A Novel Benchmark for Refinement of Noisy Localization Labels in Autolabeled Datasets for Object Detection

Andreas Bär[1] * Jonas Uhrig[2 3] * Jeethesh Pai Umesh[1 †] * Marius Cordts[2] Tim Fingscheidt[1]

[1]Technische Universität Braunschweig, Germany [2]Mercedes Benz AG, Germany [3]University of Freiburg, Germany
*Equal contribution †Work done during an internship at Mercedes Benz AG, Germany

{andreas.baer, j.umesh, t.fingscheidt}@tu-bs.de {jonas.uhrig, marius.cordts}@mercedes-benz.com

Abstract

Autolabeling approaches are attractive w.r.t. time and cost as they allow fast annotation without human intervention. However, can we really trust the label quality of autolabeling? And further, which potential consequences arise from resulting label noise? In this work, we address these questions for localization, a subtask of object detection, by investigating the effects on a state-of-the-art deep neural network (DNN) for object detection and the widely used Pascal VOC 2012 dataset. Our contributions are threefold: First, we propose a method to inject noise into localization labels, enabling us to simulate localization label errors of autolabeling methods. Afterwards, we train a state-of-the-art object detection DNN with these noisy labels. Second, we propose a refinement network which takes a noisy localization label and its respective image as input and performs a localization refinement. Third, we again train a state-of-the-art object detection DNN, however, this time with refined localization labels. Our insights are: Training a state-of-the-art DNN for object detection on noisy localization labels leads to a severe performance drop. Our proposed localization label refinement network is able to refine the noisy localization labels. We are able to retain the performance to some extent by retraining the state-of-the-art DNN for object detection on the refined localization labels. Our study motivates a new challenging task "refinement of noisy localization labels" and sets a first benchmark for Pascal VOC 2012. Code is available at https://github.com/ifnspaml/LocalizationLabelNoise.

1. Introduction

Deep neural networks (DNNs) are the state of the art in object detection [4, 13, 26, 40] and outperform traditional machine learning algorithms [7, 10, 29]. DNNs for object detection are usually trained in a supervised fashion which requires human-labeled datasets. Human labeling, however, typically comes with high cost and is time-consuming.

Figure 1. **Examples for our localization label refinement.** Our proposed framework takes a noisy localization label $\widetilde{\boldsymbol{y}}$ (red box) as input and outputs a refined localization label $\hat{\boldsymbol{y}}$ (green box).

Autolabeling approaches [53] address this issue. They offer an alternative to traditional human-based labeling and provide much cheaper *machine-annotated* labels. However, this comes with the price of a lower label quality, also called label noise. Looking at widely established computer vision benchmarks, some studies [17, 33, 43] show that public datasets for image classification (ImageNet [44], CIFAR-10 [19]), semantic segmentation (Cityscapes [6]), as well as object detection (Pascal VOC 2012 [9]) also contain label noise. Label noise poses a problem in DNN training. Many works [1, 2, 5] have shown that label quality is of utmost importance as it directly affects the DNN's generalization capability. In particular, a line of works in image classification [15, 39, 48] highlights that the presence of noisy labels degrades the network's final performance. However, studies in more complex tasks, such as localization (a subtask of object detection, cf. Fig. 1), are rather rare or not existing.

In this paper, we investigate localization label noise. We focus on localization, since a larger line of works on noisy labels in image classification (a subtask of object detection) already exist [15, 39, 48]. In particular, we address three research questions in our paper.

First, *how can we effectively investigate localization la-*

bel noise? One option is the use of autolabeling methods. However, a new question arises regarding the choice and mix of autolabeling methods. In addition, autolabeling methods lack control over the label quality. Instead, we propose to inject noise into ground-truth localization labels of Pascal VOC 2012 [9]. Different to autolabeling approaches, our proposed experimental approach is controllable regarding the amount of label noise. We can adjust it to our needs and in consequence establish a new benchmark for training and evaluating with localization label noise. In a sense, it can be considered as a fully controllable autolabeling error simulator.

Second, *which effects does localization label noise have on the final object detection performance?* We train the state-of-the-art `Cascade R-CNN` [3] on Pascal VOC 2012 with our pre-generated noisy localization labels. We observe a drop in performance with increased localization label noise which complements the investigations being made on image classification [1, 2, 5].

Lastly, *how can we reduce the effects of localization label noise?* Assuming that our proposed localization label noise injection simulates autolabeling methods which provide noisy localization labels close to the ground truth, we aim at refining the noisy localization labels. To this end, we propose a new class of methods which is inspired by [22] and which we dub *localization label refinement network* (`LLRN`). Our `LLRN` takes an image and its respective noisy localization labels as input and outputs a refined noisy localization label. Our investigations show that training the `Cascade R-CNN` on refined localization labels indeed improves its final performance.

The rest of the paper is structured as follows: Sec. 2 discusses the related works. Sec. 3 introduces our method. Sec. 4 gives an overview over the experimental setup. Sec. 5 shows and discusses our experimental results. Finally, we conclude our paper in Sec. 6.

2. Related Works

Object detection and localization: Object detection extends image classification by localization of so-called regions of interest (RoIs). Early works in object detection were based on hand-crafted features, such as histogram of oriented gradients (HOG) [7], Viola-Jones [49], deformable parts model [10], etc. The advance of deep neural networks (DNNs) in image classification tasks [44], convolutional neural networks in particular [20, 46], paved the way for DNN research in object detection [3,11,13,14,21,36,40,42]. A milestone is the introduction of region with CNN features (`R-CNN`) in [14] which had a multi-stage training approach and set a new state of the art. Soon, improvements in the form of `Fast R-CNN` [13] and `Faster R-CNN` [40] followed by introducing a new two-stage architecture and faster computations. In addition, `Cascade R-CNN` [3] trades high complexity for high performance by using a multi-head cascade.

In this work, we use `Cascade R-CNN` [3] with a `ConvNeXt-Tiny` backbone [27] to investigate the effect of localization label noise. In particular, we compare training the `Cascade R-CNN` [3] on noisy localization labels with training it on refined localization labels using our proposed localization label refinement network (`LLRN`).

Label noise and autolabeling: Label noise in image classification tasks, such as CIFAR10 [19], MNIST [8], and Imagenet [44], are well studied [18, 33, 34, 45]. Almost all works on image classification show that label noise leads to a drop in final performance when trained using standard loss functions and architectures [23, 32, 54]. To address this problem, [35,50] propose methods for label noise reduction. However, the proposed methods build upon a relation between clean and noisy labels, which in practice is often difficult to formulate. Further, while machine-annotated labels, also dubbed autolabels, can serve as an alternative to human labeling, they are known to be error prone [37, 51, 52]. In addition, the problem of label noise in more complex tasks, such as localization (a subtask of object detection) is understudied, although still being highly relevant. In particular, He et al. [17] point out that well-known object detection datasets, such as MS COCO [25], Pascal VOC 2012 [9], and Youtube Bounding Boxes [38], contain noisy labels.

In this work, we investigate the effect of localization label noise on the task of object detection. In particular, we propose a method for localization label noise injection which gives us full control over the noise strength. In fact, we simply use this method to simulate autolabeling. Our proposed framework can be used as a benchmark for investigations in localization label errors.

Label refinement: Although previous works [33, 34] show the existence of noisy labels in image classification datasets, studies were not conducted to rectify these label errors. Many works address training under the presence of label noise instead of refining the labels. The objective of these works was mainly to train a robust network that is not influenced by label errors. For example, the authors of [12, 39] introduce a robust loss function that prevents the network from learning label noise, while [15, 48] employ teacher-student networks and use the weighted average of model weights as their final model. Further, [23, 32] consider the combination of teacher-student networks with robust loss functions for learning under noisy labels.

In this work, we investigate the effect of localization label noise on the final performance of an object detection network. Different to previous works, we do not incorporate robust loss functions or teacher-student networks to cope with the label noise. Instead, we propose a localization label refinement network (`LLRN`) inspired by [22], which refines the noisy localization labels as a preprocess-

ing step. We would like to mention [30, 31], where a localization label refinement is introduced during training instead. In [22], the authors propose to apply their method on localizations predicted by a weakly supervised object detector (WSOD) [47]. The chosen WSOD, however, produces localizations with only low intersection over union to the ground truth. Different to [22], we perform our experiments using a fully controllable localization label noise injection (LLNI) to simulate label errors of autolabeling methods. In particular, our LLNI yields noisy localization labels with an intersection over union to the ground truth ranging from low to high values. This makes our setup practically more relevant. Further, we also investigate whether equipping and training our `LLRN` in a multi-pass or cascaded-pass fashion [3] leads to better results in localization label refinement.

3. Method Description

In the following, we will introduce our proposed method in three separate parts. First, we start with a theoretical background to build mathematical foundations. Next, we give an overview of our proposed localization label noise injection which we use to simulate autolabeling errors. Finally, we introduce our proposed localization label refinement network.

3.1. Theoretical Background

We define $\boldsymbol{x} \in \mathbb{I}^{H \times W \times C}$ as a normalized image, i.e., $\mathbb{I} = [0, 1]$. Further, let H, W, and $C = 3$ refer to its height, width, and number of color channels. In addition, $\boldsymbol{x}_{h,w} \in \mathbb{I}^C$ is a pixel, with $\boldsymbol{x} = (\boldsymbol{x}_{h,w})$, pixel position $(h, w) \in \mathcal{I}$, and pixel positions set $\mathcal{I} = \{(1, 1), ..., (H, W)\}$.

Let's assume $\boldsymbol{x}$ contains objects, each having a ground-truth identifier $\overline{o} \in \overline{\mathcal{O}} = \{1, 2, ..., N_{\overline{\mathcal{O}}}\}$. We define $\overline{\boldsymbol{y}}_{\overline{o}} \in \mathcal{I}^2 \subset \mathbb{N}^4$ to be a localization label of object with identifier $\overline{o}$ and the $(N_{\overline{\mathcal{O}}} \times 4)$-dimensional matrix of the localization labels of all objects being $\overline{\boldsymbol{y}} = (\overline{\boldsymbol{y}}_{\overline{o}})$. Note that we introduced the diacritic $\overline{(\,)}$ to refer to ground-truth output entities. Further, localization label

$$\overline{\boldsymbol{y}}_{\overline{o}} = \left(\left(\overline{h}_{\overline{o}}^{(\mathrm{L})}, \overline{w}_{\overline{o}}^{(\mathrm{L})}\right), \left(\overline{h}_{\overline{o}}^{(\mathrm{R})}, \overline{w}_{\overline{o}}^{(\mathrm{R})}\right)\right) \tag{1}$$

is represented as a bounding box with top-left corner $\left(\overline{h}_{\overline{o}}^{(\mathrm{L})}, \overline{w}_{\overline{o}}^{(\mathrm{L})}\right) \in \mathcal{I}$ and bottom-right corner $\left(\overline{h}_{\overline{o}}^{(\mathrm{R})}, \overline{w}_{\overline{o}}^{(\mathrm{R})}\right) \in \mathcal{I}$ ground-truth h- and w-coordinates. We obtain the size of the bounding box by

$$\begin{aligned} \overline{H}_{\overline{o}} &= \left|\overline{h}_{\overline{o}}^{(\mathrm{L})} - \overline{h}_{\overline{o}}^{(\mathrm{R})}\right|, \\ \overline{W}_{\overline{o}} &= \left|\overline{w}_{\overline{o}}^{(\mathrm{L})} - \overline{w}_{\overline{o}}^{(\mathrm{R})}\right|, \end{aligned} \tag{2}$$

with $\overline{H}_{\overline{o}}$ and $\overline{W}_{\overline{o}}$ referring to object height and width, respectively. We further define $\overline{\mathcal{Y}}_{\overline{o}}$ as the set of all pixels $(h, w) \in \mathcal{I}$ inside the bounding box. In addition, each $\overline{\boldsymbol{y}}_{\overline{o}}$ is tagged with an object class $\overline{v}_{\overline{o}} \in \mathcal{S}$, from a set of object classes $\mathcal{S} = \{1, 2, ..., S\}$, with the $N_{\overline{\mathcal{O}}}$-dimensional vector of the classes of all objects being $\overline{\boldsymbol{v}} = (\overline{v}_{\overline{o}})$. For completeness, $\overline{v}_{\overline{o}} = \arg\max_{s \in \mathcal{S}} \overline{u}_{\overline{o},s}$, where $\overline{\boldsymbol{u}}_{\overline{o}} = (\overline{u}_{\overline{o},s}) \in \{0, 1\}^S$ is the one-hot representation of $\overline{v}_{\overline{o}}$. Similar to $\overline{\boldsymbol{y}}$ and $\overline{\boldsymbol{v}}$, the $(N_{\overline{\mathcal{O}}} \times S)$-dimensional matrix $\overline{\boldsymbol{u}} = (\overline{\boldsymbol{u}}_{\overline{o}})$ refers to the one-hot encoded class vectors of all objects. In our study, we focus solely on errors in the localizations $\overline{\boldsymbol{y}}$ and keep $\overline{\boldsymbol{v}}$ and $\overline{\boldsymbol{u}}$ untouched.

Autolabeling: Now, let $\boldsymbol{a} : \mathbb{I}^{H \times W \times C} \to \mathcal{I}^{N_{\widetilde{\mathcal{O}}} \times 2}$ be a localization autolabeling method which takes $\boldsymbol{x}$ as input and outputs localization labels

$$\boldsymbol{a}(\boldsymbol{x}) = \widetilde{\boldsymbol{y}} = \left(\widetilde{\boldsymbol{y}}_{\widetilde{o}}\right), \tag{3}$$

with top-left corner and bottom-right corner represented by $\widetilde{\boldsymbol{y}}_{\widetilde{o}} \in \mathcal{I}^2 \subset \mathbb{N}^4$ similar to $\overline{\boldsymbol{y}}_{\overline{o}}$ (1), object identifier $\widetilde{o} \in \widetilde{\mathcal{O}}$, and the set of autolabeled object identifiers $\widetilde{\mathcal{O}} = \{1, 2, ..., N_{\widetilde{\mathcal{O}}}\}$. Note that we introduced diacritic $\widetilde{(\,)}$ for autolabel output entities. Assuming correctly referenced objects ($\widetilde{o} = \overline{o}$ as well as $\widetilde{\mathcal{O}} = \overline{\mathcal{O}}$), we obtain

$$\widetilde{\boldsymbol{y}} = \overline{\boldsymbol{y}} + \boldsymbol{n}, \tag{4}$$

where $\boldsymbol{n} = (\boldsymbol{n}_{\overline{o}})$ represents object-specific label noise for top-left and bottom-right coordinates, similar to (1), injected by $\boldsymbol{a}$. In summary, although autolabeling method $\boldsymbol{a}$ is capable of capturing all objects in $\boldsymbol{x}$, we assume an uncertainty in its output represented by noise $\boldsymbol{n}$.

Deep neural network for object detection: Let $\boldsymbol{F} : \mathbb{I}^{H \times W \times C} \to (\mathcal{I}^2 \times \mathcal{S})^{N_{\mathcal{O}}}$ be a deep neural network for object detection which takes $\boldsymbol{x}$ as input and outputs object predictions $\boldsymbol{F}(\boldsymbol{x}) = (\boldsymbol{y}, \boldsymbol{v})$, with predicted object localizations $\boldsymbol{y} = (\boldsymbol{y}_o)$ and predicted object classes $\boldsymbol{v} = (v_o)$, where $o \in \mathcal{O} = \{1, 2, ..., N_{\mathcal{O}}\}$ is the object identifier. Each predicted pair $(\boldsymbol{y}_o, v_o)$ represents an object in image $\boldsymbol{x}$, with $\boldsymbol{y}_o \in \mathcal{I}^2 \subset \mathbb{N}^4$ similar to (1) and $v_o = \arg\max_{s \in \mathcal{S}} u_{o,s}$ being the respective object class with highest probability, where $\boldsymbol{u}_o = (u_{o,s}) \in [0, 1]^S$ is the predicted object class output vector. Note that $N_{\mathcal{O}} = N_{\overline{\mathcal{O}}}$ not necessarily holds, as we may have false positive or false negative detections. Network $\boldsymbol{F}$ is trained by minimizing

$$J = J^{\mathrm{cls}} + \lambda J^{\mathrm{reg}}, \tag{5}$$

with classification loss

$$J^{\mathrm{cls}} = -\frac{1}{N_{\overline{\mathcal{O}}}} \sum_{\overline{o} \in \overline{\mathcal{O}}} \left(\overline{\boldsymbol{u}}_{\overline{o}}\right)^{\mathrm{T}} \cdot \mathbf{log}\left(\boldsymbol{u}_{\overline{o}}\right), \tag{6}$$

where $\mathbf{log}(\cdot)$ is an element-wise logarithm and $(\,)^{\mathrm{T}}$ denotes the vector transpose. The regression loss J^{reg} (10) will be introduced in Sec. 3.3. As we will concentrate on the localization properties in this work, we will neglect the object class representation v_o and $\boldsymbol{u}_o$ from this point on and solely focus on the object positioning $\boldsymbol{y}_o$.

3.2. Localization Label Noise Injection (LLNI)

In order to perform studies on refinement of noisy localization labels, we require to have access to ground-truth labels $\overline{\boldsymbol{y}}$ and to a method injecting noise $\boldsymbol{n}$ to the ground-truth labels $\overline{\boldsymbol{y}}$, according to (4). For the former, we simply take a dataset $\mathcal{D}$ which is designed for object detection tasks. For the latter, we have two choices: We either collect a set of autolabeling methods $\mathcal{A}$, with $\boldsymbol{a} \in \mathcal{A}$, or we simply simulate $\mathcal{A}$. We proceed with the latter method as it comes with a major advantage, which is full control over the distribution of label noise $\boldsymbol{n}$ (4) as well as its strength denoted as ϵ. Thus, we can adjust the localization label error to our needs. This is not possible for autolabeling methods as each individual $\boldsymbol{a} \in \mathcal{A}$ might have its own inductive bias depending on the method itself, the underlying optimization procedure, the used dataset $\mathcal{D}$, etc.

Let's assume we have data $(\boldsymbol{x}, \overline{\boldsymbol{y}}) \in \mathcal{D}$, consisting of images $\boldsymbol{x}$ with respective object localization labels $\overline{\boldsymbol{y}} = (\overline{\boldsymbol{y}}_{\overline{o}})$ (1). To inject label noise to $\mathcal{D}$ we need to define a noise sampling scheme to obtain noisy labels $\widetilde{\boldsymbol{y}} = (\widetilde{\boldsymbol{y}}_{\overline{o}})$ (4). To this end, we formulate object-specific uniform distributions

$$\begin{aligned} U_h(-u_h, u_h), \text{ with } u_h &= \frac{\epsilon}{2}\overline{H}_{\overline{o}}, \\ U_w(-u_w, u_w), \text{ with } u_w &= \frac{\epsilon}{2}\overline{W}_{\overline{o}}, \end{aligned} \tag{7}$$

with noise strength $\epsilon \geq 0$, to deviate from the top-left corner and bottom-right corner coordinates $(\overline{h}_{\overline{o}}^{(\mathrm{L})}, \overline{w}_{\overline{o}}^{(\mathrm{L})})$ and $(\overline{h}_{\overline{o}}^{(\mathrm{R})}, \overline{w}_{\overline{o}}^{(\mathrm{R})})$ of $\overline{\boldsymbol{y}}_{\overline{o}}$ (1), respectively. Following (4), we then obtain

$$\begin{aligned} \big(\widetilde{h}_{\overline{o}}^{(\mathrm{L})}, \widetilde{w}_{\overline{o}}^{(\mathrm{L})}\big) &= \big(\overline{h}_{\overline{o}}^{(\mathrm{L})}, \overline{w}_{\overline{o}}^{(\mathrm{L})}\big) + \big(\Delta h_{\overline{o}}^{(\mathrm{L})}, \Delta w_{\overline{o}}^{(\mathrm{L})}\big), \\ \big(\widetilde{h}_{\overline{o}}^{(\mathrm{R})}, \widetilde{w}_{\overline{o}}^{(\mathrm{R})}\big) &= \big(\overline{h}_{\overline{o}}^{(\mathrm{R})}, \overline{w}_{\overline{o}}^{(\mathrm{R})}\big) + \big(\Delta h_{\overline{o}}^{(\mathrm{R})}, \Delta w_{\overline{o}}^{(\mathrm{R})}\big), \end{aligned} \tag{8}$$

where top-left and bottom-right coordinates of $\boldsymbol{n}_{\overline{o}}$ follow

$$\begin{aligned} \Delta h_{\overline{o}}^{(\mathrm{L})}, \Delta h_{\overline{o}}^{(\mathrm{R})} &\sim U_h(-u_h, u_h), \\ \Delta w_{\overline{o}}^{(\mathrm{L})}, \Delta w_{\overline{o}}^{(\mathrm{R})} &\sim U_w(-u_w, u_w). \end{aligned} \tag{9}$$

For completeness, similar to $\overline{\boldsymbol{y}}_{\overline{o}}$ (1), noisy localization label and label noise are defined as $\widetilde{\boldsymbol{y}}_{\overline{o}} = \big(\big(\widetilde{h}_{\overline{o}}^{(\mathrm{L})}, \widetilde{w}_{\overline{o}}^{(\mathrm{L})}\big), \big(\widetilde{h}_{\overline{o}}^{(\mathrm{R})}, \widetilde{w}_{\overline{o}}^{(\mathrm{R})}\big)\big)$ and $\boldsymbol{n}_{\overline{o}} = \big(\big(\Delta h_{\overline{o}}^{(\mathrm{L})}, \Delta w_{\overline{o}}^{(\mathrm{L})}\big), \big(\Delta h_{\overline{o}}^{(\mathrm{R})}, \Delta w_{\overline{o}}^{(\mathrm{R})}\big)\big)$, respectively. Very important to our approach and proposal of localization label noise injection is that the sampling (7), (9) is performed for each object $\overline{o} \in \overline{O}$ represented in $\overline{\boldsymbol{y}}$. In other words, *the power of the label noise* $\boldsymbol{n} = (\boldsymbol{n}_{\overline{o}})$ *in* (4) *depends on each object's bounding box size* (2). This ensures that deviations between $\overline{\boldsymbol{y}}$ and the resulting $\widetilde{\boldsymbol{y}}$ remain realistic and controllable via ϵ (7). This gives us a major advantage over the method used in [22], which only yields rough localization estimates with a very low intersection over union to the ground truth. In summary, following this procedure, we can create datasets with noisy labels denoted as $\mathcal{D}^{(\epsilon)}$ with noise strength ϵ.

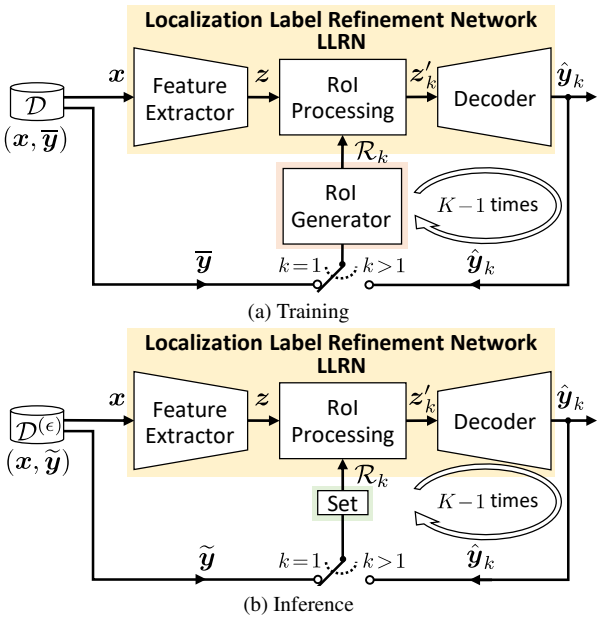


Figure 2. **Proposed single-pass ($k = 1$) and multi-pass ($k > 1$) localization label refinement**. *Training* (Fig. 2a): Image $\boldsymbol{x}$ is fed into a feature extractor yielding the latent feature representation $\boldsymbol{z}$. A subsequent region-of-interest (RoI) processing yields cropped feature representations $\boldsymbol{z}'_{k=1}$ using sampled region proposals $\mathcal{R}_{k=1}$ based on ground-truth localization label $\overline{\boldsymbol{y}}$. Finally, the decoder outputs a refined localization label $\hat{\boldsymbol{y}}_{k=1}$. For multi-pass localization label refinement we repeat this procedure by further using $\hat{\boldsymbol{y}}_{k\geq 1}$ instead of $\overline{\boldsymbol{y}}$. For simplicity, we do not display any subsequent loss computations. *Inference* (Fig. 2b): Noisy localization label $\widetilde{\boldsymbol{y}}$ instead of ground-truth localization label $\overline{\boldsymbol{y}}$ or refined localization label $\hat{\boldsymbol{y}}$ (multi-pass only) are used. Note that the RoI generator is not used during inference. Instead, a set operation yields $\mathcal{R}_{k=1} = \{\widetilde{\boldsymbol{y}}\}$ and $\mathcal{R}_k = \{\hat{\boldsymbol{y}}_k\}$ for $k > 1$.

3.3. Localization Label Refinement Network

Next, we design a method which is capable of refining noisy localization labels. As we are only interested in localizing objects without a subsequent classification, we build our method on top of an object localization framework. To this end, we choose the universal bounding box regressor (UBBR) [22] approach, which is a class-agnostic and anchor-free method for localization. As our objective is the refinement of noisy localization labels, we dub our proposed framework localization label refinement network (`LLRN`). Next, we give a brief overview of the most important components of our `LLRN`. We then proceed by introducing three possible variants of our `LLRN` depending on the refinement procedure: single-pass, multi-pass, and cascaded-pass refinement.

General overview: Fig. 2 displays the most important components of our LLRN. We neglect the subscript k in the following. Similar to conventional DNNs for object detection, our LLRN consists of a feature extractor, a region-of-interest (RoI) processing (in our case RoI align [16]), and a decoder. To generate regions of interest during training (Fig. 2a), we do not use a region proposal network (RPN) [40] but a RoI generator instead. The RoI generator takes object localizations ($\overline{\boldsymbol{y}}$, $\widetilde{\boldsymbol{y}}$, or $\hat{\boldsymbol{y}}$) as input and yields a set of M randomly sampled RoI proposals $\mathcal{R}$ per object following the sampling policy in [22], thus $N_{\overline{\mathcal{O}}} \cdot M$ in total. Note that the sampled RoI proposals can be understood as noisy localizations which are processed through a RoI processing followed by a refinement in the decoder. During inference (Fig. 2b) we use a set operation instead of the RoI generator which appends the object localization's input ($\overline{\boldsymbol{y}}$, $\widetilde{\boldsymbol{y}}$, or $\hat{\boldsymbol{y}}$) to the set of region proposals $\mathcal{R}$. Similar to the training phase, we feed the $N_{\overline{\mathcal{O}}}$ (noisy) object localizations through a RoI processing followed by a refinement in the decoder. Next, we elaborate on the single-pass and multi-pass LLRN.

Single-pass and multi-pass refinement: Fig. 2 gives an overview of both single-pass and multi-pass refinement. We start with the single-pass setting ($k = 1$). First, we distinguish between training (Fig. 2a) and inference (Fig. 2b). In both training and inference, the feature extractor takes $\boldsymbol{x}$ as input and outputs a latent feature representation $\boldsymbol{z}$ which is then fed to the RoI processing. During training, we use the ground-truth localization label $\overline{\boldsymbol{y}}$ and a RoI generator to produce a set of M randomly sampled RoI proposals $\mathcal{R}_{k=1}$ for each localization label in $\overline{\boldsymbol{y}}$. The RoI proposals are then forwarded to the RoI processing to output the RoI-cropped latent feature representation $\boldsymbol{z}'_{k=1}$ which yields a refined localization label $\hat{\boldsymbol{y}}_{k=1}$ after decoding. During inference, we do not generate RoI proposals anymore, but instead feed the noisy localization label $\widetilde{\boldsymbol{y}}$ to the RoI processing.

In the multi-pass setting ($k > 1$), we additionally feed the refined localization label $\hat{\boldsymbol{y}}_k$ back to the RoI processing and decoder until we reach the total number of iterations $k = K$. The idea is that $\hat{\boldsymbol{y}}_k$ is iteratively improved.

Cascaded-pass refinement: Fig. 3 depicts the cascaded localization label refinement network. It is inspired by the decoder structure of Cascaded R-CNN [3] and follows a similar procedure as the multi-pass localization label refinement network in Fig. 2b. The main difference to the multi-pass LLRN is that instead of using the same decoder in each iteration, we have K distinct decoders. This way, each decoder is able to handle a different quality of RoI proposals. In our experiments, we found that this is important as the multi-pass LLRN with a single decoder surprisingly did not improve the quality of localization labels but rather worsens it. This aligns with observations made in [3].

Loss functions during training: We train the single-pass, multi-pass, and cascaded-pass LLRN using a combination of smooth L_1 loss [13], J^{smooth}, and the generalized intersection-over-union (gIoU) loss [41], J^{gIoU}. During training, we combine the losses to

$$J^{\text{reg}} = J^{\text{smooth}} + J^{\text{gIoU}}. \tag{10}$$

Please refer to Supplement Sec. 1 for a detailed description of J^{smooth} and J^{gIoU}. For the multi-pass and cascaded-pass LLRN, we extend (10) with weight factors. In particular, we have

$$J^{\text{reg}} = \sum_{k=1}^{K} 2^{k-1} J_k^{\text{reg}}, \tag{11}$$

with the J_k^{reg} loss (10) being computed for iteration k (cf. Fig. 2a) or the k-th decoder (cf. Fig. 3). Note that for training the Cascade R-CNN we also use loss (10) in (5).

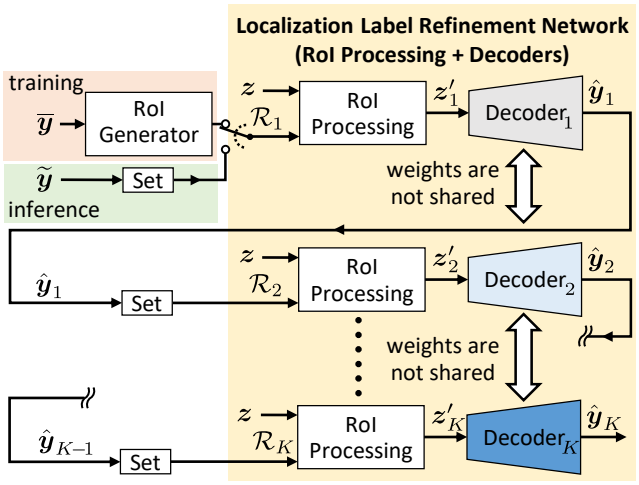
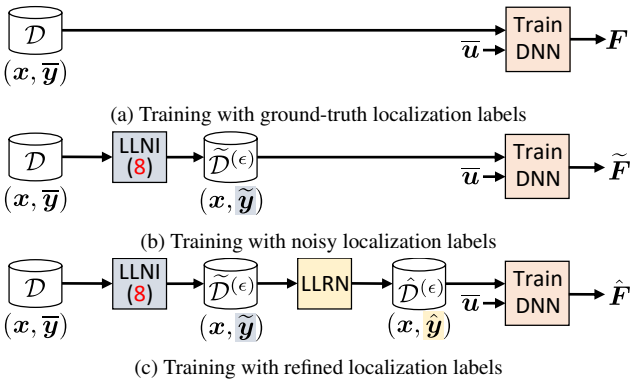


Figure 3. **Proposed cascaded-pass localization label refinement**. *Training:* Latent feature representation $\boldsymbol{z}$ and sampled region proposals $\mathcal{R}_1$ based on ground-truth localization label $\overline{\boldsymbol{y}}$ are fed into a region-of-interest (RoI) processing yielding cropped feature representations $\boldsymbol{z}'_1$. A subsequent decoder (subscript 1) outputs a refined localization label $\hat{\boldsymbol{y}}_1$. Next, $\hat{\boldsymbol{y}}_1$ is fed into a second RoI processing yielding $\mathcal{R}_2$ and $\boldsymbol{z}'_2$, respectively. A second distinct decoder (subscript 2) outputs a refined localization label $\hat{\boldsymbol{y}}_2$. This process is repeated exactly K times to obtain the final refined localization label $\hat{\boldsymbol{y}}_K$. *Inference:* Noisy localization label $\widetilde{\boldsymbol{y}}$ instead of ground-truth localization label $\overline{\boldsymbol{y}}$ is used. Note that the RoI generator is disabled during inference. Instead, a set operation yields $\mathcal{R}_{k=1} = \{\widetilde{\boldsymbol{y}}\}$ and $\mathcal{R}_k = \{\hat{\boldsymbol{y}}_k\}$ for $k > 1$. Best viewed in color.

3.4. Proposed Evaluation Setup

We evaluate our proposed method by performing three distinct trainings. Fig. 4 gives an overview. The standard training is displayed in Fig. 4a, where we simply train a model on the original dataset $\mathcal{D}$ yielding network $\boldsymbol{F}$. Next, in Fig. 4b, we incorporate our proposed localization label noise injection (LLNI, Sec. 3.2) to obtain the noisy dataset



(a) Training with ground-truth localization labels

(b) Training with noisy localization labels

(c) Training with refined localization labels

Figure 4. **Proposed training schemes for subsequent evaluation**. Each training uses the ground-truth class labels $\overline{\boldsymbol{u}}$ of original dataset $\mathcal{D}$. Fig. 4a: Standard training with original dataset $\mathcal{D}$ and $\overline{\boldsymbol{u}}$ yielding model $\boldsymbol{F}$. Dataset $\mathcal{D}$ consists of images $\boldsymbol{x}$ and ground-truth localization labels $\overline{\boldsymbol{y}}$. Fig. 4b: Training with noisy dataset $\widetilde{\mathcal{D}}^{(\epsilon)}$ and $\overline{\boldsymbol{u}}$ yielding model $\widetilde{\boldsymbol{F}}$. Note that noisy localization labels $\widetilde{\boldsymbol{y}}$ are used, where the noise of strength ϵ (7) is injected using our proposed localization label noise injection (LLNI) (8). Fig. 4c: Training with refined dataset $\hat{\mathcal{D}}^{(\epsilon)}$ and $\overline{\boldsymbol{u}}$ yielding model $\hat{\boldsymbol{F}}$. Our proposed localization label refinement network (LLRN) refines the noisy localization labels $\widetilde{\boldsymbol{y}}$ to refined localization labels $\hat{\boldsymbol{y}}$.

$\mathcal{D}^{(\epsilon)}$. We then train another model on noisy dataset $\widetilde{\mathcal{D}}^{(\epsilon)}$ yielding $\widetilde{\boldsymbol{F}}$. Lastly, in Fig. 4c, we additionally introduce our proposed localization label refinement network (`LLRN`) to obtain the refined noisy dataset $\hat{\mathcal{D}}^{(\epsilon)}$ and again train a model yielding $\hat{\boldsymbol{F}}$. Note that for the latter, we keep the subscript ϵ on purpose to indicate that the localization label refinement was performed on $\widetilde{\mathcal{D}}^{(\epsilon)}$ having (original) noise strength ϵ. That being said, we perform ablation studies on various noise strengths ϵ (7). The evaluation is performed using standard object detection metrics, which will be introduced in the evaluation details.

4. Experimental Setup

In the following, we introduce our experimental setup. Further details can be found in our code which will be published upon acceptance.

Employed datasets: Tab. 1 lists all data splits in our experiments. All our experiments are performed on Pascal VOC 2012 [9]. We extend the original dataset by noisy versions using our proposed localization label noise injection (Sec. 3.2) and refined noisy versions using our proposed `LLRN` (Sec. 3.3).

We train an object detection network on either the original training set $\mathcal{D}_{\text{train}}$, our noisy version $\widetilde{\mathcal{D}}^{(\epsilon)}_{\text{train}}$ having noise strength ϵ, or the refined noisy version $\hat{\mathcal{D}}^{(\epsilon)}_{\text{train}}$ (cf. Fig. 4). After training, we evaluate the performance using the original validation set $\mathcal{D}_{\text{val}}$.

On the other hand, all variations of `LLRN` are solely trained on the original training set $\mathcal{D}_{\text{train}}$. After training, we perform evaluations on the original validation set $\mathcal{D}_{\text{val}}$ and our noisy version $\widetilde{\mathcal{D}}^{(\epsilon)}_{\text{val}}$. Taking $\widetilde{\mathcal{D}}^{(\epsilon)}_{\text{val}}$ as input, our `LLRN` yields $\hat{\mathcal{D}}^{(\epsilon)}_{\text{val}}$, which we compare to the original validation set $\mathcal{D}_{\text{val}}$. Further, the refined noisy version $\hat{\mathcal{D}}^{(\epsilon)}_{\text{train}}$ of the original training set $\mathcal{D}_{\text{train}}$ is created by processing $\widetilde{\mathcal{D}}^{(\epsilon)}_{\text{train}}$ using our trained `LLRN`.

Table 1. **Data splits** used in our experiments, with noise strength $\epsilon \in \mathcal{E} = \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$. Note $\widetilde{\mathcal{D}}^{(\epsilon=0)}_{[.]} = \mathcal{D}_{[.]}$.

Dataset	Official subset	#Images	Symbol
Pascal VOC 2012 [9]	train	5,717	$\mathcal{D}_{\text{train}}$
		5,717	$\widetilde{\mathcal{D}}^{(\epsilon)}_{\text{train}}$
		5,717	$\hat{\mathcal{D}}^{(\epsilon)}_{\text{train}}$
	val	5,823	$\mathcal{D}_{\text{val}}$
		5,823	$\widetilde{\mathcal{D}}^{(\epsilon)}_{\text{val}}$
		5,823	$\hat{\mathcal{D}}^{(\epsilon)}_{\text{val}}$

Network architecture: We deploy the `Cascade R-CNN` [3] as our object detection network. We use an ImageNet-pretrained [44] `ConvNeXt-tiny` [27] backbone, followed by a feature pyramid network (FPN) [24] with four feature pyramid levels, RoI Align [16], and a subsequent cascaded decoder [3]. We follow the standard procedure for feature pyramid level selection [24]. The output of the selected feature pyramid is fed into the `Cascade R-CNN` decoder. It uses three cascades, each based on `Faster R-CNN` [40] having IoU thresholds 0.5, 0.75, or 0.9, respectively. The output of the last cascade is used as the final output.

Our `LLRN` follows a similar structure. The single-pass and multi-pass `LLRN` also use an ImageNet-pretrained [44] `ConvNeXt-tiny` [27] backbone, followed by a feature pyramid network (FPN) [24] with four levels and standard feature pyramid level selection [24] with subsequent RoI Align [16]. The decoder consists of three consecutive fully connected layers with 4096, 4096, and four outputs, respectively. Note that the four outputs correspond to refined localization labels $\hat{\boldsymbol{y}}$ (cf. Fig. 2). Our cacsaded-pass `LLRN` builds on top of the architectures of the single-pass and multi-pass `LLRN` and employs cascaded decoder heads as in [3]. In our experiments, we found out that it is sufficient to use $K = 3$ cascaded decoders.

Training details: We train the `Cascade R-CNN` with a frozen backbone following [3]. In particular, we train for 36 epochs using the AdamW [28] optimizer with learning rate 10^{-4}, weight decay 0.05, loss weight $\lambda = 1$ (5), and a batch size of 4. Regarding data augmentations, we resize the images to 512×512 and do a random horizontal flipping.

For training our `LLRN`, we follow mostly [22]. In par-

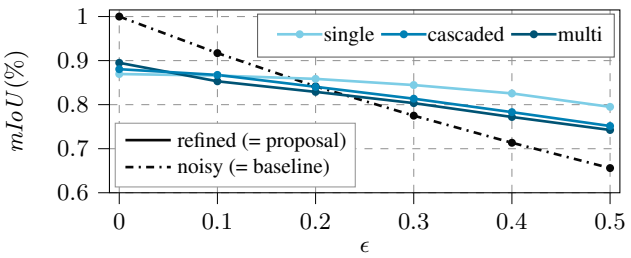


Figure 5. **Localization label quality for noisy and refined data.** Mean intersection-over-union ($mIoU$) (13) between ground truth data $\mathcal{D}_{\text{val}}$ and noisy data $\widetilde{\mathcal{D}}_{\text{val}}^{(\epsilon)}$, and between ground truth data $\mathcal{D}_{\text{val}}$ and refined data $\hat{\mathcal{D}}_{\text{val}}^{(\epsilon)}$ for different noise strengths ϵ (7) are reported. The refined dataset $\hat{\mathcal{D}}_{\text{val}}^{(\epsilon)}$ is created by using our single-pass, multi-pass, or cascaded-pass localization label refinement network (LLRN).

ticular, we train for 100 epochs using the AdamW [28] optimizer with learning rate 10^{-4}, weight decay 0.05, and a batch size of 32. Further, we resize the images to 512×512. The training is performed with a frozen backbone. Note that during training, the RoI generator (cf. Figs. 2 and 3) samples $M = 50$ RoI proposals following the sampling policy in [22] with sampling hyperparameters $\alpha = 0.15$ and $\beta = 0.25$.

Evaluation details: To evaluate our approach, we make use of the mean intersection-over-union ($mIoU$) and the mean average precision (mAP). We define class-specific IoU_s as

$$IoU_s = \frac{1}{|\overline{\mathcal{O}}_s|} \sum_{o \in \overline{\mathcal{O}}_s} \frac{|\mathcal{Y}_{o,s} \cap \overline{\mathcal{Y}}_{o,s}|}{|\mathcal{Y}_{o,s} \cup \overline{\mathcal{Y}}_{o,s}|}, \tag{12}$$

where $\mathcal{Y}_{o,s}$ is the set of pixels $(h, w) \in \mathcal{I}$ enclosed by $\boldsymbol{y}_o$ having class s, and $\overline{\mathcal{Y}}_{o,s}$ is the set of pixels $(h, w) \in \mathcal{I}$ enclosed by ground truth $\overline{\boldsymbol{y}}_o$ with class s. Further, set $\overline{\mathcal{O}}_s$ refers to all objects of class s. Similarly, we also compute the IoU_s between the set of pixels $\widetilde{\mathcal{Y}}_{o,s}$ of noisy localization label $\widetilde{\boldsymbol{y}}_o$ and the ground-truth set of pixels $\overline{\mathcal{Y}}_{o,s}$ on the one hand, and between $\hat{\mathcal{Y}}_{o,s}$ of the refined localization label $\hat{\boldsymbol{y}}_o$ and the ground-truth set of pixels $\overline{\mathcal{Y}}_{o,s}$ on the other hand. Finally, the mean intersection-over-union is defined as

$$mIoU = \frac{1}{S} \sum_{s \in \mathcal{S}} IoU_s. \tag{13}$$

We report the object detection performance of the Cascade R-CNN using mAP_κ [25] defined as

$$mAP_\kappa = \frac{1}{S} \sum_{s \in \mathcal{S}} AP_{\kappa,s}, \tag{14}$$

where mAP_κ is the mean average precision for IoU threshold κ, and $AP_{\kappa,s}$ is the average precision for IoU threshold κ and class s.

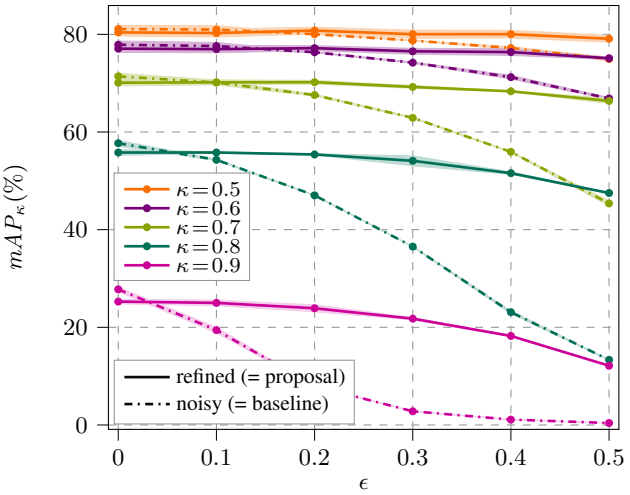


Figure 6. **Object detection performance on noisy and refined data**. Mean average precision (mAP_κ) (14), $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$, on $\mathcal{D}_{\text{val}}$ is reported. We trained the Cascade R-CNN on noisy datasets $\widetilde{\mathcal{D}}_{\text{train}}^{(\epsilon)}$ and refined datasets $\hat{\mathcal{D}}_{\text{train}}^{(\epsilon)}$ with different (initial) noise strengths ϵ (7). Dataset $\hat{\mathcal{D}}_{\text{train}}^{(\epsilon)}$ is created using our single-pass LLRN.

5. Experimental Evaluation and Discussion

In the following, we discuss our experimental results.

5.1. Baseline Performance

First, we report $mIoU$ (13) between ground-truth localization labels $\overline{\boldsymbol{y}}$ and noisy localization labels $\widetilde{\boldsymbol{y}}$ with various noise strengths $\epsilon \in \mathcal{E}$ using our validation datasets $\mathcal{D}_{\text{val}}$ and $\widetilde{\mathcal{D}}_{\text{val}}^{(\epsilon)}$. The results are shown in Fig. 5 (dashed-dotted line). We observe an almost linear decline in $mIoU$ between ground-truth localization labels $\overline{\boldsymbol{y}}$ and noisy localization labels $\widetilde{\boldsymbol{y}}$ when increasing the noise strength ϵ of our LLNI.

Next, we use $\mathcal{D}_{\text{train}}$ and $\widetilde{\mathcal{D}}_{\text{train}}^{(\epsilon)}$, $\epsilon \in \mathcal{E}$, to train distinct Cascade R-CNN (cf. Figs. 4a and 4b). We evaluate the performance using mAP_κ with IoU thresholds $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. The results are shown in Fig. 6 (dashed-dotted lines). We observe that increasing the noise strength ϵ of our LLNI not only leads to a drop in $mIoU$ between ground-truth localization labels $\overline{\boldsymbol{y}}$ and noisy localization labels $\widetilde{\boldsymbol{y}}$ (cf. Fig. 5, dashed-dotted line) but also to a drop in mAP of Cascade R-CNN over all κ. This is indeed a problem as we cannot expect to have perfect localization labels at all times. We further observe that especially for mAP_κ with high IoU thresholds ($\kappa \geq 0.7$) the performance drop is severe.

5.2. Label Refinement

Next, we investigate to what extent our proposed LLRN is capable of mitigating the effect of localization label noise.

To this end, we first take a look at the $mIoU$ after localization label refinement using either the single-pass, multi-pass, or cascaded-pass LLRN. In Fig. 5 (solid lines), we report the $mIoU$ between ground-truth localization labels $\overline{\boldsymbol{y}}$ and refined localization labels $\hat{\boldsymbol{y}}$ with various (original) noise strengths $\epsilon \in \mathcal{E}$ using our validation datasets $\mathcal{D}_{\text{val}}$ and $\hat{\mathcal{D}}_{\text{val}}^{(\epsilon)}$. We observe that for noise strengths $\epsilon < 0.2$ all variations of our LLRN are not able to improve nor retain the $mIoU$. However, for $\epsilon > 0.2$ all LLRN setups are able to improve the $mIoU$, with the single-pass LLRN yielding the best results with an $mIoU$ improvement of up to 13.9% absolute over the baseline (for $\epsilon = 0.5$). This is actually quite surprising as we expected both multi-pass LLRN and cascaded-pass LLRN to have better performance than single-pass LLRN. Examples for refinement are depicted in Fig. 1. Further examples can be found in the Supplement.

5.3. Performance on Refined Localization Labels

Finally, we use $\hat{\mathcal{D}}_{\text{train}}^{(\epsilon)}$, $\epsilon \in \mathcal{E}$, generated by our best method, i.e., single-pass LLRN, to train distinct Cascade R-CNN (cf. Fig. 4c). We evaluate the performance using mAP_κ, with $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. The results are shown in Fig. 6 (solid lines), where the shaded area indicates the standard deviation over three runs. We observe that our single-pass LLRN improves mAP_κ over almost all ϵ by a large margin, for $\epsilon = 0.5$ and $\kappa = 0.8$ even up to 34.2% absolute mAP. However, we also observe that training with refined ground-truth data ($\epsilon = 0$) slightly degrades the performance. We provide the exact value of each data point in the Supplement, Table 2.

Although Fig. 5 indicates that single-pass LLRN is superior to multi-pass and cascaded-pass LLRN, we were still interested to what extent this superiority is reflected in the mAP_κ. To this end, we train the Cascade R-CNN using refined localization labels from multi-pass LLRN and the cascaded-pass LLRN. Fig. 7 depicts the mAP_κ for both LLRN setups and our single-pass LLRN, with $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. Similar to Fig. 6, the shaded area indicates the standard deviation over three runs. We observe that single-pass LLRN (solid lines) is in most cases superior to or on par with both multi-pass LLRN (dotted lines) and cascaded-pass LLRN (dashed lines). The highest deviation is measured for $\epsilon = 0.5$ and $\kappa = 0.8$ with about 10% absolute higher mAP on average. For cases $\kappa = 0.5...0.6$ and $\epsilon = 0...0.3$, we find that the cascaded-pass LLRN is slightly better on average than the single-pass LLRN (up to 1.1% absolute mAP). However, we also observe that we have a higher variance over three runs (shaded area) which is why we consider these deviations to be not significant. We conclude that the single-pass LLRN is better than both multi-pass LLRN and cascaded-pass LLRN.

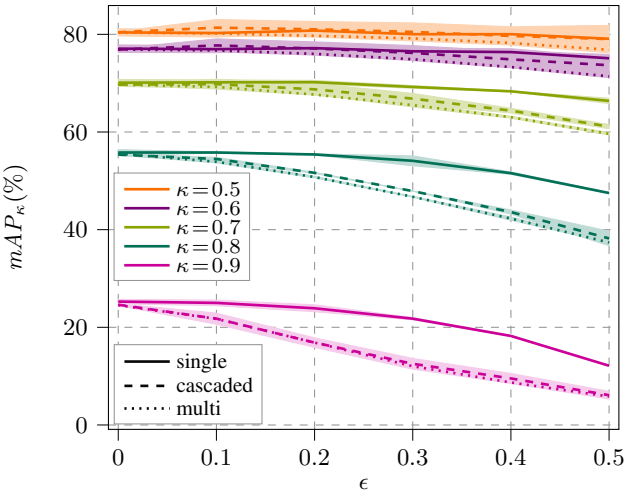


Figure 7. **Object detection performance on refined data by our three investigated LLRN methods**. Mean average precision (mAP_κ) (14), $\kappa \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$, on $\mathcal{D}_{\text{val}}$ is reported. We trained the Cascade R-CNN on noisy datasets $\tilde{\mathcal{D}}_{\text{train}}^{(\epsilon)}$ and refined datasets $\hat{\mathcal{D}}_{\text{train}}^{(\epsilon)}$ with different (original) noise strengths ϵ (7). Dataset $\hat{\mathcal{D}}_{\text{train}}^{(\epsilon)}$ is created using our single-pass LLRN, multi-pass LLRN, or cascaded-pass LLRN.

6. Conclusions

In this paper, we address localization label errors stemming from autolabeling. In particular, we first propose a *method to generate localization label errors* by inducing label noise to ground-truth localization labels of Pascal VOC 2012. Different to conventional autolabeling methods, this gives us the freedom to adjust the amount of noise in the labels offering a more insightful evaluation. Using the Cascade R-CNN, we were able to show that *training with localization label errors indeed leads to a worse final performance*. Further, we propose a *method to refine noisy localization labels* dubbed localization label refinement network (LLRN) to address the performance gap. We show that our proposed method indeed refines the noisy localization coming from our localization label error generator. In numbers, we were able to restore up to 13.9% absolute $mIoU$ towards the ground-truth localization labels. A subsequent training of Cascade R-CNN with refined localization labels shows improved performance by a large margin, improving $mAP_{0.8}$ up to 34.2% absolute. Our study introduces a practically relevant and challenging task which we dub "refinement of noisy localization labels" and also sets a first benchmark on Pascal VOC 2012.

References

[1] Dana Angluin and Philip Laird. Learning From Noisy Examples. *Machine Learning*, 2(4):343–370, Apr. 1988. 1, 2

[2] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A Closer Look at Memorization in Deep Networks. In *Proc. of ICML*, pages 233–242, Sydney, Australia, July 2017. 1, 2

[3] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving Into High Quality Object Detection. In *Proc. of CVPR*, pages 6154–6162, Salt Lake City, UT, USA, June 2018. 2, 3, 5, 6

[4] Jiale Cao, Yanwei Pang, Jungong Han, and Xuelong Li. Hierarchical Shot Detector. In *Proc. of ICCV*, pages 9704–9713, Seoul, South Korea, Oct. 2021. 1

[5] Filipe R. Cordeiro and Gustavo Carneiro. A Survey on Deep Learning with Noisy Labels: How to Train Your Model When you Cannot Trust on the Annotations? In *Proc. of SIBGRAPI*, pages 9–16, virtual conference, Nov. 2020. 1, 2

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Proc. of CVPR*, pages 3213–3223, Las Vegas, NV, USA, June 2016. 1

[7] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *Proc. of CVPR*, pages 886–893, San Diego, CA, USA, June 2005. 1, 2

[8] Li Deng. The MNIST Database of Handwritten Digit Images for Machine Learning Research. *IEEE Signal Processing Magazine*, 29(6):141–142, Nov. 2012. 2

[9] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015. 1, 2, 6

[10] Pedro F. Felzenszwalb, Ross Girshick, David McAllester, and Deva Ramanan. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept. 2010. 1, 2

[11] Tim Fingscheidt, Hanno Gottschalk, and Sebastian Houben, editors. *Deep Neural Networks and Data for Automated Driving: Robustness, Uncertainty Quantification, and Insights Towards Safety*. Springer International Publishing, Cham, 2022. 2

[12] Aritra Ghosh, Himanshu Kumar, and P. S. Sastry. Robust Loss Functions Under Label Noise for Deep Neural Networks. In *Proc. of AAAI*, pages 1919–1925, San Francisco, CA, USA, Feb. 2017. 2

[13] Ross Girshick. Fast R-CNN. In *Proc. of ICCV*, pages 1440–1448, Las Condes, Chile, Dec. 2015. 1, 2, 5

[14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *Proc. of CVPR*, pages 580–587, Columbus, OH, USA, June 2014. 2

[15] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-Teaching: Robust Training of Deep Neural Networks With Extremely Noisy Labels. In *Proc. of NeurIPS*, pages 8536–8546, Montréal, QC, Canada, Dec. 2018. 1, 2

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proc. of ICCV*, pages 2980–2988, Venice, Italy, Oct. 2017. 5, 6

[17] Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang. Bounding Box Regression with Uncertainty for Accurate Object Detection. In *Proc. of CVPR*, pages 2888–2897, Long Beach, CA, USA, June 2019. 1, 2

[18] Sara Hooker, Aaron Courville, Yann Dauphin, and Andrea Frome. What Do Compressed Deep Neural Networks Forget? *arXiv*, 1911.05248:1–20, Sept. 2019. 2

[19] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, Toronto, ON, Canada, Apr. 2009. 1, 2

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Proc. of NIPS*, pages 1106–1114, Lake Tahoe, NV, USA, Dec. 2012. 2

[21] Fabian Küppers, Anselm Haselhoff, Jan Kronenberger, and Jonas Schneider. Confidence Calibration for Object Detection and Segmentation. In Tim Fingscheidt, Hanno Gottschalk, and Sebastian Houben, editors, *Deep Neural Networks and Data for Automated Driving: Robustness, Uncertainty Quantification, and Insights Towards Safety*, pages 225–250, Cham, 2022. Springer International Publishing. 2

[22] Seungkwan Lee, Suha Kwak, and Minsu Cho. Universal Bounding Box Regression and Its Applications. In *Proc. of ACCV*, pages 373–387, Perth, Australia, Dec. 2018. 2, 3, 4, 5, 6, 7

[23] Junnan Li, Richard Socher, and Steven C. H. Hoi. DivideMix: Learning With Noisy Labels as Semi-Supervised Learning. In *Proc. of ICLR*, pages 1–14, virtual conference, Apr. 2020. 2

[24] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *Proc. of CVPR*, pages 2117–2125, Honolulu, HI, USA, July 2017. 6

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *Proc. of ECCV*, pages 740–755, Zurich, Switzerland, Sept. 2014. 2, 7

[26] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proc. of ICCV*, pages 10012–10022, virtual conference, Oct. 2021. 1

[27] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *Proc. of CVPR*, pages 11976–11986, New Orleans, LA, USA, June 2022. 2, 6

[28] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *Proc. of ICLR*, pages 1–18, New Orleans, LA, USA, May 2019. 6, 7

[29] David G. Lowe. Object Recognition from Local Scale-Invariant Features. In *Proc. of ICCV*, pages 1150–1157, Kerkyra, Greece, Sept. 1999. 1

[30] Jiafeng Mao, Qing Yu, and Kiyoharu Aizawa. Noisy Localization Annotation Refinement For Object Detection. In *Proc. of ICIP*, pages 2006–2010, virtual conference, Oct. 2020. 3

[31] Jiafeng Mao, Qing Yu, Yoko Yamakata, and Kiyoharu Aizawa. Noisy Annotation Refinement for Object Detection. In *Proc. of BMVC*, pages 1–12, virtual conference, Nov. 2021. 3

[32] Duc Tam Nguyen, Chaithanya Kumar Mummadi, Thi Phuong Nhung Ngo, Thi Hoai Phuong Nguyen, Laura Beggel, and Thomas Brox. SELF: Learning to Filter Noisy Labels With Self-Ensembling. In *Proc. of ICLR*, pages 1–16, virtual conference, Apr. 2020. 2

[33] Curtis Northcutt, Anish Athalye, and Jonas Mueller. Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks. In *Proc. of NeurIPS - Datasets and Benchmarks*, pages 1–13, virtual conference, Dec. 2021. 1, 2

[34] Curtis Northcutt, Lu Jiang, and Isaac Chuang. Confident Learning: Estimating Uncertainty in Dataset Labels. *Journal of Artificial Intelligence Research*, 70:1373–1411, May 2021. 2

[35] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making Deep Neural Networks Robust to Label Noise: A Loss Correction Approach. In *Proc. of CVPR*, pages 1944–1952, Honolulu, HI, USA, July 2017. 2

[36] Christopher Plachetka, Jenny Fricke, Marvin Klingner, and Tim Fingscheidt. DNN-Based Recognition of Pole-Like Objects in LiDAR Point Clouds. In *Proc. of ITSC*, pages 2889–2896, Montreal, QC, Canada, Sept. 2021. 2

[37] Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid Training Data Creation with Weak Supervision. *International Journal on Very Large Data Bases*, 29:709–730, May 2020. 2

[38] Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. YouTube-BoundingBoxes: A Large High-Precision Human-Annotated Data Set for Object Detection in Video. In *Proc. of CVPR*, pages 5296–5305, Honolulu, HI, USA, July 2017. 2

[39] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training Deep Neural Networks on Noisy Labels with Bootstrapping. *arXiv*, 1412.6596:1–11, Apr. 2015. 1, 2

[40] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection With Region Proposal Networks. In *Proc. of NeurIPS*, pages 91–99, Montréal, QC, Canada, Dec. 2015. 1, 2, 5, 6

[41] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression. In *Proc. of CVPR*, pages 658–66, Long Beach, CA, USA, June 2019. 5

[42] Tobias Riedlinger, Marius Schubert, Karsten Kahl, and Matthias Rottmann. Uncertainty Quantification for Object Detection: Output- and Gradient-Based Approaches. In Tim Fingscheidt, Hanno Gottschalk, and Sebastian Houben, editors, *Deep Neural Networks and Data for Automated Driving: Robustness, Uncertainty Quantification, and Insights Towards Safety*, pages 251–275, Cham, 2022. Springer International Publishing. 2

[43] Matthias Rottmann and Marco Reese. Automated Detection of Label Errors in Semantic Segmentation Datasets via Deep Learning and Uncertainty Quantification. In *Proc. of WACV*, pages 3214–3223, Waikoloa, HI, USA, Jan. 2023. 1

[44] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, Dec. 2015. 1, 2, 6

[45] Vaishaal Shankar, Rebecca Roelofs, Horia Mania, Alex Fang, Benjamin Recht, and Ludwig Schmidt. Evaluating Machine Accuracy on ImageNet. In *Proc. of ICML*, pages 8634–8644, virtual conference, July 2020. 2

[46] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proc. of ICLR*, pages 1–14, San Diego, CA, USA, May 2015. 2

[47] Peng Tang, Xinggang Wang, Xiang Bai, and Wenyu Liu. Multiple Instance Detection Network With Online Instance Classifier Refinement. In *Proc. of CVPR*, pages 2843–2851, Honolulu, HI, USA, July 2017. 3

[48] Antti Tarvainen and Harri Valpola. Mean Teachers Are Better Role Models: Weight-Averaged Consistency Targets Improve Semi-Supervised Deep Learning Results. In *Proc. of NIPS*, pages 1195–1204, Long Beach, CA, USA, Dec. 2017. 1, 2

[49] Paul Viola and Michael Jones. Rapid Object Detection Using a Boosted Cascade of Simple Features. In *Proc. of CVPR*, pages 511–518, Kauai, HI, USA, Dec. 2001. 2

[50] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning From Massive Noisy Labeled Data for Image Classification. In *Proc. of CVPR*, pages 2691–2699, Boston, MA, USA, June 2015. 2

[51] Doris Xin, Eva Yiwei Wu, Doris Jung-Lin Lee, Niloufar Salehi, and Aditya Parameswaran. Whither AutoML? Understanding the Role of Automation in Machine Learning Workflows. In *Proc. of CHI*, pages 1–16, virtual conference, May 2021. 2

[52] Jieyu Zhang, Cheng-Yu Hsieh, Chao Zhang Yue Yu, and Alexander Ratner. A Survey on Programmatic Weak Supervision. *arXiv*, 2202.05433:1–8, Feb. 2022. 2

[53] Shikun Zhang, Omid Jafari, and Parth Nagarkar. A Survey on Machine Learning Techniques for Auto Labeling of Video, Audio, and Text Data. *arXiv*, 2109.03784:1–13, Sept. 2021. 1

[54] Weihe Zhang, Yali Wang, and Yu Qiao. MetaCleaner: Learning to Hallucinate Clean Representations for Noisy-Labeled Visual Recognition. In *Proc. of CVPR*, pages 7365–7374, Long Beach, CA, USA, June 2019. 2