Label Assistant: A Workflow for Assisted Data Annotation in Image Segmentation Tasks

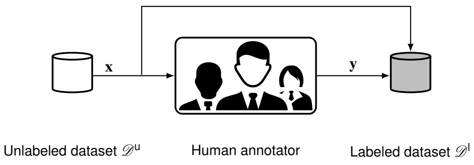
Marcel P. Schilling[1], Luca Rettenberger[1], Friedrich Münke[1], Haijun Cui[2], Anna A. Popova[2], Pavel A. Levkin[2], Ralf Mikut[1], Markus Reischl[1]

[1]Institute for Automation and Applied Informatics
[2]Institute of Biological and Chemical Systems
Karlsruhe Institute of Technology
Hermann-von-Helmholtz-Platz 1, 76344 Eggenstein-Leopoldshafen
E-Mail: marcel.schilling@kit.edu

Abstract

Recent research in the field of computer vision strongly focuses on deep learning architectures to tackle image processing problems. Deep neural networks are often considered in complex image processing scenarios since traditional computer vision approaches are expensive to develop or reach their limits due to complex relations. However, a common criticism is the need for large annotated datasets to determine robust parameters. Annotating images by human experts is time-consuming, burdensome, and expensive. Thus, support is needed to simplify annotation, increase user efficiency, and annotation quality. In this paper, we propose a generic workflow to assist the annotation process and discuss methods on an abstract level. Thereby, we review the possibilities of focusing on promising samples, image pre-processing, pre-labeling, label inspection, or post-processing of annotations. In addition, we present an implementation of the proposal by means of a developed flexible and extendable software prototype nested in hybrid touchscreen/laptop device.

DOI: 10.58895/ksp/1000138532-14 erschienen in:
Proceedings - 31. Workshop Computational Intelligence : Berlin, 25. - 26. November 2021
DOI: 10.58895/ksp/1000138532 | https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000138532/



x
y
Unlabeled dataset $\mathscr{D}^{\mathrm{u}}$
Human annotator
Labeled dataset $\mathscr{D}^{\mathrm{l}}$

Figure 1: Naïve Workflow: A human annotator iterates over an unlabeled dataset $\mathscr{D}^{\mathrm{u}}$ to sequentially label a sample $\mathbf{x}$ in order to generate labels $\mathbf{y}$ to build a labeled dataset $\mathscr{D}^{\mathrm{l}}$ without any form of assistance.

1 Introduction

Current research in the domain of image processing is focused on Deep Learning (DL) architectures. Deep Neural Networks (DNNs) like for instance Convolutional Neural Networks (CNNs) show very promising results to solve computer vision tasks like image classification or segmentation. For example, AlexNet [1] with more than 80.000 citations (date of statistic: May, 2021) w.r.t. image classification on ImageNet [2] shows the impact of DL in the field of image processing. Walsh et al. [3] argue that DNNs are beneficial to achieve accurate prediction quality in complex scenarios like biomedical applications.

However, the authors in [3, 4] name as one general bottleneck of DL that image annotation[1] is time-consuming and often requires expert knowledge as a bottleneck. Besides, following the arguments of Northcutt et al. [5], label quality can negatively affect model performance. This may lead to a selection of sub-optimal machine learning models since benchmarks with errors in labels are not reliable in general. Karimi et al. [6] argue that especially in small data scenarios like biomedical problems, an erroneous annotation may significantly reduce the performance of DNNs.

The naïve way to generate a labeled dataset $\mathscr{D}^{\mathrm{l}} = \{(\mathbf{x}_i, \mathbf{y}_i) \mid i = 1, \ldots, M\}$ composed of M instances is represented in Figure 1. An annotator adds sequentially corresponding labels $\mathbf{y}_i$ to samples $\mathbf{x}_i$ of the unlabeled dataset $\mathscr{D}^{\mathrm{u}} = \{\mathbf{x}_i \mid i =$

[1] Label and annotation are used as a synonym in this article.

$1,\ldots,N\}$ assembled of $N \geq M$ instances without any form of assistance. The labeled dataset $\mathscr{D}^{\mathrm{l}}$ incrementally increases during labeling.

There are several ideas to enhance image annotation for the development of DL applications w.r.t. decreasing annotation effort and improving annotation quality which will be presented as an overview in Section 2.

Current research predominantly focuses on separate aspects of ways to enhance a naïve generation of annotated datasets. However, to the best of our knowledge, there is no generic workflow summarizing and combing ideas of improving the image annotation procedure. We are structuring the ideas and thereby propose a comprehensive workflow. The proposal is intended to serve as a template that can be used as an initial starting point for DL projects in cases where a labeled dataset for supervised learning is required.

Our key **contributions** are the following:

- a survey of methods/approaches to assist data annotation for DL,
- a generic workflow build on meaningful combinations as well as extensions of them, and
- the introduction of a developed and extendable software prototype which can be used for assisted labeling in practical problems.

Related work is summarized in Section 2. Our workflow and methods are presented in Section 3. Besides, the software implementation is described in Section 4 following obtained results in Section 5. Finally, we conclude our work in Section 6.

2 State of the Art

The requirement of annotated data is an often addressed issue in the context of supervised DL approaches. Data efficient architectures [7, 8], self-supervised learning [9], semi-supervised learning [10], and transfer learning [11] are methods to deal with hurdle of obtaining labeled data from the perspective of network architecture/training. Considering data annotation, there are two aspects to take into account - *labeling effort* [3, 4] and *label quality* [6, 5]. In general, decreasing manual effort for users while maintaining high label quality is desired.

There are basic **software packages** like LabelMe [12], Pixel Annotation Tool [13], Image Labeling Tool [14] or the basic release of Fiji/ImageJ [15] for annotating images in the context of segmentation like depicted as naïve workflow in Figure 1.

In the context of labeling, **Deep Active Learning** (DAL) surveyed in [16] is proposed as a method to reduce labeling effort. The key concept of the mostly considered pool-based sampling is using a more elaborate sampling strategy in contrast to do a straightforward sequential approach. Based on a criterion, also named as query strategy, the human annotator should focus on the most promising samples instead of annotating without any sampling strategy naïvely. As depicted in [16], criteria can be in terms of model uncertainty or diversity of the dataset (e.g. measured via distances in latent feature space). However, DAL research mainly focuses on a theoretical perspective. Implementations in open-source labeling tools like [14, 17, 18, 19] lack, only few commercial supplier like Labelbox [20] provide interfaces to affect sampling.

A few software tools already have implemented the idea of **pre-labeling**. The general idea of pre-labeling is using a heuristic as an initial guess to simplify labeling. For instance, the Computer Vision Annotation Tool [19] or Fiji/ImageJ plugins presented in [17, 18] implement an interface for using deep learning models in order to do image pre-labeling. However, Fiji/ImageJ is implemented in Java and consequently a deployment of models nested in state-of-the-art python-based frameworks like PyTorch [21] or TensorFlow [22] requires additional effort. Commercial tools like Labelbox [20] also offer an interface to upload pre-labels. Besides, there is a function in terms of automatically creating clusters of pixels based on regional image properties in order to simplify labeling. The tool ilastik [23] enables semi-automatic image segmentation by a combination of edge detection and watershed algorithm [24]. The authors in [25] propose a pipeline for obtaining initial labels based on traditional image processing approaches like Otsu thresholding [26] and watershed algorithm [24], but an open-source software implementation lacks. Moreover, the tool LabelMe [12] offers functionality to use previous neighboring labels as pre-labels which may be beneficial for 3D/spatial or temporal data.

Furthermore, image **pre-processing** is another form of assistance in the context of image annotation. For instance, Fiji/ImageJ offers a raw image pre-processing with operations like adjustment of the contrast or noise filtering. The software BeadNet [27] is an example for image preparation in the sense that images are resampled in order to simplify labeling.

Karimi et al. [6] and Northcutt et al. [28] address the issue of noisy labels and survey options to handle them. For instance, the authors in [6] present methods like pruning wrong labels, adapting DNN structures, developing more elaborate objectives, or changing training procedures to cope with noisy labels. Northcutt et al. [28] propose Confident Learning, which is a method for pruning wrong labels in a labeled dataset after labeling has finished. Hereby, each sample is ranked concerning the disagreement between predictions of a trained model and corresponding noisy labels. However, the ideas are detached from the actual labeling process and focus on classification.

In particular, the idea of giving direct **feedback** concerning segmentation labels is a concept that is not considered in state-of-the-art approaches. Hence, software tools do neither support the possibility of scoring labels w.r.t. quality nor allow post-processing of them. Only some tools like Labelbox [20] enable manual tagging of images for a review process in order to allow further manual inspection by other annotators.

The toolbox LabelMe [12] allows using watershed algorithm [24] in order to do **post-processing** of coarse annotations. However, state-of-the-art tools lack w.r.t. post-processing functions allowing customization depending on the problem.

Moreover, the general approach is that labeling is performed using a mouse as **input device**. The work of [29] compares mouse devices with touch devices. The experiments of the authors show that in case of bimanual tasks, like fitting a mask on an object, touchscreens are beneficial.

The main open problems/questions of related work can be summarized in: (i) no definition of a comprehensive workflow combining different approaches of improving image annotation, (ii) lack of smart methods concerning sample selection directly integrated into the annotation process, (iii) no possibility for direct feedback w.r.t. label quality in the annotation process, and (iv) a missing flexible software implementation to make use of combinations of label assistance.

3 Methods

3.1 Properties and challenges in datasets

In order to introduce a workflow, we give a brief overview of properties in datasets and arising challenges as one part of our contribution:

- A dataset may have temporal or spatial relations like videos or 3D images. In this case, neighboring frames are often very similar.
- Related to this, datasets composed of video sequences are often very homogeneous within a scene, but quite heterogeneous when comparing different sequences.
- Dealing with for instance microscopy images, areas of interest may be depending on relative changes in gray value/color channels. Thus, not the whole value range in high-resolution images is relevant.
- Furthermore, noise in datasets may impede image annotation.
- The level of difficulty to solve the task can range from already available heuristics to solve the problem coarsely to hard problems. Here, there are no ways to tackle the problem directly. Besides, within a dataset, there may be a variance in examples w.r.t. difficulty to interpret them.
- Depending on the problem, there is often prior knowledge before starting labeling, e.g. a specific number of segments per sample or the desired property of no holes within a segment.
- Annotations by humans are not guaranteed to be perfect. Intra-observer and inter-observer variance may lead to errors.

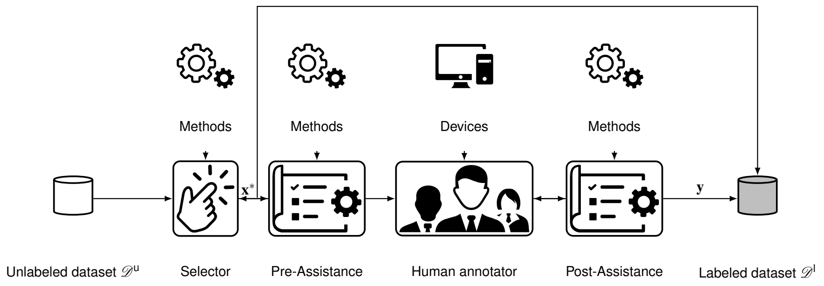


Figure 2: Assisted Labeling Workflow: A selector chooses promising samples $\mathbf{x}^*$ out of the unlabeled dataset $\mathscr{D}^{\mathrm{u}}$. The pre-assistance and post-assistance module guide the human annotator during the labeling procedure. Final labels $\mathbf{y}$ are obtained and the labeled dataset $\mathscr{D}^{\mathrm{l}}$ increases gradually.

The aforementioned properties serve as motivation for following presented approaches and methods included in the workflow proposal (Section 3.2).

3.2 Workflow

Our proposed workflow is represented in Figure 2. Firstly, starting from a unlabeled dataset $\mathscr{D}^{\mathrm{u}}$, a selector (cf. Sec. 3.3) prioritizes between all unlabeled samples and favors the next sample to label, denoted as $\mathbf{x}^*$. The subsequent pre-assistance module can yield assistance in two ways: providing pre-labels (cf. Sec. 3.4.2) as initial guesses as well as pre-processing of samples (cf. Sec. 3.4.1) to simplify annotation. Afterward, the labeling is done by the human annotator. This process can be performed using different input devices as depicted in Section 3.5. Finishing the labeling of the sample, post-assistance is a further part of the workflow. On the one hand, labels can be inspected based on defined metrics in order to provide feedback to the human annotator (cf. Sec. 3.6.1). On the other hand, based on post-processing functions, corrections of the labels are possible (cf. Sec. 3.6.2). Hence, the final label $\mathbf{y}$ is obtained and the number of labeled images in $\mathscr{D}^{\mathrm{l}}$ increases. It should be noted that Figure 2 represents the workflow in total, but in practical applications, the assistance is related to the dataset/task. Hence, in general, not all modules need to be activated.

The following sections are composed of two parts: an introduction of the *concept in general* and *presented methods*. Results of the presented methods can be found in Section 5.

3.3 Selector

General Concept The basic idea of the selector is to allow the user to affect the sampling of images during the labeling procedure and focus on promising samples $\mathbf{x}^*$ instead of labeling all images. Let an abstract query strategy, denoted as $a_j \in \mathscr{A}$, be part of the set $\mathscr{A}$ of A query strategies. Thus, a_j takes all unlabeled samples of $\mathscr{D}^\mathrm{u}$ into account and maps to a score $s_j(\mathbf{x}) \in [0,1]$ regarding each sample $\mathbf{x}$. An increasing $s_j(\mathbf{x})$ describes more relevance of a sample. To provide a generic sampling approach, the final score is obtained using weighted averaging

$$s(\mathbf{x}) = \frac{1}{\sum_{j=1}^{A} w_j^a} \sum_{j=1}^{A} w_j^a \, s_j(\mathbf{x}) \tag{1}$$

based on weights $w_j^a \geq 0$ in order to favor query strategies. The weights $w_j^a \geq 0$ are hyperparameters that need to be obtained depending on the underlying problem and query strategies a_j. The next promising sample is obtained by $\mathbf{x}^* = \underset{\mathbf{x}\in(\mathscr{D}^u \setminus \mathscr{D}^l)}{\operatorname{argmax}} \; s(\mathbf{x})$.

Presented Methods Examples for query strategies in the context of DAL can be found in [16], like for instance using model uncertainty or heterogeneity for sampling. Firstly, we present a novel cherry-picking function for users. The annotator could inspect the dataset and assign $s_i(\mathbf{x}) = 1$ for relevant samples $\mathbf{x}$ or $s_i(\mathbf{x}) = 0$ for images which should not be considered directly at the beginning of the labeling. This clears the hurdle of manually creating a list in parallel, to mark relevant samples.

Furthermore, we investigate the potential of an automated selector in the context of a sequential dataset. Thereby, we introduce two additional query strategies apart from the traditional ordered sequential sampling. On the one hand, random sampling can serve as a query strategy. On the other hand, we propose a sequence-aware sampler. If the Euclidean difference in reduced gray-level feature space between two images is larger than a pre-defined threshold, a new sequence or strong change within a sequence is detected. Afterward, the sampler selects randomly a sample per cluster and only if each cluster is represented in $\mathscr{D}^1$, a cluster is considered multiple times. It should be remarked, that for complex problems a more elaborate feature reduction method is advantageous.

3.4 Pre-Assistance

3.4.1 Image Pre-processing

General Concept The key idea of image pre-processing is not directly displaying the initial raw image during image annotation. Instead of this, a pre-processed image is generated. Abstractly speaking, the image pre-processing module is a generic function h which yields a pre-processed form of the raw sample $\mathbf{x}$ in terms of

$$\tilde{\mathbf{x}} = h(\mathbf{x}). \tag{2}$$

The objective is to accelerate annotation via displaying $\tilde{\mathbf{x}}$ where image understanding is simplified. However, it should always be considered the same pre-processing during labeling a specific dataset since varying image modalities may lead to inconsistent annotation results.

Presented Methods The desired methods are highly correlated to the depicted dataset. Therefore, we limit our presented pre-processing to two example functions h: noise filtering to deal with noisy samples and image normalization to handle high-resolution images with relative changes as depicted in Section 3.1. Custom functions can be easily implemented to find a solution that is suitable for the individual problem.

3.4.2 Pre-labeling

General Concept The main idea in the pre-labeling module is utilizing prior knowledge/heuristics, which can serve as an initial guess. Since a correction of labels is in many cases easier than starting labeling from scratch, we propose pre-labeling to boost the annotation of images. Generally speaking, an initial guess

$$\hat{\mathbf{y}} = l(\mathbf{x}) \tag{3}$$

is proposed applying a pre-label function l. However, it must be considered that pre-labeling is only meaningful if a function exists that solves the problem coarsely. In cases where l predicts mostly wrong labels, correction can slow down annotation in contrast to boost it. To evaluate quality and suitability of a pre-label function, e.g. Dice-Sørensen coefficent [30]

$$DSC\big(\mathbf{y}(\mathbf{x}), \hat{\mathbf{y}}(\mathbf{x})\big) = \frac{2 \mid \mathbf{y}(\mathbf{x}) \cap \hat{\mathbf{y}}(\mathbf{x}) \mid}{\mid \mathbf{y}(\mathbf{x}) \mid + \mid \hat{\mathbf{y}}(\mathbf{x}) \mid} \tag{4}$$

can be utilized as metric comparing initial guess $\hat{\mathbf{y}}(\mathbf{x})$ and ground truth $\mathbf{y}(\mathbf{x})$. Hence, the most suitable pre-label function l or a failure of pre-labeling in total can be determined via (4) evaluating a small set of labeled images.

Presented Methods Pre-labeling functions may be various as presented in Section 2. We present several approaches in our software prototype, which can be extended. Firstly, the traditional Otsu segmentation algorithm [26] is shown in order to assist in easier segmentation problems like enumerated in Section 3.1. Moreover, we present pre-labeling via DNNs which have already been trained on a subset of labeled samples or datasets of adjacent domains. This is beneficial in difficult image processing problems, where no suitable other heuristic exists. Besides, for sequential datasets (e.g. time-series or spatial relations) a pre-labeling is shown where previous adjacent labels are presented. Though, in this case, only a sequential image sampler is meaningful.

3.5 Human Annotator

General Concept Following the results of Forlines et al. [29], the general idea of the proposed workflow w.r.t. human annotation increases flexibility. Hence, the input device is seen as a selectable parameter of the workflow.

Presented Methods The status quo in the context of image annotation is using a mouse as an input device. We present an extension of utilizing a touchscreen for image annotation. Thereby, the touchscreen can be used with a touch pencil and fingers as well to provide a maximum level of flexibility and adaption to annotators' preferences.

3.6 Post-Assistance

3.6.1 Label inspection

General Concept As motivated in Section 2, label inspection addresses noisy labels in datasets. The general idea is to score the annotations based on G metrics $g_j \in \mathscr{G}$ which form a set $\mathscr{G}$. Each metric g_j maps labels $\mathbf{y}$ to quality scores $\gamma_j \in [0,1]$. A warning is thrown, if the final weighted score

$$\gamma(\mathbf{y}) = \frac{1}{\sum_{j=1}^{G} w_j^g} \sum_{j=1}^{G} w_j^g \, \gamma_j(\mathbf{y}) \tag{5}$$

falls below a user defined warning threshold $\gamma_0 \in [0,1]$. Analogously to equation (1), weights $w_j^g \geq 0$ allow to prioritize metrics in the final scoring. The user can reinspect the labels in case of $\gamma(\mathbf{y}) \leq \gamma_0$ and errors may be recognized immediately.

Presented Methods Metrics to inspect labels of human annotators can be various. We present in our software prototype methods which rely on expert knowledge. Thereby, we use these priors in combination with region proposals. Thus, the number of holes within a segment or number of segments serve as a quality measure. Thereby, we compare the deviation to a target property defined by an expert (e.g. only one segment per sample). Since the metric is highly correlated to the problem, custom metrics can be implemented to extend the software functionality. Moreover, using predictions of a DNN trained on a small set of labeled data for benchmarking purpose may serve as an alternative approach, which is more generic. However, this is currently not implemented in the prototype.

3.6.2 Post-processing

General Concept Practical experiments show that some specific errors are reoccurring. In these cases, post-processing can be meaningful. In general, we propose the opportunity to have an abstract post-processing function in the labeling process in order to tackle the problem of noisy labels. Hence, annotators can use this idea in cases where post-processing of labels may be helpful. Displaying a comparison of labels before and after post-processing ensures that assistance is still supervised by human annotators avoiding unwanted changes in post-processing.

Presented Methods We recognized that especially holes or small noisy segments may come up as reoccurring errors. Thus, we implemented morphological operators as a possibility to post-process segmentation maps. Analogously, the post-processing is depending on the dataset and extensions (considering properties like aspect ratio, size, or area) are possible.

4 Implementation

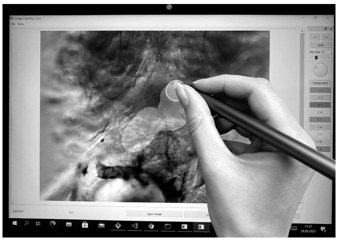
The whole generic workflow depicted in Figure 2 is transferred to practical application. Therefore, a software prototype is developed following the modular architecture of the presented workflow in Section 3. The proposed concept is implemented in a python package and therefore setup respectively integration via pip is easy to manage for users. Besides, the Graphic User Interfaces (GUIs) are developed using Qt5 [31] and thus are flexible for extensions in order to do further development. We refer to the Image Labeling Tool [14] for drawing image segmentation masks, since it allows a very flexible way of including pre-labeling without modifying the source code of the tool. Moreover, the publishers provide the tool across different platforms (Linux, Windows). All modules of our proposed workflow include examples concerning processing, scoring, and query functions according to Section 3. However, as mentioned, each module allows the implementation of custom functions in order to gain more flexibility. Consequently, users can customize the proposed workflow to the needs being faced with their individual problem respectively dataset. This may boost the application of the workflow prototype in the research community. Especially, the underlying implementation clears the hurdle to connect the proposed workflow with implementations based on state-of-the-art DL frameworks like TensorFlow and PyTorch [21, 22].

Our software prototype can be used in combination with Windows and Linux operation systems since the implementation is python-based and, using the Image Labeling Tool, relies on a cross-platform segmentation mask drawing tool. We tested it on Windows 10 and Ubuntu 20.04. The system can be used with desktop computers with mouse input devices and tablets as well. Our objective is to provide annotators (e.g. biologists) capsuled hardware, which allows labeling without any installation. Consequently, we deployed our software prototype on a Lenovo X12 Detachable which can be easily handed over to experts as capsuled system. This hardware allows a very flexible usage in terms of offering touch via fingers, touch via a pencil, and laptop mode via keyboard/mouse in parallel. Figure 3 shows the hardware in a practical use-case.

5 Results

5.1 Datasets

We demonstrate an excerpt of the concept functionalities using two biomedical binary image segmentation datasets depicted in Figure 4.


(a) Tablet mode with pencil.

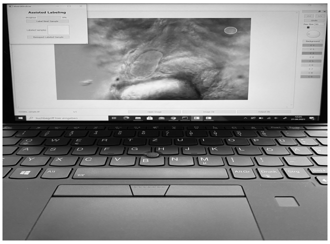
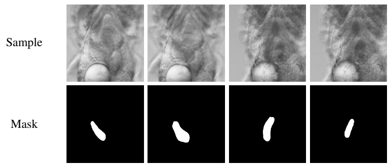
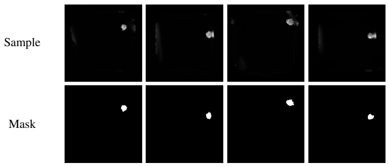

(b) Laptop mode.

Figure 3: Software prototype on Lenovo X12 Detachable.



(a) Medaka [32].



(b) DMA Spheroid [33].

Figure 4: Datasets visualizing exemplary samples and corresponding label masks.

Medaka Dataset The medaka dataset is presented in [32]. It has been released to quantify ventricular dimensions which can be relevant for the understanding of human cardiovascular diseases. An accurate image segmentation of the medaka heart is needed in order to solve this quantification task. The dataset contains 8-bit RGB images and corresponding segmentation masks describing pixels belonging to the ventricle. It includes 565 frames of training data and 165 test samples. Figure 4a illustrates examples and binary segmentation masks. The authors in [32] use the DNN U-Net [34] to solve the image segmentation task. Looking at the example frames, it becomes clear that image segmentation is difficult in this project and thus a simple thresholding algorithm would fail. Furthermore, the dataset is based on roughly 30 video sequences and as presented in Figure 4a neighboring frames may be similar.

Droplet Microarray Spheroid Dataset The spheroid dataset is recorded in a high-throughput Droplet Microarray (DMA) experiment [33]. Currently, the dataset is not publicly available, a description of the experiment is pre-

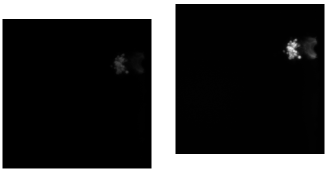
Table 1: Comparison *DSC* of different sampling scenarios (sequential/neighboring, random, sequence-aware) and dataset amounts $|\mathscr{D}^l_{\text{train}}|$ on medaka dataset [32].

	Configurations			
	Sequential/neighboring	Random	Sequence-aware	Baseline
$\mid\mathscr{D}^l_{\text{train}}\mid$	32	32	32	400
DSC in %	46.50	77.67	80.63	82.70

sented in the work of Popova et al. [33]. DMA experiments intend to do investigations for drug development and therefore accurate segmentation of fluorescence images is needed. It contains 16-bit high-resolution mono images with corresponding labels obtained by an expert. Thereby, it includes 470 frames of training data and 118 test samples. Being faced with this dataset, the main challenge is to distinguish between artifacts at image boundaries and spheroids. Thus, a straightforward thresholding approach like Otsu [26] is not accurate enough. Figure 4b illustrates this problem using example frames respectively segmentation masks.

5.2 Experiments

Selector To present the potential of the selector module, we first utilize the medaka dataset introduced in Section 5.1, which is a composition of different sequences. In order to evaluate the experiment, we compare *DSC* (4) using DNN U-Net [34] trained on different sampled training datasets (subsets of the initial training dataset) evaluated on a fixed test dataset. The baseline experiment uses almost the entire dataset (400 samples). Hereby, we compare the methods presented in Section 3.3. Results are shown in Table 1. A comparison of DNN performance in terms of *DSC* shows that by considering only a small subset, random sampling and sequence-aware sampling (selecting one random image of each sequence) are superior to standard labeling of neighboring frames in an ordered sequential fashion. However, in this example, the more elaborate sequence-aware approach did not outperform random sampling. If there are no strong imbalances w.r.t. the distribution of the dataset as well as no priors concerning the dataset, random sampling is definitely a proper starting point. Moreover, it can be recognized that the gap from an amount



(a) Raw. (b) Pre-processed.

Figure 5: Example pre-processing on DMA data: a raw sample (a) is processed to a normalized image (b) to enhance image understanding.

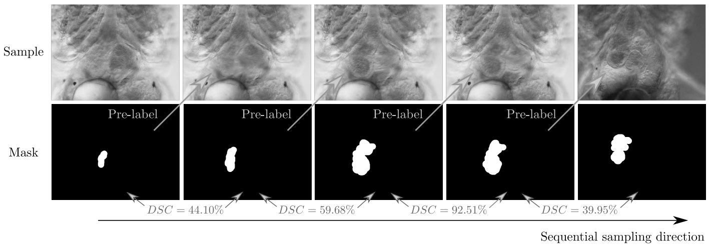


Figure 6: Comparison of sample, corresponding mask, and *DSC* between neighboring frames to illustrate temporal pre-labeling (sequential sampling form left to right) on the medaka dataset.

of $\mid \mathscr{D}^l_{\text{train}} \mid = 32$ training samples to the baseline with 400 samples is comparatively small. Hence, with an adapted sampling strategy a small amount is sufficient to obtain accurate results shown by a $DSC > 80\%$.

Pre-processing To get an impression of pre-processing, Figure 5 represents an example of the DMA spheroid dataset. Thereby, a raw high-resolution DMA mono image is compared to a pre-processed sample. The pre-processing function normalized the gray levels in the image. Thus, relative changes are visible, image understanding is enhanced, and therefore annotating segmentation masks is simplified.

Pre-labeling Firstly, the potential of the proposed previous label usage is analyzed at the medaka dataset since it is composed of video sequences like presented in Section 5.1. Figure 6 illustrates a sequence of the sequential

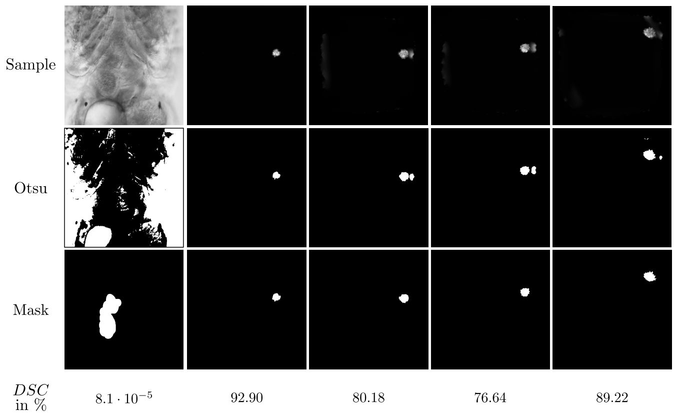


Figure 7: Illustration of visual differences between Otsu pre-labeling and ground truth mask as well as *DSC* to quantify the similarity of masks on medaka (first column) and DMA spheroid samples (remaining columns).

sampling and used pre-labels. In addition to the visual impression, *DSC* (4) is printed to compare neighboring label masks. It can be shown that the first three pre-labels are beneficial since there is a direct relation between frames. Consequently, *DSC* is larger than 40% in each of those frames. Especially, frames 3 and 4 are very similar, which can be demonstrated by a $DSC = 92.51\%$. However, the last frame illustrates a remaining problem in the method if sequences change. Hereby, the displayed pre-label is not helpful in order to do image annotation of the last sample. Figure 7 presents pre-labeling using Otsu thresholding [26]. In order to execute Otsu on RGB medaka images, an upstream transformation to a gray-level image space is done at first. However, the algorithm is not suitable as a pre-labeling strategy for medaka images, which, in addition to visual inspection, a *DSC* tending to zero demonstrates, too. Thus, in this case, pre-labeling would impede annotation instead of simplifying it. Nevertheless, Otsu performs very well on DMA samples shown by $DSC \geq 76\%$. Hence, it provides helpful initial guesses w.r.t. DMA data. Having a closer inspection and comparing it with the ground truth masks, it can be recognized, that there are still small wrong mask segments. However, deleting the wrong mask segment, in this case, is much more efficient than starting image annotation from scratch. The main reason is that curved boundaries of the spheroid are already correctly predicted for the most part.

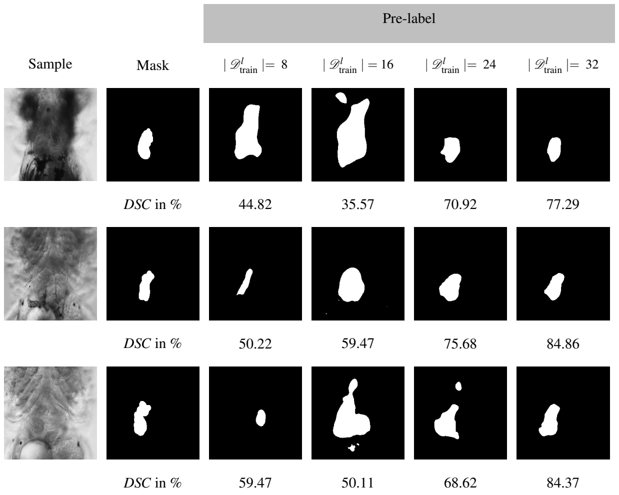
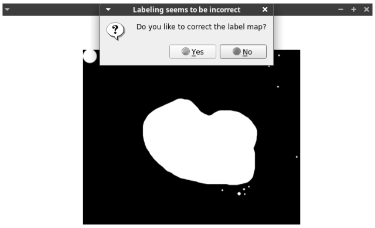

Pre-label
Sample
Mask
$|\mathscr{D}^l_{\text{train}}| = 8$
$|\mathscr{D}^l_{\text{train}}| = 16$
$|\mathscr{D}^l_{\text{train}}| = 24$
$|\mathscr{D}^l_{\text{train}}| = 32$
DSC in %
44.82
35.57
70.92
77.29
DSC in %
50.22
59.47
75.68
84.86
DSC in %
59.47
50.11
68.62
84.37

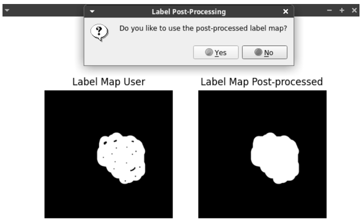
Figure 8: Illustration of DNN pre-labeling performance: comparison different amounts of training data ($|\mathscr{D}^l_{\text{train}}|$) w.r.t. visual impression and *DSC* between ground truth mask and pre-labels respectively DNN predictions.

Since there is no obvious heuristic for medaka dataset, we investigate how DNN U-Net trained on a small labeled dataset can be used as pre-labeling. Results for different amounts of training data $|\mathscr{D}^l_{\text{train}}|$ following random sampling presented in Section 3.3 can be found in Figure 8. We compare pre-labels and ground truth masks of samples $\mathbf{x} \notin \mathscr{D}^l_{\text{train}}$ not represented in the training dataset by visual impression and *DSC* (4) in parallel. Our experiments show, that by using only $|\mathscr{D}^l_{\text{train}}| = 32$ labels, a DNN can serve as a meaningful and generic pre-label strategy on medaka dataset. Furthermore, we offer in our tool the opportunity to export a training job that can directly be sent to data scientists to avoid the requirements of a graphics processing unit on the labeling device. Hence, the annotator only needs to select DNN weights provided by a data scientist. The inference time on the introduced hardware (Intel i3-1110G4) of $t_{\text{inference}} = 0.75$ s is a feasible processing amount during labeling.

Human Annotator User Experience We have presented our implemented software prototype nested in a touchscreen device to several users and have requested feedback concerning labeling comfort. The overall feedback of users


Labeling seems to be incorrect
Do you like to correct the label map?
Yes
No

(a) Label inspection.



(b) Label post-processing.

Figure 9: Examples of post-assistance: (a) an inspector warns the user since there is more than one segment labeled, (b) a post-processing can be performed to fill holes.

has been positive. Most of the users named a comfort enhancement during image annotation using a touchscreen. However, very experienced users w.r.t. mouse labeling remark that for them touchscreen labeling is not superior to using a mouse as an input device since they are used to it. Thus, especially for an average user labeling via touchscreen may facilitate access to the procedure. Concluding results, several possibilities of user input maintain the maximum level of adaption to the needs of users.

Post-Assistance Figure 9a illustrates a label inspection evaluating deviations of connected segments to the desired segment number as a quality metric γ_i introduced in Equation (5). Large deviations lead to the presented warning prompt and give users the possibility to relabel images. Consequently, using the feedback mechanism can help to increase attention w.r.t. noisy labels directly during annotation. Post-processing links reoccurring errors with an opportunity to straightforwardly solve them. Figure 9b presents post-processing in form of closing intending to avoid holes in segment masks. Similar to label inspection, the annotator can adopt the post-processing suggestion or reject it avoiding unwanted changes. Therefore, post-processing enables a way of handling common error sources using algorithms like morphological operations or custom functions depending on the underlying problem.

The key results can be summarized the following: A selector can help to reduce the amount of labeled data needed to achieve accurate DNN results. Pre-processing and pre-labeling can facilitate annotation and decrease the effort needed for labeling an image. Human annotators gain more flexibility by providing different types of input devices. Label inspection and post-processing build awareness of label quality and ways to deal with it.

6 Conclusion

Dealing with Deep Learning (DL), labeling plays an important role. We motivated that assisting annotators during labeling is desired (reducing labeling effort and increasing label quality). Methods to tackle these issues are various, but a summary and combination of those in a general concept is lack. We contribute a summary of properties and challenges in datasets w.r.t. annotation. Besides, we propose a generic workflow combing and extending various ideas of labeling enhancement. Especially, an evolved concept of label inspection and post-processing implemented directly within the annotation process is presented as a novel way to increase label quality. Our contribution is intended to serve as a template, which can be used by the community for practical DL projects where a labeled dataset is required. To make this concept applicable, we present a software prototype implementation as an initial starting point that can be customized. Several functionalities are demonstrated using the prototype processing two biomedical image segmentation datasets. The prototype enables further research on enhancing image annotation and investigations of new underlying methods like more generic feedback approaches or active learning in the proposed pipeline modules. For instance, the initial required amount of labeled data or further quantification of enhancement using an assisted labeling approach may be part of further research.

Acknowledgement

This work was funded in the KIT Future Fields project "Screening Platform for Personalized Oncology (SPPO)" and was performed on the computational resource bwUniCluster 2.0 funded by the Ministry of Science, Research and the Arts Baden-Württemberg and the Universities of the State of Baden-Württemberg, Germany, within the framework program bwHPC.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[2] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[3] Joseph Walsh et al. "Deep learning vs. Traditional computer vision". In: *Advances in Computer Vision*, pp. 128–144, 2019.

[4] Weicheng Chi et al. "Deep learning-based medical image segmentation with limited labels". In: *Physics in Medicine & Biology*, 65(23):235001, 2020.

[5] Curtis G. Northcutt, Anish Athalye, and Jonas Mueller. "Pervasive label errors in test sets destabilize machine learning benchmarks". arXiv:2103.14749 [stat.ML], 2021.

[6] Davood Karimi et al. "Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis". In: *Medical Image Analysis*, 65(5):101759, 2020.

[7] Tim Scherr et al. "Cell segmentation and tracking using CNN-based distance predictions and a graph-based matching strategy". In: *PLOS ONE*, 15(12):1–22, 2020.

[8] Fabian Isensee et al. " nnU-Net: A self-configuring method for deep learning-based biomedical image segmentation". In: *Nature Methods*, 18(2):203–211, 2021.

[9] Ting Chen et al. "A simple framework for contrastive learning of visual representations". In: *Proceedings of the 37th International Conference on Machine Learning*, pp. 1597–1607, 2020.

[10] Xiaokang Chen et al. “Semi-supervised semantic segmentation with cross pseudo supervision”. In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2613–2622, 2021.

[11] Chuanqi Tan et al. “A survey on deep transfer learning”. arXiv:1808.01974 [cs.LG], 2018.

[12] Kentaro Wada. “Labelme: Image polygonal annotation with python”. 2016, Accessed: 2021-05-28. Available: `https://github.com/wkentaro/labelme`.

[13] Amaury Bréhéret. “Pixel Annotation Tool”. 2017, Accessed: 2021-05-27, Available: `https://github.com/abreheret/PixelAnnotationTool`.

[14] Andreas Bartschat. “ImageLabelingTool”. 2019, Accessed: 2021-05-31, Available: `https://bitbucket.org/abartschat/imagelabelingtool`.

[15] Johannes Schindelin et al. “Fiji: an open-source platform for biological-image analysis”. In: *Nature Methods*, 9(7):676–682, 2012.

[16] Pengzhen Ren et al. “A survey of deep active learning”. arXiv:2009.00236 [cs.LG], 2020.

[17] Thorsten Falk et al. “U-net – deep learning for cell counting, detection, and morphometry”. In: *Nature Methods*, 16(1):67–70, 2019.

[18] Réka Hollandi et al. “AnnotatorJ: An ImageJ plugin to ease hand annotation of cellular compartments”. In: *Molecular Biology of the Cell*, 31(20):2179–2186, 2020.

[19] Boris Sekachev et al. “Computer Vision Annotation Tool (CVAT)”. 2020. Accessed: 2021-05-31, Available: `https://github.com/openvinotoolkit/cvat`.

[20] Manu Sharma, Daniel Rasmuson, and Brian Rieger. “Labelbox”. 2021. Accessed: 2021-05-28, Available: `https://labelbox.com`.

[21] Adam Paszke et al. “PyTorch: An imperative style, high-performance deep learning library”. In: *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

[22] Martín Abadi et al. “TensorFlow: Large-scale machine learning on heterogeneous distributed systems”. arXiv:1603.04467 [cs.DC], 2016.

[23] Stuart Berg et al. “Ilastik: Interactive machine learning for (bio)image analysis”. In: *Nature Methods*, 16(12):1226–1232, 2019.

[24] Serge Beucher and Christian Lantuéjoul. “Use of watersheds in contour detection”. In: *International Workshop on Image Processing: Real-Time Edge and Motion Detection/Estimation*, pp. 17–21, 1979.

[25] Fabian Englbrecht, Iris E. Ruider, and Andreas R. Bausch. “Automatic image annotation for fluorescent cell nuclei segmentation”. In: *PLOS ONE*, 16(4):1–13, 2021.

[26] Nobuyuki Otsu. “A threshold selection method from gray-level histograms”. In: *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979.

[27] Tim Scherr et al. “BeadNet: Deep learning-based bead detection and counting in low-resolution microscopy images”. In: *Bioinformatics*, 36(17):4668–4670, 2020.

[28] Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. “Confident learning: Estimating uncertainty in dataset labels”. In: *Journal of Artificial Intelligence Research*, pp. 1373–1411, 2021.

[29] Clifton Forlines et al. “Direct-touch vs. mouse input for tabletop displays”. In: *Conference on Human Factors in Computing Systems*, pp. 647–656, 2007.

[30] Shruti Jadon. A survey of loss functions for semantic segmentation. In: *IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pp. 1–7, 2020.

[31] Tor Arne Vestbø and Alessandro Portale. “Qt”. 2021, Accessed: 2021-05-31, Available: `https://github.com/qt`.

[32] Mark Schutera et al. “Machine learning methods for automated quantification of ventricular dimensions”. In: *Zebrafish*, 16(6):542–545, 2019.

[33] Anna A. Popova et al. “Facile one step formation and screening of tumor spheroids using droplet-microarray platform”. In: *Small*, 15(25):1901299, 2019.

[34] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241, 2015.