mathematics

Article

Model-Assisted Labeling and Self-Training for Label Noise Reduction in the Detection of Stains on Images of Laundry

Tamino Huxohl [1,*] and Franz Kummert [1]

1 CoR-Lab, Bielefeld University; 33615 Bielefeld, Germany; franz@techfak.uni-bielefeld.de
* Correspondence: thuxohl@techfak.uni-bielefeld.de; Tel.: +49-521-106-12207

Abstract: In this work, the creation of a dataset labeled in a pixel-wise manner for the uncommon domain of stain detection on patterned laundry is described. The unique properties of images in this dataset—stains are small and sometimes occur in large amounts—led to the creation of noisy labels. Indeed, the training of a fully convolutional neural network for salient object detection with this dataset revealed that the model predicts stains missed by human labelers. Thus, the reduction in label noise by adding overlooked regions with the help of the model's predictions is examined in two different experiments. In the model-assisted labeling experiment, a simulation is ran where a human selects correct regions from the predictions. In the self-training experiment, regions of high certainty are automatically selected from the predictions. Re-training the model with the revised labels shows that model-assisted labeling leads to an average improvement in performance by 8.52%. In contrast, with self-training, the performance increase is generally lower (2.58% on average) and a decrease is even possible since regions of high certainty are often false positives.

Keywords: self-training; model-assisted labeling; salient object detection; surface defect detection; stain detection

check for updates

Citation: Huxohl, T.; Kummert, F. Model-Assisted Labeling and Self-Training for Label Noise Reduction in the Detection of Stains on Images of Laundry. *Mathematics* **2021**, *9*, 2498. https://doi.org/10.3390/math9192498

Academic Editors: Wolfram Schenck and Alaa Tharwat

Received: 1 September 2021
Accepted: 30 September 2021
Published: 5 October 2021

1. Introduction

Currently, supervised deep learning methods are the state of the art in many computer vision tasks such as salient object detection (SOD) [1–3] and semantic segmentation [4–6]. To achieve the best results, these methods require a huge amount of labeled data, whose creation is often expensive. This is an issue if deep learning methods are applied to uncommon tasks where labeled datasets are not readily and publicly available. Another problem is that labels can be noisy or may contain errors. For example, SOD and semantic segmentation require pixel-wise labeling, which can easily contain erroneous pixel labels at object borders. Correct and consistent labels, however, are particularly important because "the accuracy of a trained model heavily depends on the consistency of the labels provided to it during training" [7]. There are several different approaches to work with scarce and noisy labels which are outlined in the following paragraphs.

1.1. Weakly Supervised Learning

In weakly supervised learning, target outputs are learned from other, more cheaply created labels. Therefore, it works around the scarcity of labels by making the labeling process less time consuming or by using other available labels. For example, Khoreva et al. apply weakly supervised learning to semantic segmentation by using bounding box labels [8]. They use Multiscale Combinatorial Grouping [9] and a variation of GrabCut [10] to compute pixel-wise labels for each bounding box. Afterwards, a DeepLab architecture [11] is trained with the intersection of the pixel-wise labels computed by the two methods. Hsu et al. also achieved semantic segmentation from bounding box labels [12]. In contrast to [8], they assumed that bounding boxes are tight—each side of the bounding box touches the object—to formulate a loss function. This allowed them to train a Mask Region-Based Convolutional Neural Network (CNN) [13] in an end-to-end fashion. Lu et al. achieved semantic segmentation from image-level labels [14]. They performed a superpixel segmentation and assigned each image-level label to all superpixels. Subsequently, the superpixel labels were iteratively optimized by visual similarity, noise sparsity and by a machine learning model that predicted their labels. Wang et al. used image-level labels for SOD [15]. They re-modeled a CNN for classification as a fully convolutional network to preserve spacial information in a score map. The score map was upsampled through deconvolutions to the original input dimensions for the SOD. They implemented an iterative learning procedure which also includes self-training.

1.2. Considering Label Noise

To deal with erroneous labels, many methods incorporate errors as noisy labels into the training. They can be categorized into two non-exclusionary groups: 1. Noise is incorporated into the loss function to concurrently learn a noise model. 2. Before the training, noisy labels are detected and either sorted out or their noise is reduced. An example for the first category is shown by Bekker and Goldberger [16]. They present an expectation maximization algorithm to deal with noisy labels in image classification. In the expectation step, a neural network is trained with the current labels, while in the maximization step a noise model, which is a confusion matrix stating that class i is mistaken for j, is updated. Zhang et al. applied a similar algorithm to learn SOD from noisy labels generated by unsupervised saliency detectors [17]. In [18], Han et al. describe an approach to deal with noisy labels in a classification setting belonging to the second category. They find prototypes for each class based on the features computed by a CNN. Afterwards, labels are corrected through prototype matching. Luo et al. implement a noise reduction method in an SOD setting [19]. They evaluate if a pixel-wise label is correct by comparing the results of a classifier to the original image with an image where the labeled region is filled by PatchMatch [20] and with an image where the labeled region is uniformly filled with a color. Yi et al. apply noise reduction in a semi-supervised semantic segmentation setting [21]. They train a CNN on a small, fully labeled dataset as well as the class activation maps [22] of a classification network to propagate labels through a graph structure on superpixels. This way, denoised labels are computed for unlabeled images.

1.3. Self-Training

A different approach, which can be used to deal with erroneous as well as scarce labels, is self-training. In semi-supervised self-training, a model is trained on a small, fully labeled dataset [23]. Subsequently, the predictions of the model on unlabeled data are used as labels to train a new model. This method can also be used to adapt a model to a different domain. Usually, this procedure includes methods to reduce noise in the predictions of the model. For example, a kind of curriculum learning [24] can be used in which the self-training starts with objects where the model is certain in its predictions and then gradually continues to more uncertain, more difficult objects. Zou et al. use self-training in a semantic segmentation setting for domain adaptation [25]. They note that only considering objects predicted with high certainty leads to models entirely ignoring objects that differ widely between domains. Thus, they propose class-balanced self-training, in which certainty thresholds are applied for each object category separately. In addition, they integrate spatial priors, such as roads appearing in the bottom center of images, to select good predictions. In [26], class-balanced self-training is improved through smooth labels. Bagherinezhad et al. apply self-training to classification [7]. They notice that one-hot encoded labels are often noisy since multiple objects can be present in an image. This becomes an even larger problem if heavily cropped images are used for training. As a result, a completely different object than that labeled in the one-hot vector may be visible. Therefore, they use self-training to reduce noise in an iterative training procedure; the predictions of a previously trained model are used as smooth labels to train the next model.

1.4. Hybrid Approaches

Naturally, it is possible and common to combine the approaches presented above. In fact, most of the referenced sources do this, e.g., many of the methods incorporating label noise use an expectation maximization approach that could also be classified as self-training [16–18]. Furthermore, many self-training approaches try to reduce label noise in some way [8,25], for example, through the integration of prior assumptions or through the usage of weak labels. In addition, many weakly supervised methods also apply self-training approaches to improve their results [14,15].

1.5. Our Approach

In this paper, we examine the reduction in label noise in the uncommon domain of stain detection on images of patterned laundry. This task can be classified as a surface defect detection task since we deal with images of flat laundry. We apply SOD methods to this task for the following reasons: 1. Usually, stains are salient objects as they automatically stand out to the eye. 2. The application of SOD methods is common in surface defect detection [27–32]. The training of a fully convolutional network for SOD to a dataset created by us revealed that human labelers missed stains. As a result, we revised the labels with the help of the model's predictions. This allows us to research how the model's predictions can be used to improve noisy labels. The contributions made in this paper are as follows:

- Unique properties of the domain of stain detection on flat and patterned laundry are pointed out.
- The reduction in label noise with the help of predictions by a model is evaluated in two experiments:
    1. Model-assisted labeling (MAL) [33]: A human is simulated as selecting overlooked stains from the predictions of the model. This approach provides a baseline for autonomous approaches and shows that predictions can be used for semi-automated labeling.
    2. High Certainty: Regions predicted with high certainty are automatically incorporated into the labels. This approach conforms to dealing with noisy labels via self-training.

This paper is structured as follows: In the next section, the dataset, its labeling and its unique properties are pointed out. Afterwards, in Section 3, fine-tuning a model of SOD for the detection of stains is described. Subsequently, Section 4 describes the self-training experiments performed as well as how they are evaluated. Section 5 presents and discusses the results of the experiments. Finally, a conclusion is drawn in Section 6.

2. A Dataset for Stain Detection

We are concerned with the detection of stains on patterned laundry that remain after conventional washing in an industrial laundry shop. After the detection, laundry with stains that did not wash out can be sorted out, and washable stains can be eliminated through a localized washing procedure. Since, at least to our knowledge, no publicly available dataset of stains on images of laundry exists, we created a dataset ourselves. To this end, we borrowed laundry from a nearby industrial laundry shop. The laundry mainly consisted of bedclothes with simple line patterns (double sheets, comforter covers, pillowcases) as they are used by hotels and hospitals.

We took pictures of the laundry with a specialized image acquisition apparatus shown in Figure 1. The apparatus consists of a conveyor belt, illumination and two line scan cameras. The conveyor belt transports the laundry in a flat state. It is about two and a half meters wide so that images of large pieces of laundry can be acquired. Above the conveyor belt, there is a beam on which the illumination is mounted. It consists of two LED stripes covering the whole width of the conveyor belt. They illuminate the laundry from above with white light as well as UV radiation to improve the visibility of certain kinds of stains.

Two line scan cameras are mounted within another beam further above and capture the conveyor belt through a slot in the center of the beam holding the lighting. The acquisition rate of the cameras is controlled by a sensor that measures the speed of the conveyor belt and adapts the rate accordingly. Both cameras record lines with a width of 2048 pixels, which results in one pixel corresponding to an area of about 0.25 mm^2. Unfortunately, the placement of the cameras inside the carrier led to a blind spot at the center of the conveyor belt (this is visible as the black borders of all example images in Figure 2). This is illustrated in Figure 1 by the dotted orange lines that show the field of view of each camera. This issue could only be fixed after the dataset was recorded. As a result, we plan to record another dataset in the future.

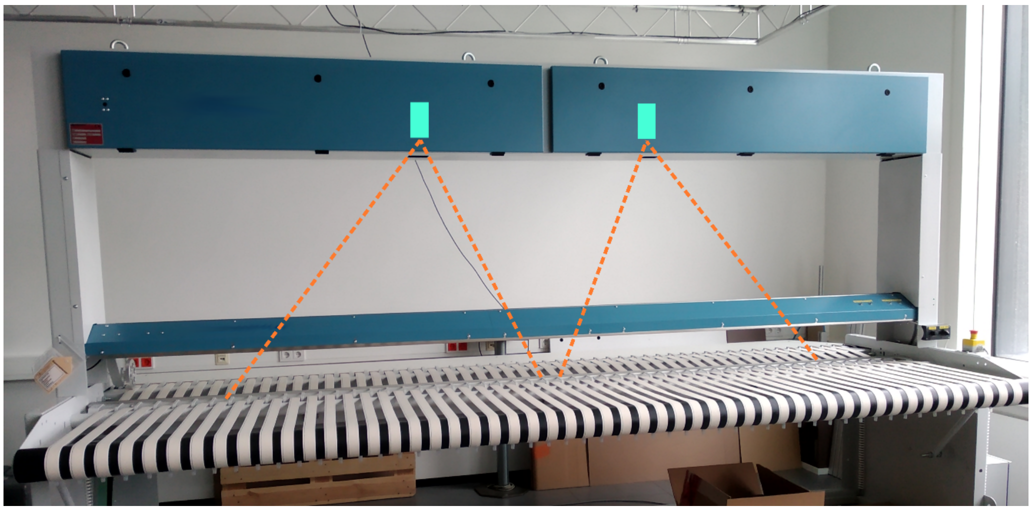


Figure 1. A photo of the image acquisition apparatus. The cyan rectangles show the positions of the line scan cameras. The dotted orange lines illustrate the field of view of each camera.

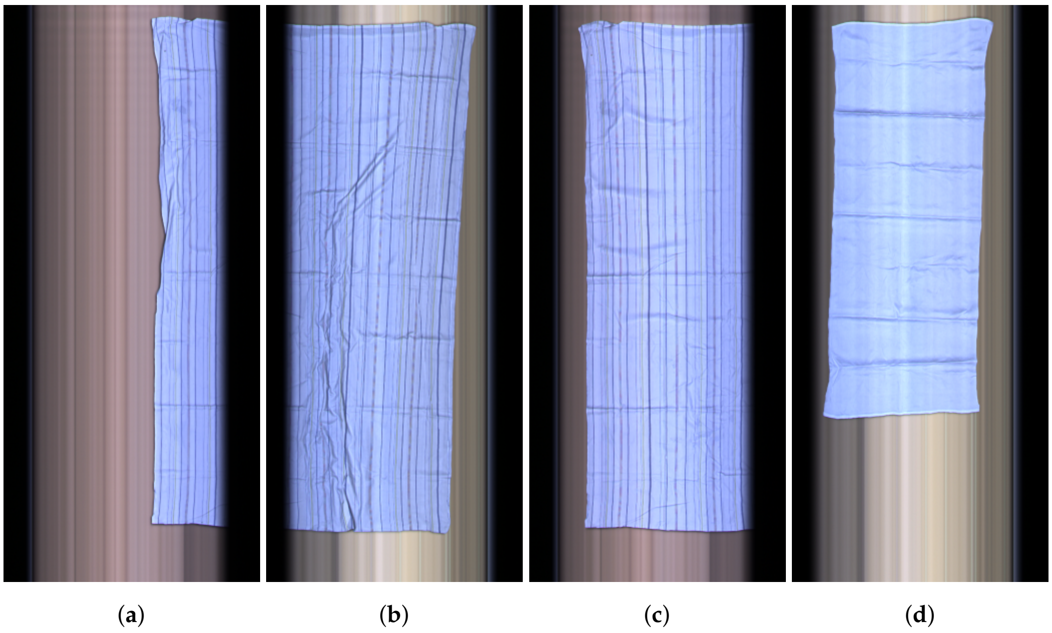


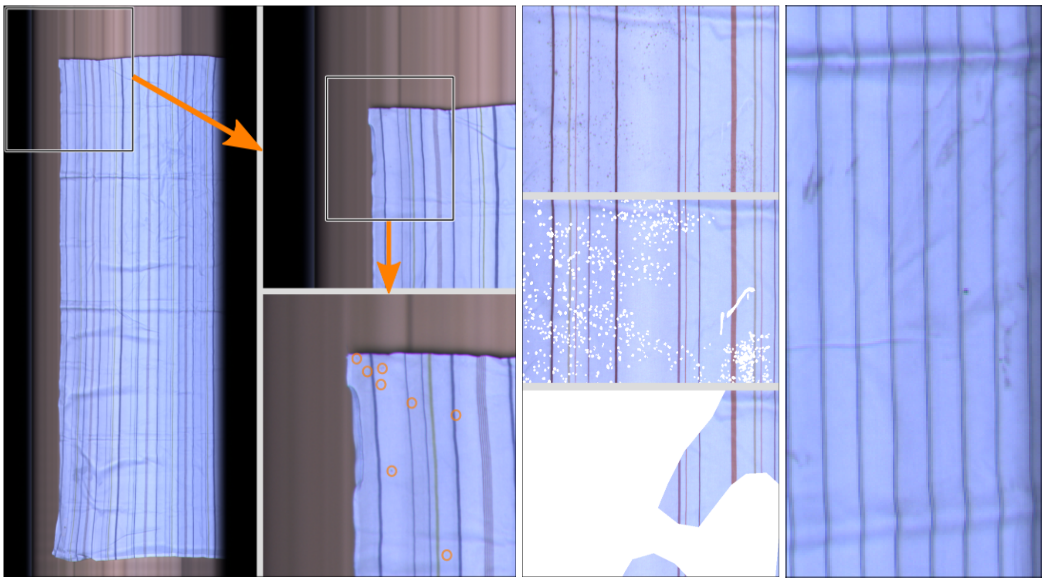
Figure 2. A few example images from our dataset. From left to right: An image acquired by the camera on the left (**a**). The corresponding image acquired by the camera on the right (**b**). An image of the same piece of laundry shifted further to the left (**c**). An image of a small piece of laundry fitting into the scope of a single camera (**d**).

Images were acquired by placing an item of laundry onto the conveyor belt and then turning the apparatus on. The apparatus automatically recorded a pre-configured number of lines, which was adapted in such a way that the piece of laundry was completely visible on the image. Thus, the acquired images have resolutions between 3048 $\times$ 2048 and 6548 $\times$ 2048 pixels. We acquired up to eight images for each piece of laundry: a single acquisition creates two images—one by each camera (see Figure 2a,b). Due to the blind spot in the center area, two acquisitions were conducted, shifting the piece of laundry to the left or right in between (compare Figure 2a,c). Additionally, we acquired images of the front and the back of each piece of laundry, doubling the number of images for each piece. Small items of laundry, such as pillowcases, fit into the scope of a single camera (see Figure 2d). Therefore, we only acquired the images of a single camera and we did not adjust the piece of laundry. Altogether, we acquired 1035 images of 141 pieces of laundry. Figure 2 displays a few examples.

Since we wanted to fine-tune a CNN for SOD for stain detection, we required a pixel-wise labeling of the images. Thus, three non-experts, including one of the authors, were tasked with creating the labels. Being non-experts should not be an issue with regard to the label quality, since stains are usually salient, meaning that they stand out to the eye. To perform the labeling, a custom tool was used in which stains could be marked by drawing their contours with the mouse. In addition, the tool provided a zooming capability so that small stains could be detected despite the huge size of the images. Originally, it was planned that every image would be labeled twice by two different labelers in order to compare the two labels as a way to prevent errors. However, this underestimated the effort of labeling large images in a pixel-wise fashion. As a result, every image was only labeled once. Moreover, a few difficulties arose during the labeling which differentiate this dataset from other datasets for SOD.

Fan et al. ascertained that most datasets for SOD do not include non-salient images, that salient objects are large and that there is a center-bias [34]. In contrast, the stains in our dataset are often small compared to the size of the image. This is illustrated in Figure 3a in which the stains are not clearly visible even when enlarged by a factor of four. The dataset does not have a center bias and some pieces of laundry contain a huge number of stains. As a result, we conducted a rough labeling (such as seen in the bottom of Figure 3b), instead of an accurate labeling (displayed in the middle of Figure 3b), because an accurate labeling of this many stains takes multiple hours for a single image. Furthermore, the borders of stains are often blurred, which is illustrated in Figure 3c. This makes it difficult to create an accurate pixel-wise label. Moreover, the folds on the laundry, created by being transported in a folded state and through the movement of the conveyor, create shadows and dark areas, which can be mistaken for stains.

Due to all of these properties, labeling the dataset is a difficult and time-consuming task. Thus, it is not surprising that our initially created labels $\mathcal{L}_i$ contained significant amounts of noise. We noticed that a CNN trained on the dataset (as described in the following section) detected stains that were missed during labeling. For this reason, the same three labelers revised all labels. To support this process, the initial labels and the predictions of the CNN were superimposed on the image. Regions from the predictions could be directly adopted into the label. These revised labels $\mathcal{L}_r$ contain 22,109 labeled regions, compared to 10,467 in $\mathcal{L}_i$, with an average of about 21 per image. These two labels $\mathcal{L}_i$ and $\mathcal{L}_r$ allow us to study how the predictions of a model can be used to reduce label noise either via self-training or by supporting human labelers.



(**a**) (**b**) (**c**)

Figure 3. Illustrations of properties that differentiate the collected dataset from others. (**a**) illustrates how large the images are in comparison to the stains/salient regions. (**b**) shows a piece of laundry with a large amount of stains. From top to bottom: the image, a precise annotation, an rough annotation. (**c**) shows examples of blurry stains.

3. Model Training

In this section, the general procedure of training a model for the created dataset is described. This training led to the discovery that the model detects stains missed in the initial labels. The training procedure as described here is embedded in the self-training experiments elucidated in Section 4. To speed up the training and to improve results with a medium-sized dataset without a great deal of variance in the data, a transfer learning approach was chosen [35,36]. As a starting point, the Cascaded Partial Decoder (CPD) was used [1]. As of the time of writing, it achieved state of the art results in SOD, and its source code as well as pre-trained weights are publicly available. Compared to other models for SOD, the CPD does not use all features of the backbone network to decode them into the salience map. Instead, in one branch the features of deep layers are used in combination with a holistic attention module to guide the attention of a second branch, also based on the features of deep layers, which computes the final salience map.

The CPD was trained on images with a resolution of 352×352. As small stains are not visible if complete images of the created dataset are resized to this resolution, and to increase the size of the dataset, crops with the resolution 512×512 were extracted and subsequently resized to a resolution of 352×352. For data augmentation purposes, overlapping crops were extracted by moving a sliding window with a stride of 256 over the image. This leads to well over 100,000 crops. However, most of these crops do not contain any pixel belonging to a stain. In order to have a more balanced dataset, all crops containing at least a single pixel belonging to a stain were selected. In addition, the same amount of crops that do not contain a stain were selected at random from all the other crops. Afterwards, the crops were split into training/validation/test data at a ratio of 60/20/20. This split was not created by randomly sampling all crops because the crops partly overlap. Moreover, as described in Section 2, the dataset contains several images for a single piece of laundry. Thus, the same stain can be visible in more than one image. To prevent memorization of specific stains, we split the crops based on the piece of laundry shown in the image. Still, we made sure that the distribution of crops with and without stains was balanced within each split. Overall, the training split contains about 19,000 cropped images while the validation and test splits contain about 6300 cropped images each.

To improve the training results, we experimented with several optimizers, learning rates and decay rates. We settled on the Adam optimizer [37] and a learning rate of 10^{-4} without decay. During training, the crops were resized to 352×352. The training was performed on a GTX 1080 Ti with the highest possible batch size of 10 based on the GPU memory. These parameters match the suggestions by the authors of the CPD [1]. For additional data augmentation, random horizontal and vertical flips were performed. Altogether, the model was trained for 50 epochs. A single training run took 16 to 24 h. Figure 4 shows a typical normalized loss curve and precision-recall (PR) curves computed during different training epochs. Note that even though the validation loss starts to rise after about 20 epochs, the PR curve can still improve. Therefore, training was not stopped if the validation loss reached a plateau. The reasons for this are that the binary cross entropy loss optimized in training only indirectly optimizes the PR curve and that post-processing was performed, which is described in the following section.

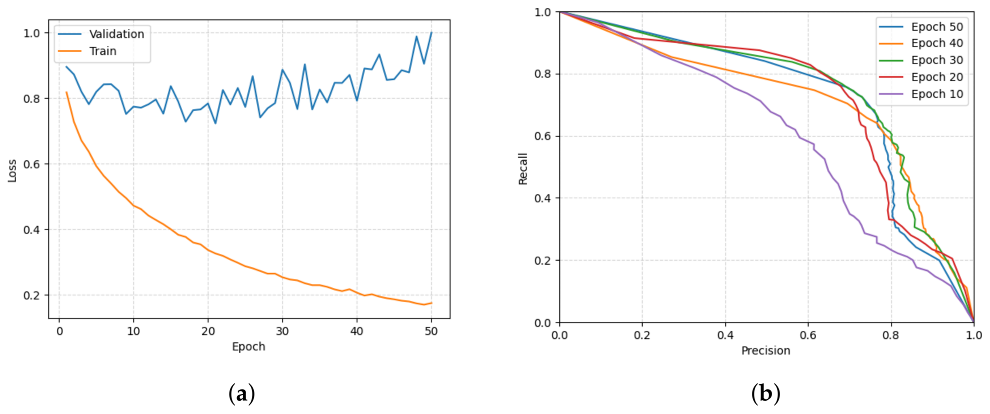


Figure 4. In (**a**), a typical normalized training loss is displayed and in (**b**) a PR curve computed on the validation split at different training epochs is shown.

4. Experiments

The existence of initial, noisy labels $\mathcal{L}_i$ and revised labels $\mathcal{L}_r$ provides a unique situation to research how model predictions can be used to reduce label noise. Particularly, it is investigated how the model predictions can be used to add stains that have been overlooked in $\mathcal{L}_i$ and how this improves overall model performance. In this section, two experiments are described: 1. In the MAL experiment, human interaction is simulated as selecting valid predicted regions, which are not part of $\mathcal{L}_i$. 2. In the self-training experiment predicted regions are automatically selected based on high certainty. In general, the experiments proceed as follows:

1. A model is trained on the training split of the dataset with the initial labels $\mathcal{L}_i^t$ and the weights m_0 from the epoch yielding the best results on the validation split with the revised labels $\mathcal{L}_r^v$ are selected.
2. The predictions P of the model m_0 on the training images are combined with the initial Labels $\mathcal{L}_i^t$ to create a model-revised labeling $\mathcal{L}_{m_0}^t = f(\mathcal{L}_i^t, P)$. Different label revision functions $f(\mathcal{L}_i^t, P)$ are used for the MAL and the self-training experiment.
3. A new model is trained on the training split of the dataset with the model-revised labels $\mathcal{L}_{m_0}^t$ and the weights m_1 from the epoch yielding the best results on the validation split with the revised labels $\mathcal{L}_r^v$ are selected.
4. The results of m_0 and m_1 on the validation split with the revised labels $\mathcal{L}_r^v$ are compared to see if training with the model-revised labels $\mathcal{L}_{m_0}^t$ increased the overall performance.

For clarity, Table 1 summarizes the different kinds of labels and their denotation, and Figure 5 illustrates the procedure.

Table 1. Summary of the different labels for clarity.

$\mathcal{L}_i$:	Our initially created labels.
$\mathcal{L}_r$:	Our revised labels.
$\mathcal{L}_{m_0}$:	Model-revised labels created with the predictions of model m_0 through the label revision function $f(\mathcal{L}_i^t, P)$.
$\mathcal{L}^t$:	The training split of the labels.
$\mathcal{L}^v$:	The validation split of the labels.

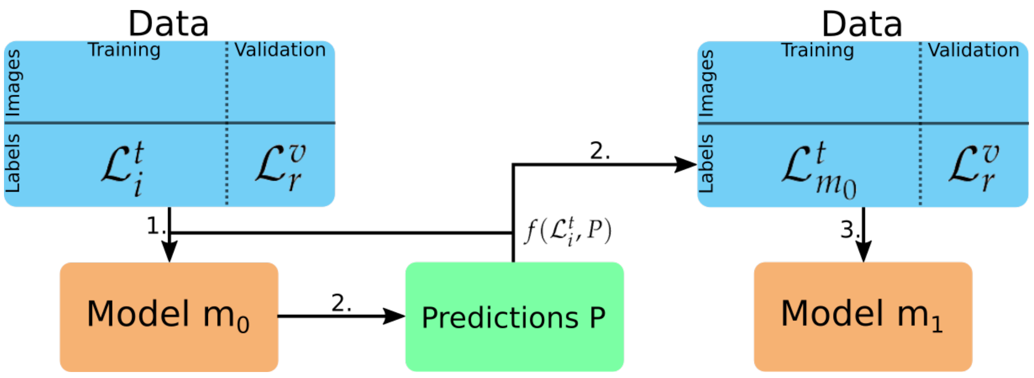


Figure 5. Illustration of the procedure of both experiments.

In the MAL experiment, the label revision function $f(\mathcal{L}_i^t, P)$ is as follows: First, a threshold δ_p is applied to the predictions P of the model to get binary predictions P_b. Subsequently, the model-revised labels $\mathcal{L}_{m_0}^t$ are created by taking $\mathcal{L}_i^t$ and adding every region from the binary prediction P_b that does not intersect a region in $\mathcal{L}_i^t$ but has an intersection over union (IoU) higher than threshold δ_{IoU} with a region from the revised labels $\mathcal{L}_r^t$. This corresponds to a human selecting predicted regions if they match a stain that was missed in the initial labels. Since selecting a valid region is faster and simpler than outlining a region, this experiment simulates model-assisted labeling.

In the self-training experiment, the label revision function $f(\mathcal{L}_i^t, P)$ is as follows: Again, a threshold δ_{upper} is applied to the predictions P of the model to obtain a binary prediction $P_{b_{upper}}$. As each pixel value in P correlates to how certain the model is that this pixel belongs to a stain, $P_{b_{upper}}$ contains regions where the model is highly certain. Then, the model-revised labels $\mathcal{L}_{m_0}^t$ are created by taking $\mathcal{L}_i^t$ and adding every region in $P_{b_{upper}}$ that does not intersect a region in $\mathcal{L}_i^t$. However, the regions from $P_{b_{upper}}$ are not directly added, but corresponding regions from a second binary prediction $P_{b_{lower}}$, created with a threshold δ_{lower} with $\delta_{lower} < \delta_{upper}$, are used. The reason for this is that, in the researched setting, it is preferable to detect more pixels of a stain than necessary rather than missing some. $P_{b_{lower}}$ contains the same regions as $P_{b_{upper}}$, but more extensive to ensure that the borders of the stains are included.

Both experiments were repeated several times since the training of a model such as the CPD involves a great deal of randomness. As a result, the results of a single run are not reliable. We only performed five repetitions since a single training took up to 24 h. Hence, a single run of an experiment including two trainings as well as evaluations took about two days. For a fair comparison, we always selected the best weights of the models m_0 and m_1 based on the evaluation of the validation split with the revised labels $\mathcal{L}_r^v$. In addition to the results of the experiments, we also show the results of a training with the revised labels $\mathcal{L}_r^t$. They provide an upper boundary since, at least in the MAL experiment, the model-revised labels cannot be better than the revised labels $\mathcal{L}_r^t$. Furthermore, the effects of the choice of the parameters δ_p, δ_{IoU}, δ_{upper} and δ_{lower} are described.

What remains to be discussed is how the model performance is evaluated. Ordinarily, in SOD, PR curves, F_β and the mean absolute error (MAE) are reported [1,38,39]. However, these measures do not correlate well with the objective to detect all stains. Since PR curves, F_β and MAE are evaluated in a pixel-wise fashion, their ratings mainly depend on the detection of a few large stains while many small stains remain unnoticed [40]. Hence, we report the application oriented measure region detection rate (RDR) [40]. It is similar to the object detection measure used in the ICDAR competition [41]. The RDR establishes criteria that define whether a predicted region is a false positive or valid detection and whether a region is considered to be correctly detected. The criteria allow one-to-one, one-to-many, many-to-one and also many-to-many matches. The final score is computed as follows:

$$RDR = \frac{C}{n + \alpha F} \tag{1}$$

where C is the number of correctly predicted regions, n is the total number of regions and F is the number of false predictions. The factor α allows to weight the importance of avoiding false predictions. For the evaluation of the RDR in the experiments, we chose the same parameters as reported in [40] because we prefer detecting all stains in contrast to attaining a few false positives. Like precision, recall and F_β, the RDR is computed for binary predictions. Therefore, we compute the RDR for a variety of thresholds and draw an RDR curve. As the final performance measure, we use the sum of the best RDR and the area under curve (AUC) to weight a good performance at all thresholds with the overall best performance.

Before computing the RDR for the models' predictions, however, another post- processing step was performed to deal with rough labels (Figures 3b and 6b). Otherwise, accurate predictions for such regions receive a low score. Thus, all predicted regions, of which 75% lie inside the same labeled region, are connected through their convex hull. An example of this procedure is displayed in Figure 6 in which many small regions in the prediction are connected to match a large region in the rough label. As another post-processing step, it would also be possible to connect regions in the label if they match to the same region in the prediction. The rationale would be that due to the rough labels the model is trained to group certain stains together. We refrained from doing this, however, since we want to promote models which produce accurate predictions rather than rough predictions.

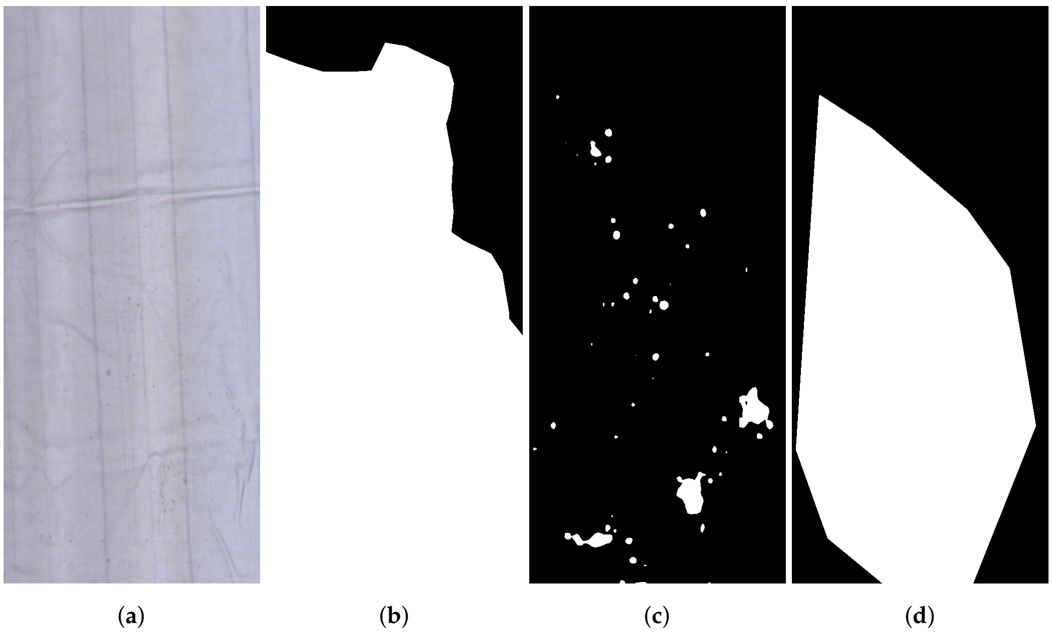


Figure 6. An illustration of the connection of predicted regions. (**a**) shows an image of our dataset and (**b**) shows a rough pixel-wise labeling. (**c**) is a prediction before the connection of regions and (**d**) is the prediction afterwards.

In Table 2, approaches from related work that apply self-training to deal with label noise are summarized in order to compare these approaches with ours. Two of these approaches [7,18] deal with image classification instead of salient object detection. Nevertheless, directly using the model's predictions as labels [7] could be carried out. However, we think that this method is not applicable because the accuracy of our model is much lower, meaning that predictions are far less accurate. Thus, we try to reduce noise by selecting regions of high certainty from the predictions based on the knowledge that stains are missing in the labels. In [19] self-training is applied to salient object detection. They utilize the fact that images contain different classes of objects by clustering images and subsequently use common color features of the clusters to improve predictions. In our case, there is only a single class of objects (stains) which makes clustering less effective. Furthermore, [19] do not deal with multiple salient objects per image.

Table 2. Comparison of different approaches to self-training for label noise reduction.

Bagherinezhad et al. [7]	The predictions of the model are directly used as labels.
Han et al. [18]	Linearly interpolate between the original noisy label and a corrected label. The corrected label is computed from the model's predictions and a kind of prototype matching.
Luo et al. [19]	The predictions of the model are refined through saliency-guided co-segmentation. Images are clustered based on salience, color and positional features and then, an interactive segmentation algorithm similar to GrabCut is applied, in which foreground and background models are complemented by models for the whole cluster.
Ours—Self-Training	Regions from the predictions made by the model are selected if the model predicts them with a high certainty.

5. Results and Discussion

In this section, the results of both experiments are presented and discussed. The presentation begins with the choice of parameters in the MAL experiment and continues with the choice of parameters in the self-training experiment. Subsequently, the overall results of the experiments are discussed.

The effect of the choice of different parameters on the label revision for both experiments are illustrated in Figure 7. The top row shows the effect of different thresholds in the MAL experiment, and the bottom row shows the effect of different thresholds in the self-training experiment. In all of the illustrations, the true positive rate (TPR) of pixels added in the revision is drawn as a blue curve and the count of regions added to the labels is drawn as an orange curve. In Figure 7a δ_{IoU} is altered while $\delta_p = 100$ is kept constant. On the contrary, in Figure 7b δ_p is altered while $\delta_{IoU} = 0.5$ is kept constant. In Figure 7c δ_{lower} is altered with $\delta_{upper} = 175$, and in Figure 7d $\delta_{lower} = 100$ is constant while δ_{upper} is altered.

The illustration of different choices for δ_{IuO} in Figure 7a shows the expected outcome. Increasing δ_{IoU} increases the TPR and decreases the amount of regions added to the labels. Our choice of $\delta_{IoU} = 0.5$ guarantees that most of the pixels added to the labels are true positives since the TPR is higher than 80%. On the contrary, the effect of the choice of δ_p as illustrated in Figure 7b shows unexpected results. Mostly, the curves follow the expectations that the higher δ_p the fewer regions are added and the higher the TPR. However, at low thresholds the amount of added regions increases, and at $\delta_p \approx 220$ there is a sudden drop in the TPR. The reasons for the former are either that large correct predictions conforming to rough labels are fragmented, which results in more predicted regions being added, or that predicted regions, which were previously too large, shrink to an acceptable size. The reason for the latter is that at $\delta_p \approx 220$ some correctly predicted regions are no longer predicted while some false predictions remain. Our choice of $\delta_p = 100$ is a trade-off between a large amount of added regions and a high TPR.

The effects of the choice of δ_{lower} and δ_{upper}, illustrated in Figure 7c,d, also show some unexpected results. Interestingly, the higher δ_{lower}, the more regions are added to the labels and the TPR decreases. This means that correct and rough predictions are fragmented through high thresholds while erroneous predictions remain. Thus, the amount of true positive pixels in the prediction decreases while the amount of false positive pixels stays the same. The same effect can be observed by the fact that the TPR decreases as δ_{upper} increases. Therefore, the plots suggest choosing low values for both δ_{lower} and δ_{upper}. Higher values, however, should promote more fine-grained predictions that do not repeat the rough labeling. Hence, we chose $\delta_{lower} = 100$ and $\delta_{upper} = 175$.

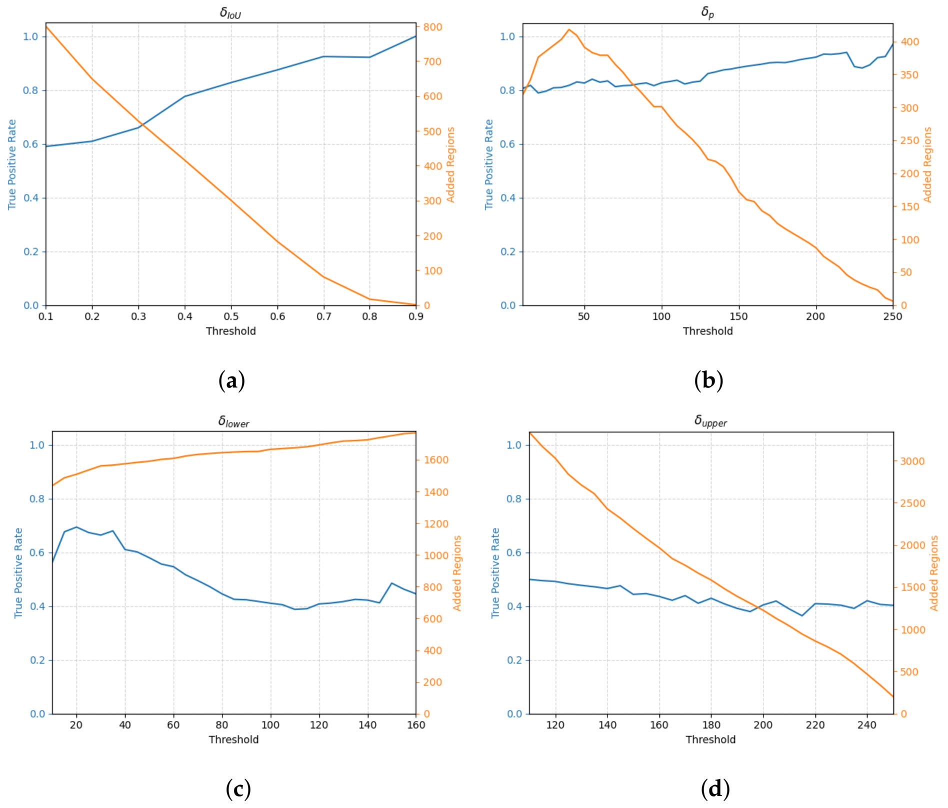
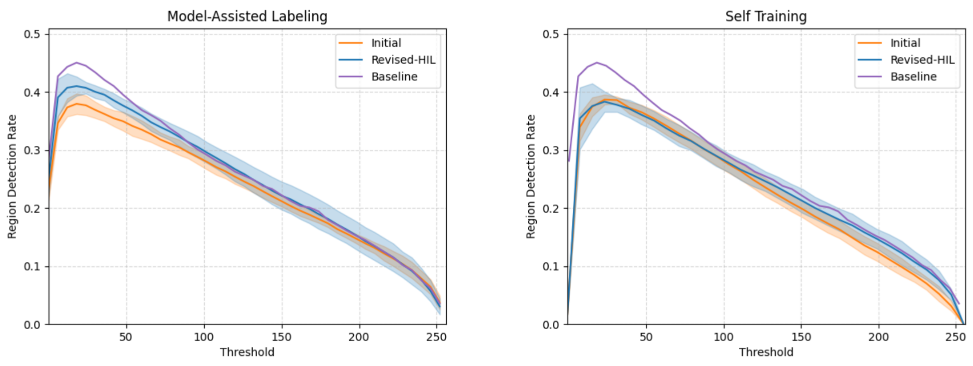


Figure 7. Illustrations of the effect of different parameters for the label revision function. In each plot, the blue curve shows the TPR of pixels added to the labels and the orange curve shows how many regions were added. The top row shows the parameters of the MAL experiment and the bottom row shows the parameters of the self-training experiment. (**a**) illustrates the effect of the choice of δ_{IoU}, (**b**) of δ_p, (**c**) of δ_{lower} and (**d**) of δ_{upper}.

Altogether, the illustrations of the effects of different parameter choices already indicate that the MAL experiment is more promising than the self-training experiment. The reason for this is that no matter the choice of parameters, the regions added in the self-training experiment show a distinctly lower TPR. Therefore, the model trained with the model-revised labels has to deal with more false positives. In this context, however, it is noteworthy to mention that the number of regions added during the label revision is much higher in the self-training experiment. This likely means that a similar amount of correct regions is added compared to the MAL experiment, but a greater number of false positives is added as well, since the model seems to be overconfident in its mistakes. This can be an effect of training with entropy minimization [26].

The main results of both experiments are visualized in Figure 8 and Table 3. Table 3 contains the scores of the models m_0 and m_1 for each run in each experiment. In Figure 8, the average RDR curves of both experiments are illustrated. On the left (Figure 8a), the curves for the MAL experiment are displayed, and on the right (Figure 8b), the curves of the self-training experiment are presented. Both figures include the results of the baseline model trained on $\mathcal{L}_r^t$. The variance of each RDR curve is indicated by the shaded area with the same color as the curve.

The results confirm the assumption that the MAL experiment achieves better results than the self-training experiment. Figure 8a shows a visible improvement of the revised model towards the baseline while in Figure 8b an improvement is only visible if the shaded area is considered. The numbers in Table 3 further underline this observation. In the MAL experiment the score improves in every trial. In the self-training experiment, the score only decreases in a single trial but the improvements are generally lower compared to the MAL experiment. Thus, it can be stated that adding regions of high certainty for self-training is applicable but the improvements are generally low so that the additional computational effort has to be considered. In contrast, if a human assists the model by only selecting good predictions, a considerable improvement can be achieved.



(**a**) (**b**)

Figure 8. Illustration of the main results of both experiments. (**a**) shows the average RDR curves for the MAL experiment and (**b**) displays the average RDR curves for the self-training experiment. The variance of the average RDR curves is visualized as a shaded area around the curve.

Table 3. The final scores (sum of the best RDR and the AUC) for each run of each experiment. For each run, the higher score is highlighted in green.

a

	Model-Assisted Labeling		
Run	m_0	m_1	**Difference**
1	0.612	0.657	0.045
2	0.621	0.633	0.012
3	0.579	0.680	0.101
4	0.621	0.661	0.040
5	0.660	0.711	0.051

b

	Self-Training		
Run	m_0	m_1	**Difference**
1	0.590	0.635	0.045
2	0.619	0.640	0.021
3	0.635	0.615	-0.020
4	0.610	0.625	0.015
5	0.619	0.635	0.026

6. Conclusions

In this paper, we described the difficulties in detecting stains on images of laundry as well as methods for reducing label noise in this setting. The creation of a custom dataset for this problem has been described. Challenges, posed by images of this dataset, were discussed: 1. Due to large numbers of small stains, the labeling is sometimes roughly grouping stains together. 2. Stains are often small compared to the overall image size. The first challenge complicates the evaluation of model performance on this dataset, since precise predictions receive a bad rating. As a result, we applied a post-processing step to the predictions of the model by connecting all regions conforming to a rough group in the labels through their convex hull.

The second challenge arose from the fact that stains were missed in our initially created labels. This was revealed by fine-tuning a model for SOD to this dataset, which correctly predicted stains overlooked in the labels. As a result, the labels were revised and label revision with the assistance of the model was researched in two different experiments. The first experiment shows that the predictions of the model can be successfully used to assist a human labeler in improving labels by suggesting stains. In the second experiment, we researched whether the label revision can be performed automatically through self-training. The results of this experiment show that a high certainty approach to select predicted regions works, but the improvements are usually small, and that in rare instances the performance can deteriorate.

Overall, our MAL method improved model performance by 8.52% on average, and our self-training method improved performance by 2.58% on average. Advantages of our methodology are that it can be easily implemented and improves performance. In the case of MAL, the increase in performance is higher compared to self-training. However, performing MAL requires greater effort than self-training, since a human has to select good predictions. A disadvantage of our approach is that the computational effort increases because the model has to be trained twice. Furthermore, our approach cannot be used to generally revise labels since it only allows adding regions which were previously overlooked. It could also be improved by removing erroneous regions and by refining roughly labeled regions. The refinement of rough labels would be especially useful, since they complicate the evaluation.

Thus, in future work, we would like to address the following questions:

- Is it possible to refine rough regions and to remove erroneous regions in addition to adding overlooked regions?
- Is it possible to apply MAL during the initial labeling by training a model with limited data and then using it to make suggestions, as was carried out by Hasty [42]?
- Can self-training be successfully applied by either using a different approach to selecting predicted regions or by filtering false positives from highly certain predictions?

Author Contributions: Conceptualization, T.H. and F.K.; data curation, T.H.; formal analysis, T.H.; funding acquisition, F.K.; investigation, T.H.; methodology, T.H. and F.K.; resources, F.K.; software, T.H.; supervision, F.K.; validation, T.H. and F.K.; visualization, T.H.; writing—original draft, T.H.; writing—review and editing, T.H. and F.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the German Federal Ministry of Economics and Technology (BMWi) through the Flachwaschen project with grants 03ET1575C.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work was supported by the German Federal Ministry of Economics and Technology (BMWi):

Supported by:

Federal Ministry
for Economic Affairs
and Energy

on the basis of a decision
by the German Bundestag

Conflicts of Interest: The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AUC	area under curve
CPD	Cascaded Partial Decoder
IoU	intersection over union
MAE	mean absolute error
MAL	model-assisted labeling
PR	precision-recall
RDR	region detection rate
SOD	salient object detection
TPR	true positive rate
CNN	Convolutional Neural Network

References

1. Wu, Z.; Su, L.; Huang, Q. Cascaded Partial Decoder for Fast and Accurate Salient Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3907–3916.
2. Qin, X.; Zhang, Z.; Huang, C.; Gao, C.; Dehghan, M.; Jagersand, M. BASNet: Boundary-Aware Salient Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7479–7489.
3. Qin, X.; Zhang, Z.; Huang, C.; Dehghan, M.; Zaiane, O.R.; Jagersand, M. U2-Net: Going deeper with nested U-structure for salient object detection. *Pattern Recog.* **2020**, *106*, 107404. https://doi.org/10.1016/j.patcog.2020.107404.
4. Zoph, B.; Ghiasi, G.; Lin, T.Y.; Cui, Y.; Liu, H.; Cubuk, E.D.; Le, Q. Rethinking Pre-training and Self-training. *arXiv* **2020**, arXiv:2006.06882.
5. Tao, A.; Sapra, K.; Catanzaro, B. Hierarchical Multi-Scale Attention for Semantic Segmentation. *arXiv* **2020**, arXiv:2005.10821.
6. Huang, Y.; Jia, W.; He, X.; Liu, L.; Li, Y.; Tao, D. Channelized Axial Attention for Semantic Segmentation. *arXiv* **2021**, arXiv:2101.07434.
7. Bagherinezhad, H.; Horton, M.; Rastegari, M.; Farhadi, A. Label Refinery: Improving ImageNet Classification through Label Progression. *arXiv* **2018**, arXiv:1805.02641.
8. Khoreva, A.; Benenson, R.; Hosang, J.; Hein, M.; Schiele, B. Simple Does It: Weakly Supervised Instance and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 876–885.
9. Pont-Tuset, J.; Arbeláez, P.; Barron, J.T.; Marques, F.; Malik, J. Multiscale Combinatorial Grouping for Image Segmentation and Object Proposal Generation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 128–140. https://doi.org/10.1109/TPAMI.2016.2537320.
10. Rother, C.; Kolmogorov, V.; Blake, A. *"GrabCut": Interactive Foreground Extraction Using Iterated Graph Cuts*; ACM SIGGRAPH 2004 Papers; ACM: New York, NY, USA, 2004; pp. 309–314. https://doi.org/10.1145/1186562.1015720.
11. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848. https://doi.org/10.1109/TPAMI.2017.2699184.
12. Hsu, C.C.; Hsu, K.J.; Tsai, C.C.; Lin, Y.Y.; Chuang, Y.Y. Weakly Supervised Instance Segmentation using the Bounding Box Tightness Prior. In *Advances in Neural Information Processing Systems 32*; Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F.d., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2019; pp. 6586–6597.
13. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–27 October 2017; pp. 2961–2969.

14. Lu, Z.; Fu, Z.; Xiang, T.; Han, P.; Wang, L.; Gao, X. Learning from Weak and Noisy Labels for Semantic Segmentation. *IEEE Trans. Pattern Anal. Mac. Intell.* **2017**, *39*, 486–500. https://doi.org/10.1109/TPAMI.2016.2552172.
15. Wang, L.; Lu, H.; Wang, Y.; Feng, M.; Wang, D.; Yin, B.; Ruan, X. Learning to Detect Salient Objects With Image-Level Supervision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 136–145.
16. Bekker, A.J.; Goldberger, J. Training deep neural-networks based on unreliable labels. In Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China, 20–25 March 2016; pp. 2682–2686. https://doi.org/10.1109/ICASSP.2016.7472164.
17. Zhang, J.; Zhang, T.; Daf, Y.; Harandi, M.; Hartley, R. Deep Unsupervised Saliency Detection: A Multiple Noisy Labeling Perspective. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 9029–9038. https://doi.org/10.1109/CVPR.2018.00941.
18. Han, J.; Luo, P.; Wang, X. Deep Self-Learning From Noisy Labels. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 5138–5147.
19. Luo, A.; Li, X.; Yang, F.; Jiao, Z.; Cheng, H. Webly-supervised learning for salient object detection. *Pattern Recog.* **2020**, *103*, 107308. https://doi.org/10.1016/j.patcog.2020.107308.
20. Barnes, C.; Shechtman, E.; Finkelstein, A.; Goldman, D.B. *PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing*; ACM SIGGRAPH 2009 Papers; Association for Computing Machinery: New York, NY, USA, 2009; pp. 1–11. https://doi.org/10.1145/1576246.1531330.
21. Yi, R.; Huang, Y.; Guan, Q.; Pu, M.; Zhang, R. Learning from Pixel-Level Label Noise: A New Perspective for Semi-Supervised Semantic Segmentation. *arXiv* **2021**, arXiv:2103.14242.
22. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Learning Deep Features for Discriminative Localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2921–2929.
23. Rosenberg, C.; Hebert, M.; Schneiderman, H. Semi-Supervised Self-Training of Object Detection Models. In Proceedings of the 2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05), Washington, DC, USA, 5–7 January 2005; Volume 1, pp. 29–36. https://doi.org/10.1109/ACVMOT.2005.107.
24. Bengio, Y.; Louradour, J.; Collobert, R.; Weston, J. Curriculum learning. In Proceedings of the 26th Annual International Conference on Machine Learning. Association for Computing Machinery, Montreal, QC, Canad, 14–18 June 2009; pp. 41–48. https://doi.org/10.1145/1553374.1553380.
25. Zou, Y.; Yu, Z.; Kumar, B.V.K.V.; Wang, J. Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 289–305.
26. Zou, Y.; Yu, Z.; Liu, X.; Kumar, B.V.K.V.; Wang, J. Confidence Regularized Self-Training. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019; pp. 5982–5991.
27. Huang, Y.; Qiu, C.; Guo, Y.; Wang, X.; Yuan, K. Surface Defect Saliency of Magnetic Tile. In Proceedings of the 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE), Munich, Germany, 20–24 August 2018; pp. 612–617. https://doi.org/10.1109/COASE.2018.8560423.
28. Bai, X.; Fang, Y.; Lin, W.; Wang, L.; Ju, B.F. Saliency-Based Defect Detection in Industrial Images by Using Phase Spectrum. *IEEE Trans. Ind. Inf.* **2014**, *10*, 2135–2145. https://doi.org/10.1109/TII.2014.2359416.
29. Song, K.; Yan, Y. Micro Surface Defect Detection Method for Silicon Steel Strip Based on Saliency Convex Active Contour Model. *Math. Probl. Eng.* **2013**, *2013*, 429094. https://doi.org/10.1155/2013/429094.
30. Gharsallah, M.B.; Braiek, E.B. Weld Inspection Based on Radiography Image Segmentation with Level Set Active Contour Guided Off-Center Saliency Map. *Adv. Mater. Sci. Eng.* **2015**, *2015*, 871602. https://doi.org/10.1155/2015/871602.
31. Song, K.C.; Hu, S.P.; Yan, Y.H.; Li, J. Surface Defect Detection Method Using Saliency Linear Scanning Morphology for Silicon Steel Strip under Oil Pollution Interference. *ISIJ Int.* **2014**, *54*, 2598–2607. https://doi.org/10.2355/isijinternational.54.2598.
32. Bonnin-Pascual, F.; Ortiz, A. A probabilistic approach for defect detection based on saliency mechanisms. In Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA), Barcelona, Spain, 16–19 September 2014. https://doi.org/10.1109/ETFA.2014.7005257.
33. About Model-Assisted Labeling (MAL). Available online: https://docs.labelbox.com/en/core-concepts/model-assisted-labeling (accessed on 26 May 2021).
34. Fan, D.P.; Cheng, M.M.; Liu, J.J.; Gao, S.H.; Hou, Q.; Borji, A. Salient Objects in Clutter: Bringing Salient Object Detection to the Foreground. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 186–202.
35. Bengio, Y. Deep Learning of Representations for Unsupervised and Transfer Learning. In Proceedings of ICML Workshop on Unsupervised and Transfer Learning, Bellevue, WA, USA, 27 June 2012, pp. 17–36.
36. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2014; Volume 2, pp. 3320–3328.
37. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**, arXiv:1412.6980.
38. Chen, S.; Tan, X.; Wang, B.; Hu, X. Reverse Attention for Salient Object Detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 234–250.

39. Hou, Q.; Cheng, M.; Hu, X.; Borji, A.; Tu, Z.; Torr, P.H.S. Deeply Supervised Salient Object Detection with Short Connections. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 815–828. https://doi.org/10.1109/TPAMI.2018.2815688.
40. Huxohl, T.; Kummert, F. Region Detection Rate: An Applied Measure for Surface Defect Localization. In Proceedings of the 2021 IEEE International Conference on Signal and Image Processing Applications (ICSIPA) (IEEE ICSIPA, Kuching, Malaysia, 18–19 November 2021.
41. Wolf, C.; Jolion, J.M. Object count/area graphs for the evaluation of object detection and segmentation algorithms. *Int. J. Document Anal. Recog. (IJDAR)* **2006**, *8*, 280–296. https://doi.org/10.1007/s10032-006-0014-0.
42. AI-Powered Annotation. Available online: https://hasty.ai/annotation/ (accessed on 27 July 2021).