIR industrial cameras with 16.100 labeled instances. PCBs from our earth observation cubesat satellite models are the main source of inspection data. As the production process of future satellites in the fleet will be streamlined significantly the process of submodule inspection is a prime target for automation. Thus the design of our system focuses on the inspection of small satellite modules with a width of less than 15cm. Although the collected data comes from small boards the inspection hardware in figure 6 can be utilized for subsystems of a length of up to 40cm and a width of 20cm. Although we took care to collect a range of different satellite PCBs our selection was limited to in-house options and thus measures were taken to extend the distribution covered by the collected instances. To prevent overfitting we created a data augmentation pipeline including the following methods:

- global geometric augmentations: zoom, rotations, affine transformations, horizontal and vertical flipping



(a) (b)

FIG 2. (a) Shows a random selection of 10 unaltered PCB images. (b) Shows a random selection of 10 augmented samples.

- pixel space augmentations: saturation, brightness, color channel inversion, color channel shuffle, additive and multiplicative white noise
- localized augmentations: gaussian blur, coarse dropout, channelwise coarse dropout
- sample interpolation: Mixup and Cutmix [16] [17]

We do not shift the input data in xy-direction as convnets are translation equivariant. During the initial experimentation phase we did however train transformer based networks [18] [19] which are not shift equivariant, and were trained with and without shift augmentation. The system was deployed to the human-robot-collaboration workspace seen in figure 6 (a) to help with the PCB inspection and reworking process. Figure 2 shows the difference between un-augmented images in (a) and the randomly augmented samples in (b). The augmentations alter the size, color and spatial relation of the input instances so the network learns more relevant features

4. SEGMENTATION NETWORK

The chosen network architecture seen in figure 3 consists of a UPerNet with a stage ratio of [3x96, 3x192, 27x384, 3x768] for the input feature pyramid layers and a stage ratio of [2x96, 2x192, 2x384, 2x768] for the upsampling layers. The feature pyramid utilizes ConvNeXt style convolutions to extract the PCB features and 2x2 strided convolutions with ReLu activations and a 2x2 kernel for downsampling. All the downsampling steps are succeeded by a normalization layer. The pyramid upsampling layers consist of the same ConvNeXt layers behind 2x2

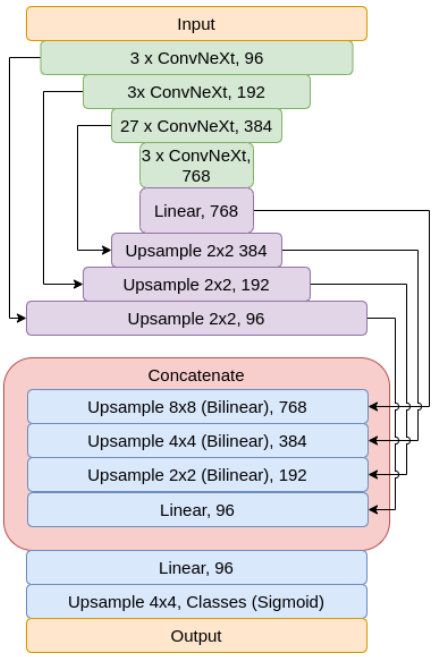


FIG 3. The network is identical to the semantic segmentation part of a UPerNet with a modernized ResNet as a backbone.

transposed convolution layers. After each upsampling step the correspondingly sized feature pyramid layer is concatenated to the upsampling output. Figure 3 shows the network architecture. Before concatenation to the upsampling outputs the feature maps are linearly projected to the upsampling layers channel depth. The ConvNeXt layers have depthwise kernels of size 7x7 and inverse bottleneck layers of size four times the size of the given number of filters. The final "Upsampling (Bilinear)" blocks scale their inputs layer width and height by the specified factors. They are concatenated and linearly projected to a 96 channel feature map of a quarter of the output resolution. Finally the feature map is upsampled by a factor of 4 and projected to the output class channel number by a 4x4 transpose convolution layer. We experimented with replacing the ConvNeXt layers partially or fully with an equal number of transformer layers, but did not notice a performance increase.

We use a soft-focal-loss [20] formulation to enable training on interpolated images with soft class labels and to allow the detection of small objects like solderballs. In samples where most of the input image consists of PCB background with only a small number of pixels belonging to the predicted class, the use of a loss function that empathizes rare classes over frequent ones is required for the model to train effectively. Because the default focal loss does not enable the evaluation of soft labels we decided to use the modified version from equation (3) to facilitate the mixup augmentation technique. On its own the cross-entropy in equation (1) can handle soft labels but will neither increase the weight of rare classes nor correct for skewed foreground-background distributions. The class weights are added by the γ-factor in equation (2). Finally equation (3) adds a class density estimation $\alpha_0 + y(\alpha_1 - \alpha_0)$ to extend the cross-entropy to the soft-target focal loss.

Training Parameters	
optimizer	Adam
epochs	1000
batch size	4
learning rate	1.0e-4
gradient clipping	1.0
β_1/β_2	0.99/0.999
Augmentation Probability	
geometric	100%
pixel space	33%
localized	33%
interpolation	33%

FIG 4. The settings for the network training process

$$(1) \quad CE(y,p) = -y \cdot \log(p) - (1-y) \cdot \log(1-p)$$

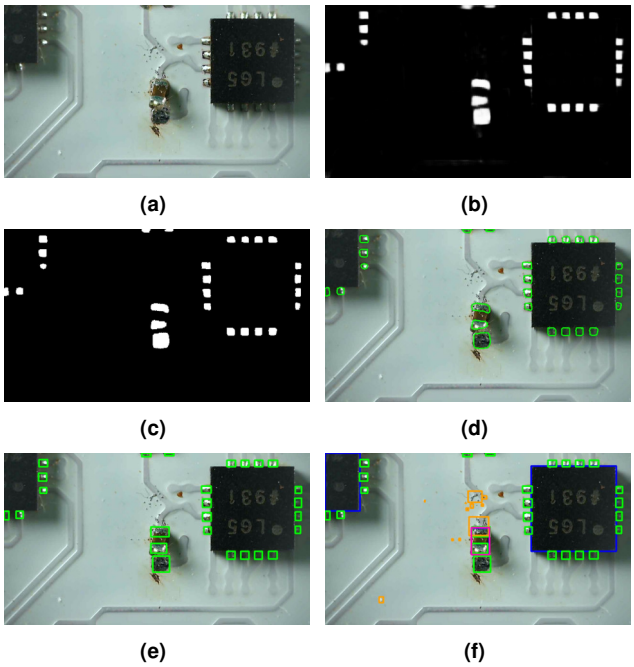
$$(2) \quad FL(x,p) = |y-p|^{\gamma} \cdot CE(y,p)$$

$$(3) \quad SFL(x,p) = [\alpha_0 + y(\alpha_1 - \alpha_0)] \cdot FL(x,p)$$

To make the evaluation of the cross entropy $CE(y,p)$ stable the value of p is clipped to [1.0e-7, 1.0-1.0e-7] during training to prevent its $\log$ from becoming undefined.

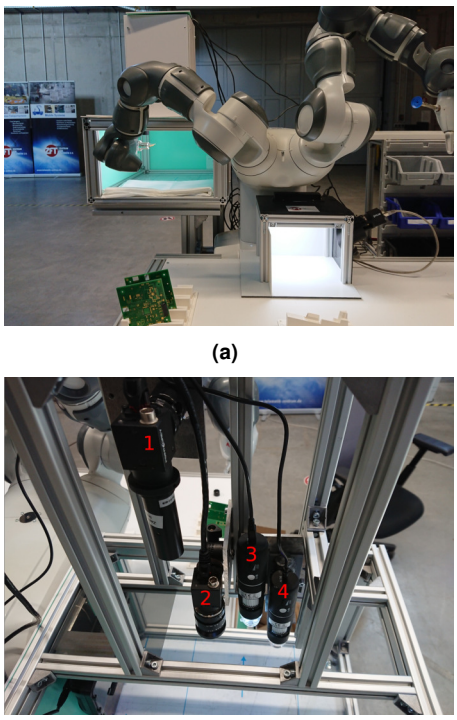
5. INSTANCE SEGMENTATION

Instances are detected in the semantic output by applying a class-wise probability threshold to create a binary mask and successive extraction of clusters via contour following. The minimum and maximum coordinates of the contour polygons define the bounding rectangle. This approach yields a result consisting of the pixelwise instance segmentation of an object, the object boundary in polygon form from the contour detection and the bounding box. Overlapping instances of the same class will be merged. While this would be a significant drawback in other applications our dataset contains very few examples like this. Although in such cases a dedicated instance segmentation network might yield better results the training of such a network on medium to high resolution images exceeds the capacity of our training hardware. To successively extract the contours we are employing border following algorithms implemented in OpenCV [21] [22]. The extracted contours are then used to crop the detected instance from the pre-threshold prediction and to create the bounding boxes by searching the minimum and maximum coordinates of the contour polygon. For each object its detection probability is given by the mean of the cropped probability mask.



(a) (b) (c) (d) (e) (f)

FIG 5. (a) The input Image is scaled to a resolution of 640x480. (b) The generated segmentation mask for the solder connections. (c) The binary mask for contour detection. (d) The detected polygonal contours . (e) The bounding rectangles of the given contours. (f) The combined prediction results after all masks are evaluated.



(a) (b)

FIG 6. (a) Image aquisition station using a robot for positioning of the components to be inspected. (b) The four integrated cameras: (1) High resolution macro camera, (2) High resolution wide angle camera, (3) Low magnification microscope, (4) High magnification microscope.

The process is illustrated for the detection of a set of solder connections on a faulty PCB in Figure 5. Tile (f) of Figure 5 shows the final detections after the segmentation masks of all classes have been evaluated.

6. INSPECTION SYSTEM

In the collaborative workspace seen in figure 6 (a), the manipulator robot automatically grasps PCBs from a carrier tray, and positions them under the corresponding inspection cameras. The generated images are then transferred to a server to be analyzed or used as training data. The acquisition process for each PCB can automatically be run seperately for each camera. Each run yields 494 images of which 364 are microscope images from camera (4) in figure 6 (b). A complete scan of a board with all cameras takes a duration of less than 5 minutes. Due to the very precise positioning needed to place the PCB in the focal point of the individual cameras, as well as to be able to inspect only certain points-of-interest on the boards, the robot-camera system is calibrated using printed glass calibration patterns, allowing for localizing features with respect to the robot coordinate system and placing boards accodingly under individual inspection cameras. The same system used during training is later employed for inspection during assembly. If a PCB is classified as polluted or faulty during inspection, the production process can adaptively be altered to either allow the human operator to clear the defect, or to remove the faulty component from the integration process. This optical inspection step thus improves production reliability by detecting potential defects already during assembly. Detected faults can be displayed to the human operator and also logged in a production database, so the whole production network can be improved during future iterations. Because the same system is used for inspection and dataset collection a faulty detection that is noticed by an operator will be stored so the production can be improved.

7. DATA LABELING

Our training samples were created in an iterative process of manual annotation, automated label generation with a pre-trained network and manual label correction. During the initial labeling phase 245 segmentation maps were created using the tool "LabelMe" [23] to kickstart the active learning procedure:w

1 Select data with out-of-fold prediction

- Create N splits of the training dataset without validation data
- Train instances of the network, one for each split
- Predict the segmentation maps on previously unseen data with each network

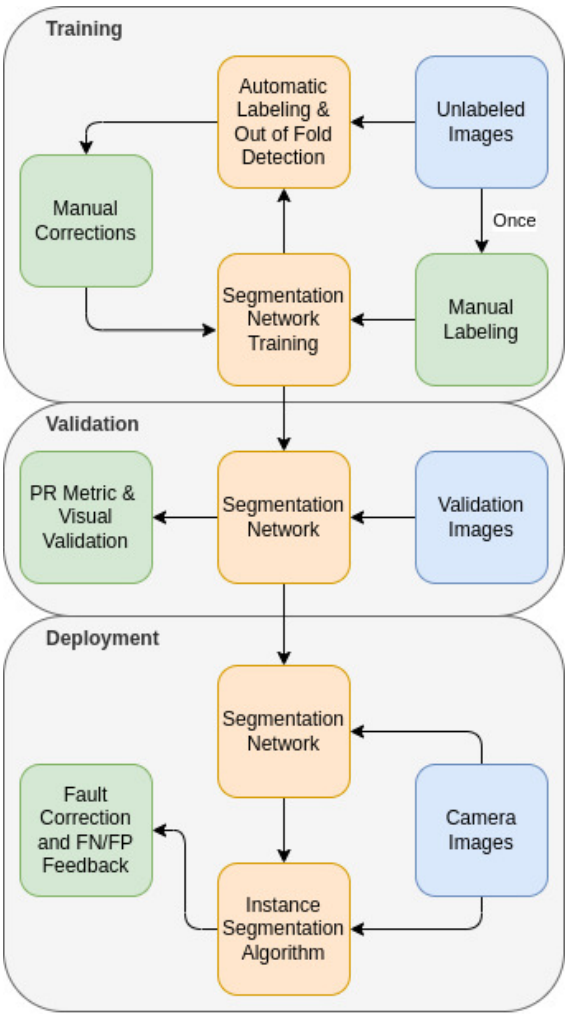


FIG 7. The training, validation and deployment process. During the training phase the network is queried to create labels for new data to assist in its own training. Blue blocks are data sources. Yellow blocks are our developments. Green blocks are manual tasks.

- Calculate the per-class-deviation between the predicted segmentation maps
- Sort the unseen data by its deviation and select the M images that have the highest error in each of the C classes

2 Create labels from predictions and correct manually

- Evaluate the current best Network on the CxM unlabeled images and create label polygons in the LabelMe format
- Load the images and predicted labels into LabelMe and correct errors
- Add the corrected images to the Dataset and go to step **1**

This process requires that the labeling takes place after each training of the N model instances or that the individual splits be extended online by new data when it becomes available. In Step 1.e the new samples are selected per-class as different classes will have

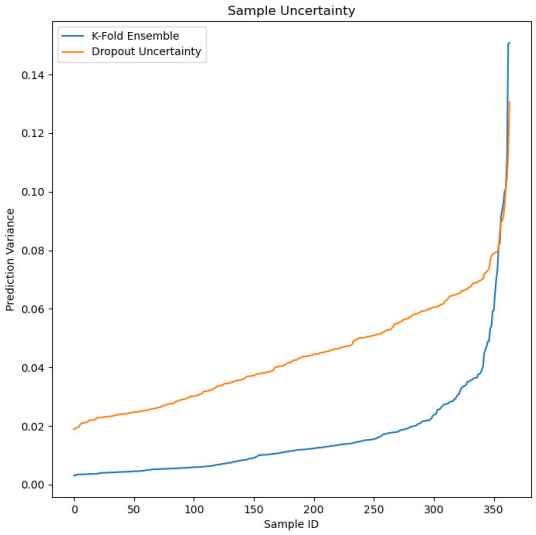


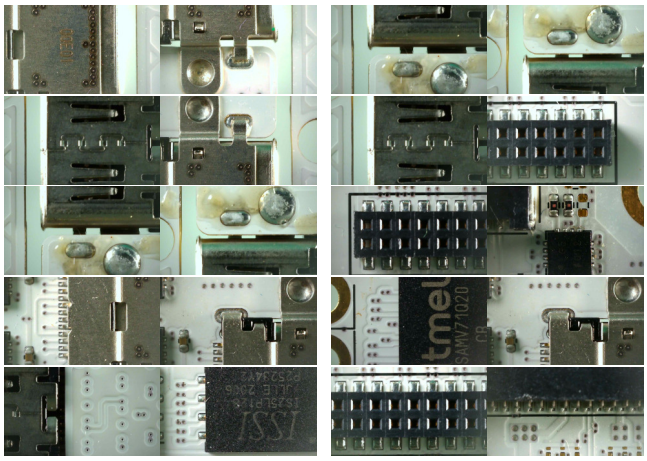
FIG 8. Blue: Variance curve of the out-of-fold prediction with $k = 3$. Orange: Variance curve of the dropout uncertainty for 1% dropout and 10 samples.

a different magnitude of corresponding pixels per image, leading to an imbalanced dataset with samples for classes that typically cover large image areas being overrepresented. Using this procedure we annotated a total of 2.600 components, 9.616 solderpads, 144 bridges, 3.657 solderballs and 60 tombstones. These instances were labeled with 11.240 rectangles, 3.929 polygons and 931 circles for a total of 16.100 instances. Although there are plenty of instances to train on we only acquired images from 21 satellite PCBs, giving us a limited assortment of mostly white solderscreens and comparably few electrical components to work on. We acknowledge the fact that the PCBs inspected by us are covering only a limited subset of the available PCB options. Nonetheless we expect the extension of our system to new types of boards to be a trivial task.

Many solderballs and imperfections are only visible under high magnification and thus the microscopic images are suited best for an in-depth inspection of PCBs. As the runtime of the inspection process can be reduced greatly by inspecting fewer images from the cameras with lower magnifications we aimed to extend its use extend their use during inspection. Once a defect is detected the analysis can be interrupted to transfer the part into the correction process. To facilitate this process we made sure around 10% of the labeled instances come from the lower magnification cameras. Anecdotally, the corresponding images are only slightly out of fold if the network is trained purely on high-magnification images when strong data augmentation is applied. To extend the usability of the network it was thus trained mainly on on images from the strongly magnifying microscope

	image #1	image #2	image #3
# Labels	43	58	22
min:sec (M)	3:30	3:17	1:03
s/label (M)	4.9	3.4	2.9
min:sec (A)	1:08	2:07	0:39
s/label (A)	1.6	2.2	1.8
	image #4	image #5	image #6
# Labels	53	75	76
min:sec (M)	3:03	5:22	4:22
s/label (M)	3.5	4.3	3.4
min:sec (A)	1:18	3:48	4:32
s/label (A)	1.5	3.0	3.6
	image #7	image #8	image #9
# Labels	35	31	28
min:sec (M)	2:10	2:20	1:36
s/label (M)	3.7	4.5	3.4
min:sec (A)	1:59	1:12	1:50
s/label (A)	3.4	2.3	3.9

TAB 1. Timings of the annotation process (M) manual, (A) assisted.



(a) (b)

FIG 9. The top 10 candidates from a new PCB. Starting with the top-1 from the upper left to lower right (a) shows the top out-of-fold candidates and (b) shows the top uncertainty-sampling candidates. The pictures mostly show microscope images of a USB connector and a pin connector. The training data contains only few examples of these components, making them good candidates to extend the dataset.

and with few images from the other three cameras. This allowed us to detect solderballs at the second stage of the camera inspection, cutting the runtime of the process by 74% when all 364 microscope images are skipped.

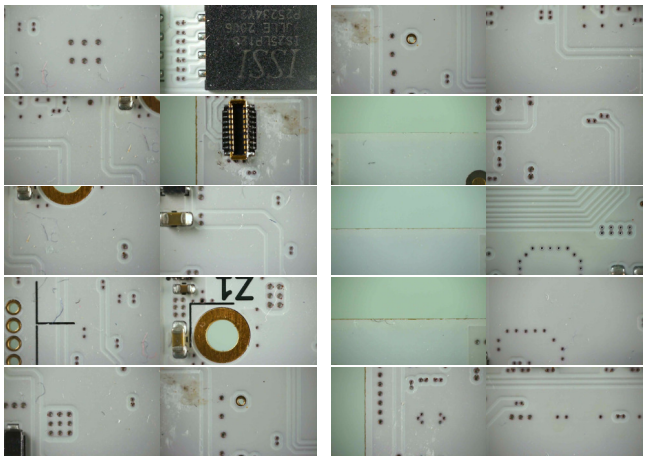
8. EVALUATION

8.1. Assisted Labeling

In order to test the effectiveness of our assisted labeling algorithm we labeled a subset of nine images twice, once with the pre-trained network and once fully manually. The images were labeled so that the assisted annotation always preceded the manual labeling. This way the annotator had an advantage on the full-manual labeling run as they have already seen the sample before. Thus these timings can be viewed as a lower bound for the expected speedup. The images contained 421 individual instances, 83 components, 270 solderpads, 5 bridges, 62 solderballs and one tombstone. From Table 1 we calculate that a single sample is labeled in an average of 3.78 ± 0.82 seconds per polygon during the fully manual phase and 2.64 ± 0.77 seconds per polygon during the assisted phase. This constitutes a speedup of 30% of the assisted labeling over the manual annotations.

The effect of the image selection process in the labeling procedure is harder to exemplify. We recognize out-of-fold prediction as inherently useful to identify novel datapoints that meaningfully extend the sample distribution in a similar way that uncertainty estimation can. Ensemble estimation, of which the k-fold training in our system is a special case, can also be used to estimate model uncertainty [24] [25]. In figure 8 we identified a power-law behavior of the calculated loss in that 5% of the novel samples were responsible for 50% of introduced variance. The difference between samples on the high end of the spectrum and the low end is more pronounced in the out-of-fold prediction curve, separating the best predictions more clearly from the rest. If the soft-focal-loss is evaluated on the ensemble prediction, the mean loss of the lower two thirds of samples that were evaluated is 8.1e-4 and thus compares favorably with the loss value of the training validation loss of 7.7e-4, suggesting that the samples exhibit only residual random noise and cannot be distinguished by our method. We compared the process to uncertainty prediction with random dropout [26] and found they yield comparable results. The uncertainty estimation was implemented by randomly dropping 1% of activations from the Input layer and the first two downsampling layers, evaluating the network 10 times and tracking the standard deviation of the output.

Figure 8 shows the resulting sorted samples. The mean index distance between identical samples in the two sorted lists of 364 images is 42 and 26 for the top 50 images, indicating the correlation between the sorting methods. A uniform random assignment would be expected to yield a mean index distance of $E(X) = 364 \cdot \int_0^1 (x^2 - x + \frac{1}{2})dx = 121.\overline{3}$. A comparison of the generated annotation candidates can be viewed in figure 9. The selected images all show components with very few training examples. The dataset previously contained only a single USB



(a) (b)

FIG 10. A selection of the top 20 of the solderball class (a) shows the top out-of-fold candidates and (b) shows the top uncertainty-sampling candidates. As expected the images show mostly pollution, which is frequently misidentified as solderballs.

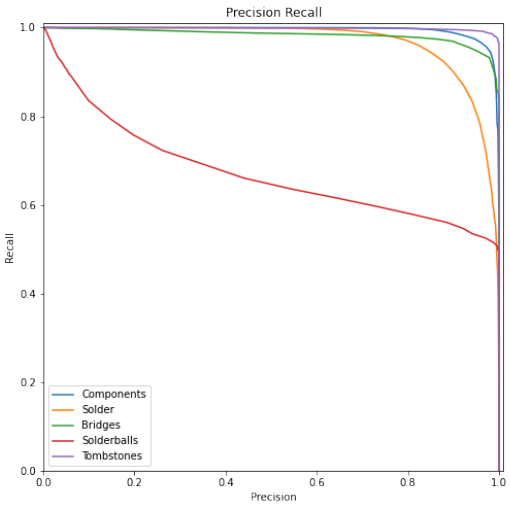


FIG 11. Precision-Recall plot for the final network

connector and pin connector of the detected variants, both with a different appearance and size from the newly selected instances.

The nature of the k-fold-differencing process introduces a potential risk. The sorting key $e_{i,c}$ for sample x_i in the sorted list of class c is calculated to be:

$$e_{i,c} = \sqrt{\sum_{a}^{N} \frac{(\mu_c - f_{\Phi_a}(x_i)_c)^2}{N}} \tag{4}$$

for f_{Φ_a} network instances with network weights Φ trained on the N splits of the dataset, μ_c the mean prediction value for class c and $f_{\Phi_a}(x_i)_c$ the prediction of the network on sample x_i. If the unseen data contains samples that introduce the same error into the predictions of all trained instances, the sorting key is low and the samples will be recognized as explainable by the training data and thus not be selected as a candidate for labeling. Although we were unable to observe such behavior it is a mathematical property of the selection process that must be taken into consideration when designing larger labeling tasks where the unlabeled data can not be skimmed for erroneous predictions. Because we could not identify erroneous candidates in our test data we suggest that this process is nonetheless an effective method for selecting new samples to annotate. A clear benefit of evaluating the k-fold cross-validated networks on new data is the speed up over the dropout method. The evaluation of the three ensemble networks for uncertainty prediction is 70% faster than evaluating the same network ten times with dropout.

8.2. Network Accuracy

From the Precision-Recall (PR) metric depicted in figure 11 it is clear that the model can identify most classes well. The only unexpected behavior is the low PR curve for the "Solderball" class. Upon inspecting a set of unlabeled test images in figure 10 the issue can be explained. The chosen focal-loss results in a strong weight for this class during training, as positive samples often contain $> 99\%$ background pixels. This results in a network that is highly sensitive to solderballs and generates a lot of false-positive samples on non-solderball objects. Additionally the solderball class is the least precisely labeled class in the dataset, with annotators frequently overlooking samples or labeling dust particles or solder-flux residue. As dust and grainy surfaces are often falsely identified as solderballs by the labeling workers the network will also be overly sensitive to these features. Figure 10 also shows very clearly that the active learning procedure picked up on this issue, suggesting images with dirty surfaces as labeling candidates. Because solderballs are frequently only single pixels in size the dropout process can produce features reminiscent of solderballs, misleading the segmentation network into identifying the dropout itself as solderballs. As a result the dropout method is less useful for the solderball class uncertainty. Even though the instance segmentation algorithm typically filters those detections through the class-specific threshold the samples will likely increase the model precision. We were able to produce a false positive rate of 1.7% in our demonstration system while maintaining a detection rate of 99.3% for all classes except the solder balls. Including the detected pollution on the PCBs the solderball class produces 23.7% false positives at a detection rate of 75.4%.

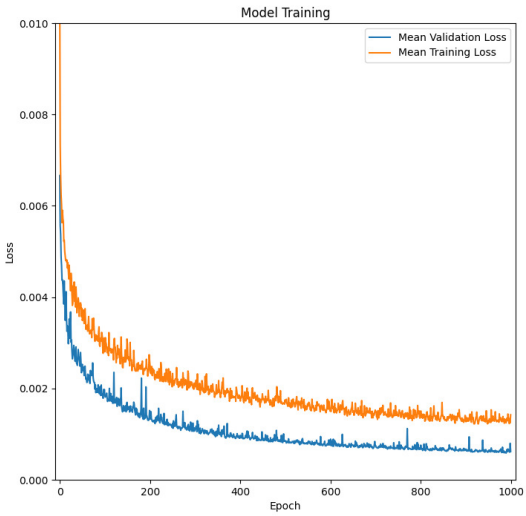


FIG 12. Mean training and validation loss values of the k-fold validated training. The validation loss values are calculated without image augmentations and as a result are slightly lower.

There are no signs of overfitting in Figure 12. It shows neither increase nor a stagnation of the validation loss, indicating that the applied data augmentation prohibits this kind of behavior up to the trained accuracy. We determined that only the geometric augmentations, rotation, zoom and shear are needed to prevent overfitting. We hypothesize this is due to the non-repeating nature of the input image features under transformation creating the necessity for transformation invariant kernels to be learned by the network. Under random affine transformation samples with distance and orientation dependent labels will create continuous distributions of training data from a discreet number of samples. As almost all of the labels in the dataset depend upon some orientational relationship, for example between solder pads and components, solder pads and solder balls or bridges and solder pads, there will be many continuous sample spaces for the network to learn.

9. CONCLUSION

We demonstrated a practical approach for data selection using k-fold cross-validation resulting in a novel dataset of PCBs with object segmentations for common defects. A comparison of uncertainty sampling and cross-validation approaches indicates the process is similarly useful and produces interesting new samples for the model training. Assisted annotation was tested during the labeling process and provided a significant speedup of the process. While these approaches are individually useful the integration of uncertainty estimation via dropout with both the cross-validation procedure and the assisted labeling process will be an interesting system to explore. It may not just be able to identify hard samples, but also pre-sample the easy parts of any hard sample image to further speed up the annotation. Another open question is the relationship between the predicted uncertainty of the unlabeled samples and the evaluated model type and size. If the sample space exploration can be facilitated by a smaller model that can be trained with less effort the active learning procedure might be significantly accelerated. The resulting system was tested in a human-robot collaborative workspace and determined to be sufficient for our use-case. Despite its minor limitations, the trained segmentation network was successful in detecting surface defects and PCB pollution on new satellite components. It was integrated into our AIT process to improve its reliability and efficiency.

Contact address:

david.bohlig@telematik-zentrum.de

References

[1] Prashant Malge and Nadaf S. Pcb defect detection, classification and localization using mathematical morphology and image processing tools. *International Journal of Computer Applications*, 87, 01 2014. DOI: 10.5120/15240-3782.

[2] Dongnian Li, Changming Li, Chengjun Chen, and Zhengxu Zhao. Semantic segmentation of a printed circuit board for component recognition based on depth images. *Sensors (Basel, Switzerland)*, 20, 09 2020. DOI: 10.3390/s20185318.

[3] Wei Li, Bernhard Esders, and Matthias Breier. Smd segmentation for automated pcb recycling. In *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, pages 65–70, 2013. DOI: 10.1109/INDIN.2013.6622859.

[4] Chia-Wen Kuo, Jacob Ashmore, David Huggins, and Zsolt Kira. Data-efficient graph embedding learning for pcb component detection. *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 551–560, 2019.

[5] Hangwei Lu, Dhwani Mehta, Olivia P. Paradis, Navid Asadizanjani, Mark Mohammad Tehranipoor, and D. Woodard. Fics-pcb: A multimodal image dataset for automated printed circuit board visual inspection. *IACR Cryptol. ePrint Arch.*, 2020:366, 2020.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.

[7] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer V2: scaling up capacity and resolution. *CoRR*, abs/2111.09883, 2021.

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *CoRR*, abs/2005.12872, 2020.

[9] Xinlong Wang, Tao Kong, Chunhua Shen, Yuning Jiang, and Lei Li. SOLO: segmenting objects by locations. *CoRR*, abs/1912.04488, 2019.

[10] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic, faster and stronger. *CoRR*, abs/2003.10152, 2020.

[11] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[12] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *CoRR*, abs/2201.03545, 2022.

[13] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. *CoRR*, abs/1807.10221, 2018.

[14] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. (available on arXiv:1505.04597 [cs.CV]).

[15] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters - improve semantic segmentation by global convolutional network. *CoRR*, abs/1703.02719, 2017.

[16] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *CoRR*, abs/1905.04899, 2019.

[17] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020.

[20] Dong Wang, Yuan Zhang, Kexin Zhang, and Liwei Wang. Focalmix: Semi-supervised learning for 3d medical image detection. *CoRR*, abs/2003.09108, 2020.

[21] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[22] Satoshi Suzuki and KeiichiA be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985. DOI: https://doi.org/10.1016/0734-189X(85)90016-7.

[23] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image. *Int. J. of Computer Vision*, 77(1), 2005.

[24] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. 2016. DOI: 10.48550/ARXIV.1612.01474.

[25] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36(2):111–147, 1974.

[26] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. 2015. DOI: 10.48550/ARXIV.1506.02142.