ALBench: A Framework for Evaluating Active Learning in Object Detection

Zhanpeng Feng[1] **Shiliang Zhang**[2] **Rinyoichi Takezoe**[2] **Wenze Hu**[1]
Manmohan Chandraker[3] **Li-Jia Li**[4] **Vijay K. Narayanan**[5] **Xiaoyu Wang**[1]

[1]Lighthouse
[2]School of Computer Science, Peking University
[3]University of California, San Diego
[4]Stanford University
[5]ServiceNow

Abstract Active learning is an important technology for automated machine learning systems. In contrast to Neural Architecture Search (NAS) which aims at automating neural network architecture design, active learning aims at automating training data selection. It is especially critical for training a long-tailed task, in which positive samples are sparsely distributed. Active learning alleviates the expensive data annotation issue through incrementally training models powered with efficient data selection. Instead of annotating all unlabeled samples, it iteratively selects and annotates the most valuable samples. Active learning has been popular in image classification, but has not been fully explored in object detection. Most of current approaches on object detection are evaluated with different settings, making it difficult to fairly compare their performance. To facilitate the research in this field, this paper contributes an active learning benchmark framework named as ALBench for evaluating active learning in object detection. Developed on an automatic deep model training system, this ALBench framework is easy-to-use, compatible with different active learning algorithms, and ensures the same training and testing protocols. We hope this automated benchmark system help researchers to easily reproduce literature's performance and have objective comparisons with prior arts. The code will be release through Github[1].

1 Introduction and Background

An automated machine learning system involves multiple components, including automated model design, training data selection, *etc.* Currently, a large body of works focus on the model design. Related works include neural architecture search, meta-learning, and hyper-parameter optimization, *etc.* The research on the other component, *i.e.*, automated data selection has not been as intensive. It is partially attributed to the complication of performance evaluation. The resulting model output by a model design strategy can be easily evaluated using standard classification/detection/segmentation datasets, following commonly used training/testing paradigms without any ambiguity. Differently, the data automation process could involve sampling strategies with extra setups, such as the number of initial training samples, the number of samples added in each iteration, when to stop, *etc.* Unfortunately, different works propose different practices which makes direct apple-to-apple comparison difficult. It obviously does not benefit the prosperity of the community.

Active learning is one of the most important techniques for data selection automation. It (Kovashka et al., 2016; Li et al., 2007; Grauman and Belongie, 2014) has been proposed to address the challenge of expensive data annotation encountered by AI models, especially deep models (He et al., 2016). Instead of annotating every sample available and training the model all at once, active

[1]https://github.com/industryessentials/ymir

Method	Training Set	Initial Size	Added Size	Stop Size	Testset
Aghdam et al. (2019)	Caltech Pedestrian, CityPersons, BDD100K	500	500	7.5K	Caltech Pedestrian testset
Haussmann et al. (2020)	self collected	100K	200K	700K	self collected
Yoo and Kweon (2019)	VOC07, VOC12	1K	1K	10K	VOC07 testset
Choi et al. (2021)	VOC07, VOC12	1K	1K	10K	VOC07 testset
	VOC07	2K	1K	4K	VOC07 testset
	COCO14	5k	1k	7k	COCO17 VAL
Kao et al. (2018)	VOC12	500	200	3.5K	VOC12 testset
	VOC07	500	200	3.5K	VOC07 testset
	COCO14	5K	1K	9K	COCO14 val
Feng et al. (2019)	KITTI	1K	200	12K	KITTI self-divided
Brust et al. (2018)	VOC12	50	50	250	VOC12 self-divided
Yu et al. (2021)	VOC07	500	500	3.5K	VOC07 testset
	VOC12	500	500	3.5K	VOC12 testset
	COCO14	5K	1K	9K	COCO14 VAL
Roy et al. (2018)	VOC07, VOC12	1655	827	7447	VOC07 testset
Yuan et al. (2021)	VOC07, VOC12	827	413	3.31K	VOC07 testset
	COCO14	2340	2340	11.7K	COCO14 VAL

Table 1: Training and testing setups utilized in active learning works for object detection, where "Added Size" denotes the number of selected samples at each iteration. The training stops when the size of augmented training set reaches the "Stop Size". It shows that current methods employ various benchmark setups which prevents objective comparison among them.

learning trains models in multiple steps. Each step tends to select and annotate a small portion of samples, which are most valuable for subsequent performance enhancement. Combined with incremental learning (Kading et al., 2016), active learning has shown great potentials in image classification, *e.g.*, it significantly reduced the annotation cost (60%) while achieving performance on par with that of using full data (Caramalau et al., 2021). Given the fact that academic datasets used for evaluating active learning have been dedicatedly prepared with sample selection and noise filtering, the improvement from active learning could be even more substantial for open world scenarios, which commonly contain more noises and redundancies.

Research on active learning for object detection has not been as popular as that for image classification. This is partially because it involves more training setups than traditional learning strategies, making the algorithm implementation, repeat, and comparison difficult. Besides that, existing active learning algorithms for detection follow different evaluation paradigms, making their performance not directly comparable. We summarize the training and test setups in existing works in Table 1. The listed studies could use:
i) different datasets. The algorithm proposed by Yoo and Kweon (2019) is only evaluated on the PASCAL VOC (Everingham et al., 2010), and is not tested on the COCO dataset (Lin et al., 2014). Some approaches are tested on autonomous driving datasets, rather than object detection datasets.
ii) different initialization settings. Both Yoo and Kweon (2019) and Kao et al. (2018) adopt the PASCAL VOC dataset, but use different number of samples in their initial training sets.
iii) different sampling settings for each iteration. The method (Haussmann et al., 2020) selects 200k images at each training step, substantially more than those determined by others (Yoo and Kweon, 2019). Introducing more samples helps to boost the performance, but leads to higher cost.
iv) different stop criteria. (Yuan et al., 2021) and (Yu et al., 2021) end up with 11.7K and 9K training samples, respectively. Stopping with more data definitely benefits the final performance.

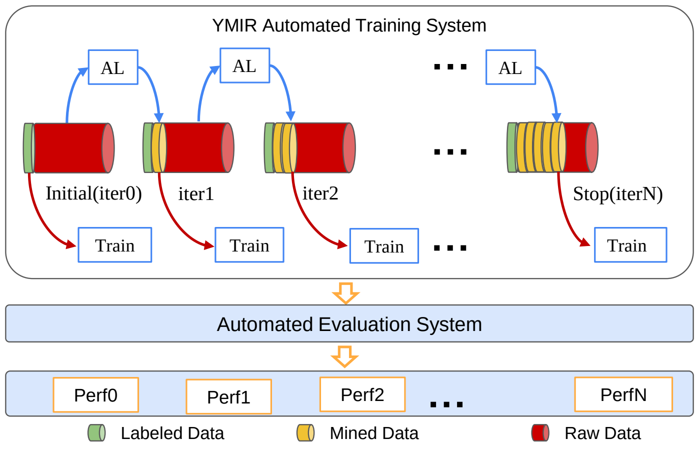

Figure 1: Illustration of active learning procedure implemented by ALBench based on YMIR (Huang et al., 2021), where "AL" denotes the active learning model provided by authors. "Train" denotes the training action using the augmented training data, each training action typically produces a model. The automated evaluation system collects all the models obtained from training actions in each iteration and produces performance profiling for the corresponding model (Perf0, Perf1, *etc.*). The comprehensive profiling results are used to compare with those of other methods, and generate the leaderboard accordingly. The whole process is automatic once the author provides the training and active learning code for a single iteration.

As the performance of active learning is closely related to both training and test settings, those observations motivated us to construct a benchmark framework for the community that is easy to reproduce and fairly compare the performance of different algorithms. To this end, we contribute a public benchmark framework named ALBench constructed based on our automated deep model training system called YMIR (Huang et al., 2021). YMIR could automatically train deep models on given training samples. Based on YMIR, ALBench implements an incremental training pipeline, which first trains initial detection models on a small annotated training set, then iteratively selects samples using active learning models. Selected samples at each step are adopted for updating deep models and active learning models with YMIR. As illustrated in Fig. 1, this procedure is iteratively repeated by ALBench until the stop criterion of active learning is met.

ALBench is a plug-and-play framework. The compatibility of various active learning algorithms with ALBench is implemented through docker images which package active learning models to fit into our training and mining framework. Within ALBench, different active learning algorithms share the same training and testing setups. We illustrate the training and testing procedures in Fig. 2, which effectively ensure an objective comparison among different active learning algorithms. Moreover, it is straightforward to study the model behavior using different settings once the related training and active learning docker images are provided. Performance on a setting can be obtained even the original work did not explore the corresponding study. We believe those characteristics of the proposed system will facilitate the research for the active learning community.

2 Benchmark Framework Design

Overview: Fig. 2 presents the framework of ALBench, which is composed of three components including the Input Interface, Training with Active Learning, and Evaluation, respectively. ALBench incrementally trains object detection models with augmented training sets obtained from the active learning process, and compares their object detection performance. ALBench is constructed based on the automated deep model training system called YMIR (Huang et al., 2021).

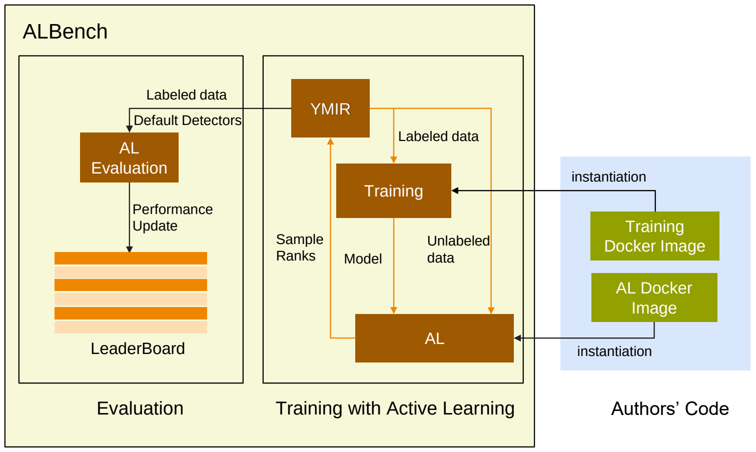

Figure 2: Illustration about how authors interact with ALBench as well as detailed process of performing the benchmark. **Authors' Code**: Implementation provided by authors. **Training with Active Learning**: Model training powered by YMIR with the provided model training code and the active learning code. **Evaluation**: Performance evaluation for the resulting model as well as all intermediate models. The brown arrows show the looping training cycle.

Open Interface: The open interface (right blue box in Fig. 2) defines how the users interact with the benchmark system. It defines protocols for packaging active learning algorithms into AL Docker Images, which are hence hosted by YMIR to run sample selection. Some active learning methodologies are highly coupled with specific detector architectures. ALBench allows the users to provide their own object detection training code through a docker image. The docker image will replace the standard training methodology in YMIR to train object detectors and update active learning models. It is fairly simple for using the benchmark system.

Training with Active Learning module takes docker images as input, and iteratively runs sample selection. The training docker image and AL docker image are hosted by the YMIR system. The training docker image is instantiated to train detection models and active learning models. The AL docker image is instantiated to perform sample selection which gives each input sample a ranking probability. The subsequent sampling procedure simply choose the top K samples. Different AL algorithms can stick to their optimal settings by packing their code and hyper-parameters in the docker image. ALBench has no dependency on these settings. Meanwhile, all AL methods share identical training setups such as initial training data, number of samples to output for each iteration, stop criteria, *etc.*, to ensure an objective performance comparison.

Evaluation: Training with Active Learning module outputs the updated training set and updated object detection models to the Evaluation module. To fairly compare the quality of their selected samples, the Evaluation module will also train default object detectors with intermediate augmented training sets in each iteration. Some active learning algorithms are coupled with specifically designed detectors and training algorithms. Evaluation module also trains them with their specific training algorithms provided by users. The performance evaluation is conducted after each active learning iteration and is reported on a leaderboard.

3 Benchmark Showcase

3.1 Default Setups

With the automated training pipeline, training and testing setups can be flexibly defined in ALBench, *e.g.*, to include new datasets and train new detectors. This paper simply tests ALBench with default setups to show its validity.

Dataset	Method	baseline	iter1	iter2	iter3	iter4
VOC07	SSD+ALDD	52.50	(+2.29) 54.79	(+5.94) 60.73	(+4.07) 64.80	(+2.13) 66.93
	SSD+CALD	52.50	(+7.47) **59.97**	(+4.19) **64.16**	(+2.78) **66.94**	(+1.56) **68.50**
	SSD+RANDOM	52.50	(+4.28) 56.78	(+5.33) 62.11	(+2.67) 64.78	(+1.70) 66.48
	SSD+ENTROPY	52.50	(+3.85) 56.35	(+5.22) 61.57	(+3.91) 65.48	(+1.09) 66.57
VOC12	SSD+ALDD	48.29	(+3.96) 52.25	(+5.20) 57.45	(+2.84) 60.29	(+2.19) 62.48
	SSD+CALD	48.29	(+7.64) **55.93**	(+2.28) **58.21**	(+3.36) **61.57**	(+0.94) **62.51**
	SSD+RANDOM	48.29	(+5.19) 53.48	(+4.53) 58.01	(+2.35) 60.36	(+1.73) 62.09
	SSD+ENTROPY	48.29	(+5.68) 53.97	(+3.50) 57.47	(+3.18) 60.65	(+1.80) 62.24

Table 2: Performance of AL methods evaluated through our system

Datasets: ALBench uses two widely used object detection datasets Pascal VOC 2007 and VOC 2012 (Everingham et al., 2010) as the default training set. For VOC2007, we use its trainval set for training and sample mining, and test on its testset. On VOC2012, we use its train set for training and sample mining, and test on the validation set. We randomly select 1000 images as the initial training set, and sample 1000 images in each iteration. The training stops after 4 iterations.

Algorithms: ALBench adopts the SSD300 (Liu et al., 2016) implemented with Vgg16 (Simonyan and Zisserman, 2015) as the default object detection model. Codes for training are acquired from the mxnet model zoo. Each active learning iteration trains for 100 epochs and picks the best model with highest mAP on the testset for sample selection. Default active learning algorithms include two commonly used baselines, *i.e.*, Entropy (Roy et al., 2018) and Random Selection (Choi et al., 2021), and two recent approaches ALDD Aghdam et al. (2019) and CALD Yu et al. (2021), respectively.

3.2 Results and Discussions

Table 2 presents the performance comparison among default active learning algorithms in ALBench. The recent CALD outperforms ALDD and baselines on VOC07 and VOC12. At iter1, CLAD substantial outperforms others. As more training samples are selected, those AL algorithms start to show similar performance. This is because the training sets of VOC07 and VOC12 only contain about 5K and 5.7K images, respectively, which minish the importance and contribution of AL algorithms as more samples are selected. Another interesting observation is that ALDD does not perform as good as simple baseline AL algorithms at iter1 and iter2. This could be because VOC training set generally contains high quality samples, making random selection achieves reasonable good performance as claimed in many studies Yu et al. (2021).

4 Limitation and Broader Impact

ALBench currently focuses on object detection task. Further work will be conducted to extend ALBench to more vision and machine learning tasks. Besides default setups, more active learning algorithms, larger and realistic training and testing sets will be included in ALBench to make more comprehensive comparisons. As the first open source benchmark platform for active learning, ALBench is expected to bring broad impact to the field of active learning.

5 Conclusions

This paper proposes an active learning benchmark framework named as ALBench to simplify the implementation and comparison of active learning algorithms in object detection. ALBench is powered by an automatic deep model training system, which applies unified training and testing setups for different active learning algorithms. ALBench framework is easy-to-use, compatible with different active learning algorithms, and ensures their objective comparison. ALBench also allows to flexibly add more training and testing setups, and detectors. As an original contribution on the benchmark of active learning, ALBench is expected to benefit the active learning community.

References

Aghdam, H. H., Gonzales-Garcia, A., Lopez, A. M., and Weijer, J. v. d. (2019). Active learning for deep detection neural networks. In *ICCV*.

Brust, C. A., Kading, C., and Denzler, J. (2018). Active learning for deep object detection. *arXiv preprint arXiv:1809.09875*.

Caramalau, R., Bhattarai, B., and Kim, T. K. (2021). Sequential graph convolutional network for active learning. In *CVPR*.

Choi, J., Elezi, I., Lee, H. J., Farabet, C., and Alvarez, J. M. (2021). Active learning for deep object detection via probabilistic modeling. In *CVPR*.

Everingham, M., L., V. G., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338.

Feng, D., Wei, X., Rosenbaum, L., Maki, A., and Dietmayer, K. (2019). Deep active learning for efficient training of a lidar 3d object detector. In *IEEE Intelligent Vehicles Symposium*.

Grauman, K. and Belongie, S. (2014). Editorial: Special issue on active and interactive methods in computer vision. *IJCV*.

Haussmann, E., Fenzi, M., Chitta, K., Ivanecky, J., Xu, H., Roy, D., Mittel, A., Koumchatzky, N., Farabet, C., and M. Alvarez, J. (2020). Scalable active learning for object detection. *arXiv preprint arXiv:2004.04699*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*.

Huang, P. X., Hu, W., Brendel, W., Manmohan, C., Li, L.-J., and Wang, X. (2021). Ymir: A rapid data-centric development platform for vision applications. In *Proceedings of the Data-Centric AI Workshop at NeurIPS*.

Kading, C., Rodner, E., Freytag, A., and Denzler, J. (2016). Fine-tuning deep neural networks in continuous learning scenarios. In *ACCV-WS*.

Kao, C. C., Lee, T. Y., Sen, P., and Liu, M. Y. (2018). Localization-aware active learning for object detection. In *ACCV*.

Kovashka, A., Russakovsky, O., Fei-Fei, L., and Grauman, K. (2016). Crowdsourcing in computer vision. *Foundations and Trends in Computer Graphics and Vision*.

Li, L., Wang, G., and Li, F. (2007). Optimol: automatic online picture collection via incremental model learning. *CVPR*.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *ECCV*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *ECCV*.

Roy, S., Unmesh, A., and Namboodiri, V. P. (2018). Deep active learning for object detection. In *BMVC*.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *ICLR*.

Yoo, D. and Kweon, I. S. (2019). Learning loss for active learning. In *CVPR*.

Yu, W., Zhu, S., Yang, T., Chen, C., and Liu, M. (2021). Consistency-based active learning for object detection. *arXiv preprint arXiv:2103.10374*.

Yuan, T., Wan, F., Fu, M., Liu, J., Xu, S., Ji, X., and Ye, Q. (2021). Multiple instance active learning for object detection. In *CVPR*.