CATEGORISE: An Automated Framework for Utilizing the Workforce of the Crowd for Semantic Segmentation of 3D Point Clouds

Michael Kölle(✉), Volker Walter, Ivan Shiller, and Uwe Soergel

Institute for Photogrammetry, University of Stuttgart, Geschwister-Scholl-Str. 24D, 70174 Stuttgart, Germany
{michael.koelle,volker.walter,ivan.shiller, uwe.soergel}@ifp.uni-stuttgart.de

Abstract. This paper discusses the CATEGORISE framework meant for establishing a supervised machine learning model without i) the requirement of training labels generated by experts, but by the crowd instead and ii) the labor-intensive manual management of crowdsourcing campaigns. When crowdworking is involved, quality control of results is essential. This control is an additional overhead for an expert diminishing the attractiveness of crowdsourcing. Hence, the requirement for an automated pipeline is that both quality control of labels received and the overall employment process of the crowd can run without the involvement of an expert. To further reduce the number of necessary labels and by this human labor (of the crowd), we make use of Active Learning. This also minimizes time and costs for annotation. Our framework is applied for semantic segmentation of 3D point clouds. We firstly focus on possibilities to overcome the aforementioned challenges by testing different measures for quality control in context of real crowd campaigns and develop the CATEGORISE framework for full automation capabilities, which leverages the *microWorkers* platform. We apply our approach to two different data sets of different characteristics to prove the feasibility of our method both in terms of accuracy and automation. We show that such a process results in an accuracy comparable to that of Passive Learning. Instead of labeling or administrative responsibilities, the operator solely monitors the progress of the iteration, which runs and terminates (using a proper stopping criterion) in an automated manner.

Keywords: Crowdsourcing · Labeling · Automation · Active Learning · Semantic segmentation · Random Forest · 3D point clouds

1 Introduction

At latest since the emergence of Convolutional Neural Networks (CNNs), it has become clear that supervised machine learning (ML) systems are severely

C. Bauckhage et al. (Eds.): DAGM GCPR 2021, LNCS 13024, pp. 633–648, 2021.
https://doi.org/10.1007/978-3-030-92659-5_41

hindered by the lack of labeled training data. To boost the development of such systems, many labeled benchmark data sets were crafted both in the domain of imagery [7,17] and 3D point clouds [8,28]. However, these data sets might be insufficient for new tasks, for instance in remote sensing (e.g., airborne vs. terrestrial systems). Although labeled data sets are also present in the remote sensing domain [18,24,27], due to the rapid development of new sensor types and system design (e.g., for airborne laser scanning (ALS): conventional ALS [24], UAV laser scanning [18] often enriched by imaging sensors, single photon LiDAR [23]) labeled data might be quickly out of date requiring new labeling campaigns. Although transfer learning might help to reduce the amount of task-specific ground truth data (GT) by building upon GT from another domain, it is often necessary to generate one's own training data [25].

Such a labeling process is typically carried out by experts [18,24], which is both time-consuming and cost-intensive. Hence, the idea is to outsource this tedious task to others in order to free experts from such duties in the sense of crowdsourcing. In this context many platforms for carrying out crowd campaigns (such as *Amazon Mechanical Turk* [5] or *microWorkers* [12]) have emerged. In addition to such crowdsourcing platforms, also services which offer to take over the complete labeling process come into focus (such as Google's *Data Labeling Service* [9]). Although the expert loses control of the labeling campaign (outsourcing vs. crowdsourcing), the justification of the latter is to avoid time-consuming campaign management (hiring, instructing, checking and paying crowdworkers). Hence, for crowdsourcing to remain competitive, the aforementioned tasks ought to be automated.

Another major challenge of employing crowdworkers is quality control of results received from the crowd. Walter & Soergel [33] have shown that data quality varies significantly. Therefore an employer either needs to check results manually (which might become even more labor-intensive than the actual labeling task) or rely on proper means for quality control. This problem is most pronounced in context of paid crowdsourcing, where often the sole aim of crowdworkers is to make money as fast as possible and there might be even malicious crowdworkers. In this regard, motivation and consequently quality of work in paid crowdsourcing differs significantly from volunteered crowdsourcing (or volunteered geographic information to be precise). In case of the latter, workers are intrinsically motivated and aim to contribute to a greater cause, which is for example freely available map data in case of *OpenStreetMap* [4]. Nevertheless, paid crowdsourcing could already be successfully used for annotation of airborne imagery [33], detecting and describing trees in 3D point clouds [32] or labeling of individual points according to a specified class catalog [15]. To minimize labeling effort (for the crowd), a common approach is Active Learning (AL).

2 Related Work on Active Learning

AL aims on selecting only most informative instances justifying manual annotation effort [29]. In Mackowiak et al. [22] querying such instances (2D image

subsets) is accomplished by combining both predictive uncertainty and expected human annotation effort for deriving a semantic segmentation of 2D imagery. Luo et al. [21] transferred the AL idea to the semantic segmentation of mobile mapping point clouds relying on a sophisticated higher order Markov Random Field. However, only few works focus on ALS point clouds, such as Hui et al. [14], who apply an AL framework for iteratively refining a digital elevation model. For semantic segmentation of ALS data, Li & Pfeifer [19] introduce an artificial oracle by propagating few available class labels to queried points based on their geometric similarity.

For exceeding the limits of automatically answering the query of the AL loop, Lin et al. [20] define an AL regime for the semantic segmentation of ALS point clouds relying on the *PointNet++* [26] architecture, where labels are given by an omniscient oracle. The inherent problem of employing CNN approaches in AL is that usually the majority of points does not carry a label and cannot contribute to the loss function. Often, this problem is circumvented by firstly performing an unsupervised segmentation for building subsets of points, which are to be completely annotated by the oracle [13,20]. Although such a procedure drastically reduces the amount of necessary labels, the oracle is still asked to deliver full annotations (of subsets) requiring a lot of human interaction. In Kölle et al. [16] this issue is directly addressed by excluding unlabeled points from the loss computation while still implicitly learning from them as geometric neighbors of labeled points. Additionally, the authors found that AL loops utilizing the state-of-the-art SCN [10] architecture can result in more computational effort due to relearning (or at least refining) features in every iteration step and might converge slower compared to conventional feature driven classifiers. Hence, a CNN design might not be optimal for AL.

In most of the aforementioned works it is assumed that labels of selected primitives are received by an omniscient oracle, which is a naive assumption, regardless of whether an expert or the crowd is labeling [33]. Consequently, for fully relieving experts from labeling efforts and to form a feasible hybrid intelligence [31] or human-in-the-loop system [2], integration of crowdsourced labeling into the AL procedure in an automated manner is required.

Our contribution can be summarized as follows: We develop a framework referred to as CATEGORISE (**C**rowd-based **A**ct**i**ve L**e**arnin**g** f**or** Po**i**nt **Se**mantics), which is tailored for 3D point annotation, but can be easily transferred to other tasks as well. This includes a detailed discussion of i) possibilities for automated quality control tested in various crowd campaigns (Sect. 3.1), ii) measures for automation of the crowd management (Sect. 3.2) and iii) a suitable intrinsic quality measure for the AL loop to enable an operator to monitor the training progress of the machine (Sect. 3.3). Please note that in contrast to related work in this domain [15,16], which mainly focuses on how to employ AL in a crowd-based scenario for semantic segmentation of point clouds, the focus of this paper lies in the automation and enables running such AL loops incorporating real crowdworkers as if the annotation is a subroutine of a program.

3 The CATEGORISE Framework

As aforementioned the backbone of our framework is to combine crowdsourcing with AL to iteratively point out only the subset of points worth labeling. Starting from an initial training data set (see Sect. 5.2), a first classifier C is trained and used to predict on the remaining training points. In our case, we apply a Random Forest (RF) classifier [3] (features are adopted from Haala et al. [11]). Predicted a posteriori probabilities $p(c|x)$ (that point x belongs to class c) are then used to determine samples the classifier is most uncertain about (i.e., the classifier would benefit from knowing the actual label). This sampling score can be derived via entropy E:

$$x_E = \underset{x}{\operatorname{argmax}} \left(- \sum_c p(c|x) \cdot \log p(c|x) \right) \tag{1}$$

In order to also consider imbalanced class occurrences, we further rely on a weighting function, which is derived based on the total number of points n_T currently present in the training data set and the number of representatives of each class n_c at iteration step i: $w_c(i) = n_T(i)/n_c(i)$. For avoiding sampling of points which are similar in terms of their representation in feature space (in context of pool-based AL) and to boost the convergence of the iteration, we adapt the recommendation of Zhdanov [35]. Precisely, we apply a k-means clustering in feature space and sample one point from each cluster (number of clusters equals number of points n_{AL} to be sampled) in each iteration step.

To especially account for the employment of real crowdworkers, we rely on a sampling add-on proposed by Kölle et al. [16], which aims on reducing the interpretation uncertainty of points situated on class borders, where the true class is hard to tell even by experts, referred to as *RIU* (*Reducing Interpretation Uncertainty*). Precisely, in each case, we use a point with highest sampling score as seed point but select an alternative point within a distance of d_{RIU} (in object space) instead. Combining these sampling strategies yields to an optimal selection of points both in context of informativeness and crowd interpretability crucial for our framework.

3.1 Automation of Quality Control

Such a fully automated framework is only applicable if the operator can trust labels received from the crowd to be used for training a supervised ML model. Since results from crowdworkers might be of heterogeneous nature [33], quality control is of high importance. Although interpretation of 3D data on 2D screens requires a distinct spatial imagination, in Kölle et al. [15] it was already shown that crowdworkers are generally capable of annotating 3D points. Within the present work, we aim to analyze in which way the performance of crowdworkers for 3D point annotation can be further improved. Quality control measures can be categorized as i) *quality control on task designing* and ii) *quality improvement after data collection* [34]. In case of labeling specific selected points, which can be

thought of as categorization task, one realization of the latter can be derived from the phenomenon of the *wisdom of the crowd* [30]. This means that aggregating answers of many yields to a result of similar quality compared to one given by a single dedicated expert. In our case *wisdom of the crowd* can be translated to simple majority vote (MV) of class labels given by a group of crowdworkers (i.e., a crowd oracle). Consequently, this raises the question of how many crowdworkers are necessary to get results sufficient to train a ML model of desired quality. Detailed discussion of experimental set up can be found in Sect. 5.1.

We would like to stress that to clarify this question it is insufficient to run a labeling campaign multiple times and vary the number n of crowdworkers employed since results would be highly prone to individual acquisitions, which might be extraordinary good or bad (especially for small n). To derive a more general result, we ran the campaign only once and each point was covered by a total of k crowdworkers ($k \geq n$). From those k acquisitions, for each n (i.e., the number of crowdworkers required; range is $[1, k]$) we derive all possible combinations:

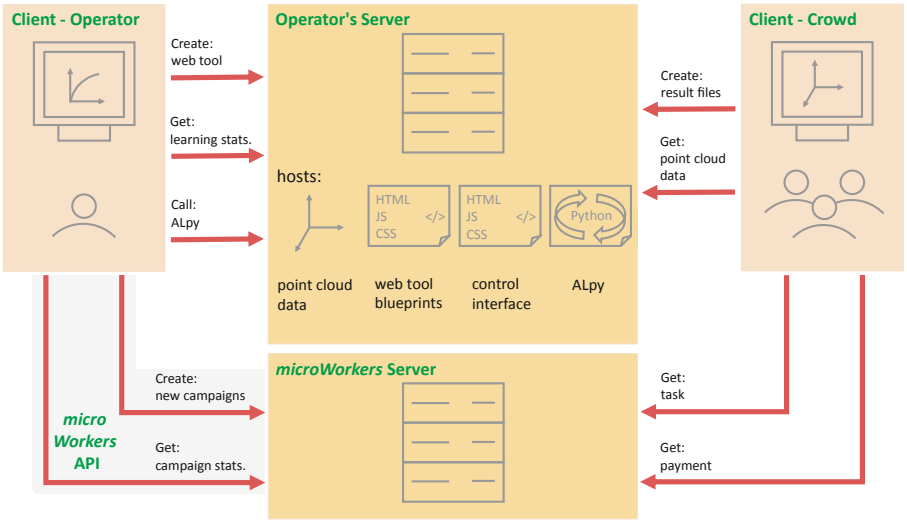
$$n_{comb} = \binom{n}{k} = \frac{n!}{(n-k)! \cdot k!} \tag{2}$$

For each n, acquisitions for each combination were aggregated via MV and evaluated according to Overall Accuracy (OA) and classwise mean F1-score. Afterwards, quality measures of each combination (for a specific n) were averaged. Consequently, our quality metrics can be considered as typical result when asking for labels of n crowdworkers.

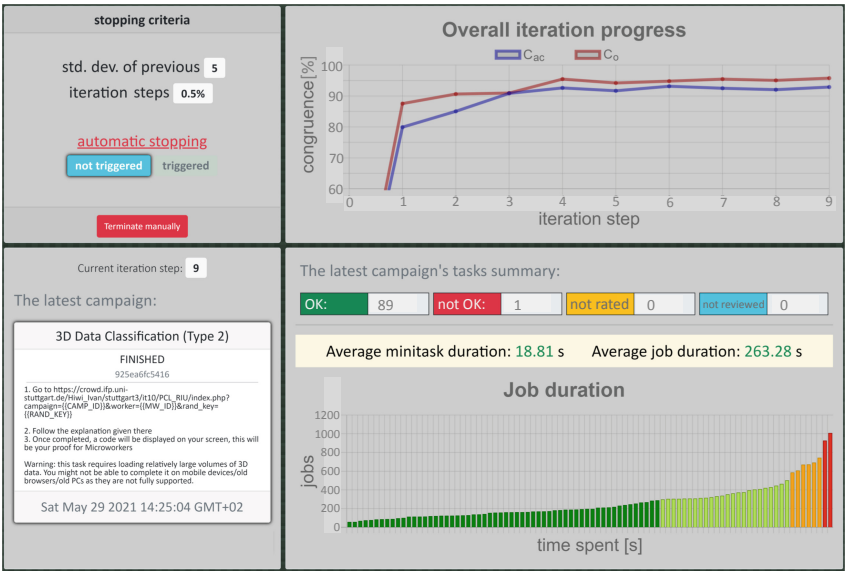
However, a drawback of accomplishing quality control by *wisdom of the crowd* is increased costs due to multiple acquisitions. Therefore, it is beneficial to also employ *quality control on task designing* [34]. In our case, this is realized by including check points in our tasks. Precisely, in addition to labeling queried AL points, each crowdworker is asked to label a specific number of check points with known class label. Those additional data points can then be used to filter and reject results of low quality. Hence, labels from: i) crowdworkers who did not understand the task, ii) crowdworkers who are not capable of dealing with this kind of data or even from iii) malicious crowdworkers (i.e., who try to maximize their income by randomly selecting labels in order to quickly finish the task) can be filtered. This poses the question of the right number of check points to be included and the consequent impact to labeling accuracy (analyzed in Sect. 5.1).

3.2 Automation of Crowd Management

To realize a truly automated process, we need to avoid any engagement between operator (i.e., employer) and crowdworkers. Within our framework (visualized in Fig. 1(a)), we draw on the crowd of *microWorkers*, which also handles the payment of crowdworkers by crediting salaries to the *microWorkers* account of the crowdworker (avoiding to transfer money to individual bank accounts, which



(a)



(b)

Fig. 1. Architecture of the CATEGORISE framework (a) and the interface used by the operator (b). The latter is designed so that the operator may monitor and control the complete AL run.

would be laborious and would cause fees). Crowd campaigns can be prompted in an automated manner by leveraging the *microWorkers* API. Simultaneously, a respective web tool is set up (by feeding parameters to custom web tool

blueprints) and input point clouds to be presented to crowdworkers are prepared (all point cloud data is hosted on the operator's server). An exemplary tool is visualized in Fig. 2(a). When a crowdworker on *microWorkers* accepts a task, he uses the prepared web tool and necessary point cloud data is transferred to him. After completion of the task, results are transmitted to the operator's server (via php) and the crowdworker receives the payment through *microWorkers*.

Meanwhile, by usage of our control interface (see Fig. 1(b)) hosted on the operator's server, the operator can both request the state of an ongoing crowd campaign from the *microWorkers*' server (e.g., number of jobs completed, respective ratings and avg. task duration (see Sect. 3.1), etc.) and the current training progress from the operator's server in order to monitor the overall progress. The control interface and web tools are implemented in Javascript (requests are handled with AJAX). As soon as all points of one iteration step are labeled, the evaluation routine is called (which is implemented in python) and the AL iteration continues (provided the stopping criterion is not met).

3.3 Automated Stopping of the AL Loop

When we recall our aim of an automated framework, it is crucial that it not only runs in an automated manner but also stops automatically when there is no significant quality gain anymore (i.e., the iteration converges). Therefore, our aim is to find an effective measure of quality upon which we can build our stopping criterion for the AL loop and which does not need to resort to GT data. Inspired by the approach of Bloodgood & Vijay-Shanker [1], we accomplish this by determining congruence of predicted labels (for the distinct test set) from the current iteration step to the previous one (i.e., we compute the relative amount of points for which the predicted class label has not changed). In addition to this overall congruence C_o, to sufficiently account for small classes, we further derive a classwise congruence value by first filtering points currently predicted as c and check whether this class was assigned to those points in the previous iteration step as well. These individual class scores can be averaged to get an overall measure C_{ac} equally sensitive for each class. For actually stopping the iteration, we assume that the standard deviation of congruence values of the previous n_{stop} iteration steps (counted from the current one) converges towards 0, which means that change in predictions stays almost constant (i.e., only classes of few most demanding points change).

4 Data Sets

All our tests were conducted on both ISPRS' current benchmark data sets, one being the well-known Vaihingen 3D (V3D) data set [24] captured in August 2008 and the other one being the recently introduced Hessigheim 3D (H3D) data set [18] acquired in March 2018. V3D depicts a suburban environment covering an area of 0.13 km^2 described by about 1.2 M points. We colorized the points by orthogonal projection of colors from an orthophoto received from Cramer

[6]. Color information is used both for deriving color based features and for presenting point cloud data to crowdworkers. H3D is an UAV laser scanning data set and consists of about 126 M points covering a village of an area of about 0.09 km^2. The class catalog of both data sets can be seen in Table 1. Please note that in order to avoid labeling mistakes of crowdworkers solely due to ambiguous class understanding, in case of V3D class *Powerline* was merged with class *Roof* and classes *Tree*, *Shrub* and *Fence* were summarized to class *Vegetation*. In case of H3D, class *Shrub* was merged with *Tree* (to *Vegetation*), *Soil/Gravel* with *Low Vegetation*, *Vertical Surface* with *Façade* and *Chimney* with *Roof*.

5 Results

Within this section, we first discuss our experiments for determining proper measures for quality control (Sect. 5.1), which constitute the basis for conducting our crowd-based AL loops presented in Sect. 5.2.

5.1 Impact of Quality Control Measures to Label Accuracy

To determine measures for quality control (see Sect. 3.1) a total of 3 crowd campaigns were conducted for H3D. These campaigns are dedicated to i) analyze labeling accuracy w.r.t the number of multiple acquisitions when quality control is done by MV only, ii) explore the impact of including check points and iii) derive the optimal number of multiple acquisitions when combining both check points and MV (i.e., realizing *quality control on task designing* and *quality improvement after data collection*). For each campaign, we randomly selected 20 points per class and organized those in jobs of 6 points each (one point per class), which results in a total of 20 jobs. Points were randomly shuffled within each job to avoid that crowdworkers realize a pattern of point organization. Each job was processed by $k = 20$ different crowdworkers, who used the web tool visualized in Fig. 2(a). In addition to the point to be labeled, we extract a 2.5 D neighborhood having a radius of 20 m (trade-off between a large enough subset for feasible interpretation and required loading time, limited by the available bandwidth) to preserve spatial context.

Figure 2(b) depicts the labeling accuracy of the crowd w.r.t. the number of acquisitions used for MV (please note that results are averaged over all n_{comb} combinations; see Eq. 2). In addition to the OA and classwise F1-scores, we further derive entropy from relative class votes to gain a measure of uncertainty for crowd labels. This first campaign shows that MV leads to almost perfect results in the long run proving the concept of the *wisdom of the crowd*. However, this requires a lot of multiple acquisitions and thus causes increased costs. F1-scores of most classes converge from about $n = 10$ acquisitions on. However, most classifiers are capable to cope with erroneous labels to some extent. In Kölle et al. [16] it was shown that about 10 % of label errors only marginally harm the performance of a classifier. Considering those findings, a significantly smaller number for n of about 5 is actually required.

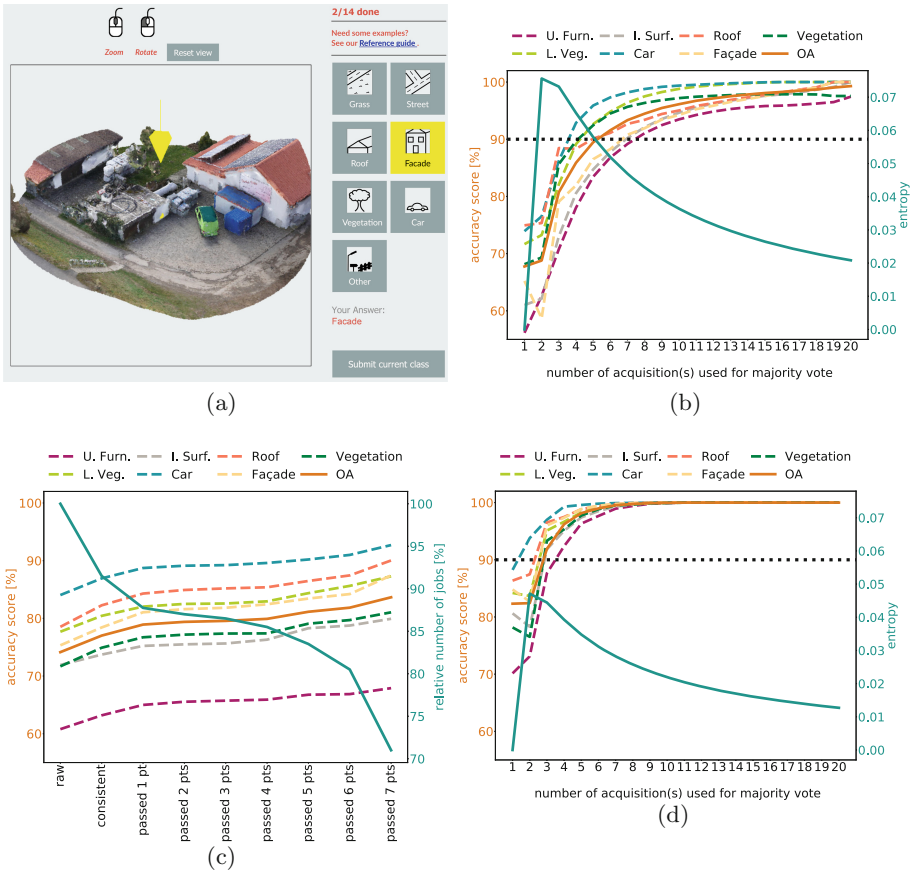


Fig. 2. Developed web tool used by crowdworkers for labeling 3D points (a) and derived results. We compare the result of pure MV (b) to the result of the same task when adding check points (d). The quality improvement by check points is displayed in (c). (web tool can be tried out at https://crowd.ifp.uni-stuttgart.de/DEMO/index.php).

Nevertheless, we aim to further save costs by introducing check points (see Sect. 3.1). We dedicated the second campaign to determining the optimal number of check points. Precisely, we added the same 7 check points to each task and showed the first and last check point twice in order to check consistency of given labels. Crowdworkers were informed about presence of check points but without giving further details or resulting consequences. In post-processing, we used these check points to gradually filter results which do not meet a certain quality level (see Fig. 2(c)). In this context, quality level *consistent* means that the check point presented twice was labeled identically but not necessarily correct. Correctness however, is assumed for level *passed 1 pt* (following quality levels additionally incorporate correctness of more than one check point). We can see that OA can be improved by about 10 percentage points (pps) when enforcing

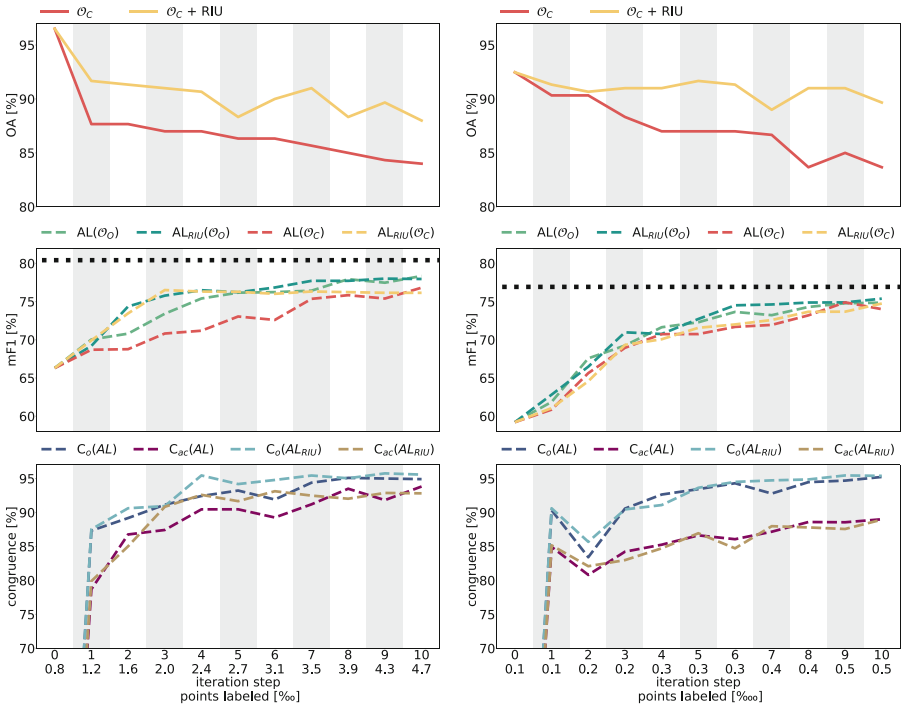
the highest quality level. On the other hand, with this quality level, about 30 % of jobs would not pass our quality control and would have to be rejected. Additionally, an extra labeling effort of 8 points per job would cause additional costs (since crowdworkers should be paid fairly proportional to accomplished work). Therefore, we decided to use quality level *passed 3 pts* for our future campaigns, which offers a good trade-off between accuracy gain and number of jobs rejected.

Considering these findings, we posted the third campaign, which differs from the previous ones as it combines both quality control strategies (MV & check points). By using a total of 3 check points (one being used twice for consistency), we aim on receiving only high-quality results as input for MV. As trade-off between error tolerance and additional incentive, we allowed false annotation of one point but offered a bonus of 0.05 $ to the base payment of 0.10 $ per job (which is also the base payment for campaign 1 and 2) when all check points are labeled correctly. Results obtained are displayed in Fig. 2(d). We observed that adding check points drastically boosts convergence of accuracy. Using check points and relying on results from 10 crowdworkers leads to an even better result than considering 20 acquisitions without check points (see Fig. 2(b) vs. (d)). This holds true for all classes with class *Urban Furniture* having the worst accuracy and class *Car* having top accuracy (overall trend is identical to the first campaign). Class *Urban Furniture* is of course difficult for interpretation since it actually serves as class *Other* [18], which makes unique class affiliation hard to determine. If we again accept a labeling OA of about 90% for training our classifier, 3 acquisitions are sufficient (2 less than for the first campaign). This offers to significantly minimize costs in case of larger crowd campaigns where many hundred points are to be annotated.

5.2 Performance of the AL Loop

Finally, we employ the CATEGORISE framework for conducting a complete AL loop for both the V3D and H3D data set. For setting up the initial training set, often random sampling is pursued. Since this might lead to severe undersampling of underrepresented classes (such as *Car*), we launch a first crowd campaign where a total of 100 crowdworkers are asked to select one point for each class. Since this kind of job cannot be checked at first (due to lack of labeled reference data), we present selected points of each crowdworker to another one for verification. Points which are tagged false are discarded. For both data sets we conduct a total of 10 iteration steps, sample $n_{AL} = 300$ points in each step and parametrize all RF models by 100 binary decision trees with maximum depth of 18.

Performance of the Crowd within the AL Loop. Figure 3 (*top row*) visualizes the accuracy of crowd labeling (obtained from MV from 3 acquisitions using quality level *passed 3 pts*) throughout the iteration process for both presenting queried points to the crowd oracle $\mathcal{O}_C$ and points which were selected by adapting the query function by *RIU*. Please note that OA is used as quality



(a) V3D (b) H3D

Fig. 3. Results of the AL loop for both the V3D and H3D data set. We compare the OA achieved by the crowd for labeling (*top*) to the mean F1-score of the classifier's performance (*middle*). *Dotted black line* represents the baseline result of PL. As intrinsic measure to describe training progress, we rely on predictive congruence (*bottom*) evaluated for our crowd-based runs (using $\mathcal{O}_C$).

metric since AL tends to sample points in an imbalanced manner w.r.t. class affiliation, so that the mean F1-score would only poorly represent actual quality. In both cases, we can observe a negative trend for labeling accuracy (V3D & H3D). This is due to our AL setting where we start with points easy to interpret (i.e., points freely chosen by the crowd), for which the crowd yields top accuracies consequently. From the first iteration step on, selection of points is handled by the classifier. In the beginning it might select points which are also easy to interpret (since such points might just be missing in the training set so far) and then it continues with points which are more and more special (and typically harder for interpretation). However, *RIU* is capable to at least alleviate this problem. Using $d_{RIU} = 1.5\,\text{m}$ leads to an OA which is about 4 pps higher in case of V3D and up to 6 pps for H3D. In case of the latter, this effect especially improves quality for later iteration steps when sampling points more complex for interpretation.

Table 1. Comparison of accuracies reached both for V3D and H3D for PL and various AL approaches using different oracle types and sampling functions.

Method	Oracle	F1-score [%]							[%]	
		U. Furn.	L. Veg.	I. Surf.	Car	Roof	Façade	Veg.	mF1	OA
V3D										
PL	–	–	82.25	91.28	65.64	94.81	62.30	86.24	80.42	88.11
AL	$\mathcal{O}_O$	–	79.35	90.70	66.00	**93.14**	**57.86**	**83.06**	**78.35**	**85.89**
	$\mathcal{O}_O$ + *RIU*	–	80.67	91.13	67.80	91.12	55.22	81.83	77.96	85.29
	$\mathcal{O}_C$	–	80.11	90.71	66.05	91.25	49.87	82.91	76.82	85.03
	$\mathcal{O}_C$ + *RIU*	–	**81.10**	**91.24**	**68.26**	88.76	45.39	82.14	76.15	84.19
H3D										
PL	–	41.14	90.68	85.26	51.63	92.96	83.77	93.05	76.92	88.16
AL	$\mathcal{O}_O$	33.93	**90.31**	82.70	56.34	88.33	79.73	**92.66**	74.86	**86.65**
	$\mathcal{O}_O$ + *RIU*	**36.97**	89.91	**83.84**	55.42	**90.05**	79.61	91.91	**75.39**	86.60
	$\mathcal{O}_C$	33.37	88.34	78.14	**57.40**	88.89	**79.83**	92.07	74.01	85.15
	$\mathcal{O}_C$ + *RIU*	34.56	89.59	81.81	56.20	89.18	79.35	92.56	74.75	86.26

Forming the training data set within AL (i.e., running a complete iteration) causes costs for the payment of the crowdworkers of 190 \$ (100 pts·0.10 \$+100 pts· 3 rep. · 0.15 \$ + 10 it. steps · (n_{AL}/10 pts per job) · 3 rep. · 0.15 \$) and is completed in about 5 days (approx. 11 h · 10 it. steps + approx. 16 h for initialization = 126 h). Compared to this, estimating total expenses of Passive Learning (PL) is hard to conduct since it is on one hand determined by external factors such as the skills and salary of the annotator, the required software, the required hardware and on the other hand by the complexity of the scene and the targeted class catalog etc.

Performance of the Machine within the AL Loop. Our RF classifier relies on these crowd-provided labels within training. We compare the results of simulated AL runs using an omniscient oracle $\mathcal{O}_O$ to the respective runs using a real crowd oracle $\mathcal{O}_C$ each with and without *RIU* and to the baseline result of PL. Figure 3 (*middle row*) & Table 1 present the accuracies achieved when predicting on the unknown and disjoint test set of each data set. For V3D, we achieve a result ($AL_{RIU}(\mathcal{O}_C)$) which only differs by about 4 pps from the result of PL both in OA and mean F1-score, while this gap is only about 2 pps for H3D. In case of V3D, relying on $AL_{RIU}(\mathcal{O}_C)$ results in a significantly higher increase of mean F1-score in early iteration steps compared to $AL(\mathcal{O}_C)$ (which is beneficial when labeling budget is limited) and performs close to the optimal corresponding AL run relying on $AL_{RIU}(\mathcal{O}_O)$. For the run without *RIU*, the performance of the RF is diminished, which is due to the lower labeling accuracy of the crowd. We would also like to underline the effectiveness of *RIU* even when labels are given by $\mathcal{O}_O$, which demonstrates that this strategy also helps to receive a more generalized training set by avoiding to sample only most uncertain points. In case of H3D, all AL runs perform similarly well with $AL_{RIU}(\mathcal{O}_C)$ marginally outperforming $AL(\mathcal{O}_C)$. For both data sets, Table 1 demonstrates that while classes with a high

in-class variance such as *Urban Furniture* and *Façade* (note that façade furniture also belongs to this class) suffer in accuracy whereas underrepresented classes such as *Car* yield better results than the PL baseline.

Terminating the AL Loop. As mentioned in Sect. 3.3, we need to provide a criterion for deciding about stopping the iteration. Congruence values derived for this purpose are displayed in Fig. 3 (*bottom row*). For the initialization, congruence is 0 due to the lack of a previous prediction. Generally, congruence curves correspond well to the test results for our AL iterations. For instance, consider accuracy values for $AL_{RIU}(\mathcal{O}_C)$ (V3D), which reach close-to-final accuracy level in the third iteration step. Therefore, from the fourth iteration step on (one step offset) intrinsic congruence values $C_o(AL_{RIU})$ & $C_{ac}(AL_{RIU})$ have also reached a stable level. We would like to stress that AL runs having a close to linear increase in accuracy ($AL(\mathcal{O}_C)$ for V3D and $AL(\mathcal{O}_C)/AL_{RIU}(\mathcal{O}_C)$ for H3D) show a similar behavior in congruence. All congruence measures flatten with increasing number of iteration steps. To avoid early stopping by a too strict stopping criterion, we set $n_{stop} = 5$ (Sect. 3.3), which indeed leads to std. dev. values close to 0 for $AL_{RIU}(\mathcal{O}_C)$ (V3D), which obviously can be stopped earlier compared to the other runs (0.4 % vs. ca. 1.4 % for the other runs at it. step 10).

In case of H3D, the decrease of congruence in the second iteration step is noteworthy indicating that the predictions have changed significantly. In other words, we assume that the newly added training points have a positive impact to the training process, which is rather unstable at this point, i.e., the iteration should not be stopped here at all. The explanation for this effect is that labels of the initialization (iteration step 0) lead to a level of prediction which is slightly improved in iteration step 1 by adding unknown but comparably similar points (w.r.t. their representation in feature space), which were just missing so far (in fact mostly façade points were selected). The second iteration step then leads to the addition of a greater variety of classes so that accuracy increased significantly, causing changed class labels of many points and by this a drop in congruence. However, one inherent limitation of our intrinsic congruence measure is that it can only detect whether a stable state of training was achieved, which does not necessarily correspond to accuracy.

6 Conclusion

We have shown that the combination of crowdsourcing and AL allows to automatically train ML models by generating training data on the fly. The peculiarity of our CATEGORISE framework is that although there are humans (i.e., crowdworkers) involved in the process, the pipeline can be controlled just like any program, so that carrying out a labeling campaign can be considered as subroutine of this program. This requires automated quality control, automation of crowd management and an effective stopping criterion all addressed within this work. Although the framework was designed for annotation and semantic segmentation of 3D point clouds, it can be easily adapted to other categorization tasks

(e.g., for imagery) by i) adapting the web tool used by crowdworkers (mainly concerning the data viewer) and ii) minor changes for the AL loop (as long as individual instances such as points or pixels are to be classified).

References

1. Bloodgood, M., Vijay-Shanker, K.: A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009), pp. 39–47. Association for Computational Linguistics, Boulder, June 2009. https://www.aclweb.org/anthology/W09-1107
2. Branson, S., et al.: Visual recognition with humans in the loop. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 438–451. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_32
3. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001). https://doi.org/10.1023/A:1010933404324
4. Budhathoki, N.R., Haythornthwaite, C.: Motivation for open collaboration: crowd and community models and the case of OpenStreetMap. Am. Behav. Sci. **57**(5), 548–575 (2012). https://doi.org/10.1177/0002764212469364
5. Buhrmester, M., Kwang, T., Gosling, S.D.: Amazon's mechanical turk: a new source of inexpensive, yet high-quality, data? Perspect. Psychol. Sci. **6**(1), 3–5 (2011). https://doi.org/10.1177/1745691610393980
6. Cramer, M.: The DGPF-test on digital airborne camera evaluation - overview and test design. Photogrammetr. - Fernerkundung - Geoinf. **2010**(2), 73–82 (2010). https://doi.org/10.1127/1432-8364/2010/0041
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Li, F.F.: ImageNet: a large-scale hierarchical image database. In: CVPR 2009, pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848
8. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3354–3361 (2012). https://doi.org/10.1109/CVPR.2012.6248074
9. Google: AI platform data labeling service [WWW Document] (2021). https://cloud.google.com/ai-platform/data-labeling/docs. Accessed 2 June 2021
10. Graham, B., Engelcke, M., v. d. Maaten, L.: 3D semantic segmentation with submanifold sparse convolutional networks. In: CVPR 2018, pp. 9224–9232 (2018)
11. Haala, N., Kölle, M., Cramer, M., Laupheimer, D., Mandlburger, G., Glira, P.: Hybrid georeferencing, enhancement and classification of ultra-high resolution uav lidar and image point clouds for monitoring applications. ISPRS Ann. Photogr. Remote Sens. Spat. Inf. Sci. **V-2-2020**, 727–734 (2020). https://doi.org/10.5194/isprs-annals-V-2-2020-727-2020
12. Hirth, M., Hoßfeld, T., Tran-Gia, P.: Anatomy of a crowdsourcing platform - using the example of microworkers.com. In: IMIS 2011, pp. 322–329. IEEE Computer Society, Washington (2011). http://dx.doi.org/10.1109/IMIS.2011.89
13. Hou, J., Graham, B., Nießner, M., Xie, S.: Exploring data-efficient 3D scene understanding with contrastive scene contexts. ArXiv abs/2012.09165 (2020). http://arxiv.org/abs/2012.09165
14. Hui, Z., et al.: An active learning method for DEM extraction from airborne LiDAR point clouds. IEEE Access **7**, 89366–89378 (2019)

15. Kölle, M., Walter, V., Schmohl, S., Soergel, U.: Hybrid acquisition of high quality training data for semantic segmentation of 3D point clouds using crowd-based active learning. ISPRS Ann. Photogr. Remote Sens. Spat. Inf. Sci. **V-2-2020**, 501–508 (2020). https://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/V-2-2020/501/2020/
16. Kölle, M., Walter, V., Schmohl, S., Soergel, U.: Remembering both the machine and the crowd when sampling points: active learning for semantic segmentation of ALS point clouds. In: Del Bimbo, A., et al. (eds.) ICPR 2021. LNCS, vol. 12667, pp. 505–520. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-68787-8_37
17. Krizhevsky, A.: Learning multiple layers of features from tiny images. Technical report TR-2009, University of Toronto, Toronto (2009)
18. Kölle, M., et al.: The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV lidar and multi-view-stereo. ISPRS Open J. Photogr. Remote Sens. **1**, 100001 (2021). https://doi.org/10.1016/j.ophoto.2021.100001
19. Li, N., Pfeifer, N.: Active learning to extend training data for large area airborne LiDAR classification. ISPRS - Int. Arch. Photogr. Remote Sens. Spat. Inf. Sci. **XLII-2/W13**, 1033–1037 (2019). https://doi.org/10.5194/isprs-archives-XLII-2-W13-1033-2019
20. Lin, Y., Vosselman, G., Cao, Y., Yang, M.Y.: Active and incremental learning for semantic ALS point cloud segmentation. ISPRS J. Photogramm. Remote. Sens. **169**, 73–92 (2020). https://doi.org/10.1016/j.isprsjprs.2020.09.003
21. Luo, H., et al.: Semantic labeling of mobile LiDAR point clouds via active learning and higher order MRF. TGRS **56**(7), 3631–3644 (2018)
22. Mackowiak, R., Lenz, P., Ghori, O., Diego, F., Lange, O., Rother, C.: CEREALS - Cost-Effective REgion-based Active Learning for Semantic Segmentation. In: BMVC 2018 (2018). http://arxiv.org/abs/1810.09726
23. Mandlburger, G., Lehner, H., Pfeifer, N.: A comparison of single photon and full waveform lidar. ISPRS Ann. Photogr. Remote Sens. Spat. Inf. Sci. **IV-2/W5**, 397–404 (2019). https://doi.org/10.5194/isprs-annals-IV-2-W5-397-2019
24. Niemeyer, J., Rottensteiner, F., Soergel, U.: Contextual classification of lidar data and building object detection in urban areas. ISPRS J. Photogramm. Remote. Sens. **87**, 152–165 (2014). https://doi.org/10.1016/j.isprsjprs.2013.11.001
25. Penatti, O.A.B., Nogueira, K., dos Santos, J.A.: Do deep features generalize from everyday objects to remote sensing and aerial scenes domains? In: 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 44–51 (2015). https://doi.org/10.1109/CVPRW.2015.7301382
26. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: deep hierarchical feature learning on point sets in a metric space. In: NIPS 2017, pp. 5105–5114. Curran Associates Inc., USA (2017). http://dl.acm.org/citation.cfm?id=3295222.3295263
27. Roscher, R., Volpi, M., Mallet, C., Drees, L., Wegner, J.D.: Semcity toulouse: a benchmark for building instance segmentation in satellite images. ISPRS Ann. Photogr. Remote Sens. Spat. Inf. Sci. **V-5-2020**, 109–116 (2020). https://doi.org/10.5194/isprs-annals-V-5-2020-109-2020
28. Roynard, X., Deschaud, J.E., Goulette, F.: Paris-Lille-3D: a large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification. Int. J. Robot. Res. **37**(6), 545–557 (2018). https://doi.org/10.1177/0278364918767506
29. Settles, B.: Active learning literature survey. Computer Sciences Technical report 1648, University of Wisconsin-Madison (2009)

30. Surowiecki, J.: The Wisdom of Crowds. Anchor (2005)
31. Vaughan, J.W.: Making better use of the crowd: how crowdsourcing can advance machine learning research. Journ. Mach. Learn. Res. **18**(193), 1–46 (2018). http://jmlr.org/papers/v18/17-234.html
32. Walter, V., Kölle, M., Yin, Y.: Evaluation and optimisation of crowd-based collection of trees from 3D point clouds. ISPRS Ann. Photogr. Remote Sens. Spat. Inf. Sci. **V-4-2020**, 49–56 (2020). https://doi.org/10.5194/isprs-annals-V-4-2020-49-2020
33. Walter, V., Soergel, U.: Implementation, results, and problems of paid crowd-based geospatial data collection. PFG **86**, 187–197 (2018)
34. Zhang, J., Wu, X., Sheng, V.S.: Learning from crowdsourced labeled data: a survey. Artif. Intell. Rev. **46**(4), 543–576 (2016). https://doi.org/10.1007/s10462-016-9491-9
35. Zhdanov, F.: Diverse mini-batch active learning. CoRR abs/1901.05954 (2019). http://arxiv.org/abs/1901.05954