Towards Dynamic and Scalable Active Learning with Neural Architecture Adaption for Object Detection

Fuhui Tang[†1]
419830756@qq.com
Dafeng Wei[†2]
weidafeng@sjtu.edu.cn
Chenhan Jiang[1]
jchcyan@gmail.com
Hang Xu[1]
chromexbjxh@gmail.com
Andi Zhang[3]
az381@cam.ac.uk
Wei Zhang[1]
wz.zhang@huawei.com
Hongtao Lu[2]
htlu@sjtu.edu.cn
Chunjing Xu[1]
xuchunjing@huawei.com

[1] Huawei Noah's Ark Lab
Shanghai, China
[2] Shanghai Jiao Tong University
Shanghai, China
[3] University of Cambridge
Cambridge, UK

Abstract

Active learning aims to reduce the annotation cost by selecting those informative samples to improve training efficiency and network accuracy. However, current active learning methods for object detection have three drawbacks: a) the network architectures of the detector during active learning are fixed without considering its saturation; (b) the detector may fall short in giving credible prediction probabilities on unlabeled data; (c) existing uncertainty measures may lead to homogenization of the samples. To overcome these problems, we propose a novel active learning strategy with dynamic neural architecture adaption for object detection. Specially, we incorporate a neural architecture adaption module that modifies and expands the current detector structure for the variation of incoming data stream. We design several network morphism modifications to enable an efficient adaption, which avoids the retraining of the detector after changing the network architecture in each round. Furthermore, we introduce Dirichlet calibration to correct the classifier for obtaining credible prediction, and present a clustering sampling scheme to select diverse samples. Experimental results show that the proposed method outperforms the previous state-of-the-art active learning methods with fixed architectures, improving 1.9% mAP on BDD and 1.6% mAP on COCO.

† Both authors contribute equally to this work.

1 Introduction

Active learning [8, 12, 13] is a machine learning algorithm that assists the learning procedure. It aims to improve data efficiency by judicious selection of samples for human labeling. Especially for the complex detection annotation [14, 27], which needs to give category information and mark out bounding box, active learning can greatly reduce the labeling costs. However, there is only few research related to active learning on detection.

Current active learning employs the previously-trained model to predict the uncertainty of the unlabeled data. It selects the most difficult and informative samples that the model has not well learned to maximize improvement performance while minimizing the labeling budget. Despite the progress of active learning [16, 19, 20], there are still some drawbacks: (i) current methods only equip with fixed detection architecture [26] without considering the constantly changing of labeled data. With the iteration of active learning, more data will be added to the labeled pool. It is unlikely that the fixed model can adapt to the incoming data stream. Because if a large model is adopted, it is prone to overfit on the small dataset in the early period of active learning, while choosing a small model with limited capacity is unable to accommodate a large amount of data in the later period. (ii) the distribution over unlabeled data may shift and eventually be very different from original training data distribution, causing the detector to give inaccurate prediction probabilities. (iii) the existing active learning methods only take uncertainty measurement as the basis of sample selection. Still, when many images with high uncertainty are from similar scenes (e.g., dark night), this selection method tends to cause sample homogenization, which is not conducive to model training.

To overcome these problems, we propose dynamic and scalable active learning with neural architecture adaption for object detection. Our approach, termed as AL-NAS, deliberately performs neural architecture adaption module at every active learning round, aiming to find a suitable depth, resolution, and receptive fields for each stage of feature hierarchy. Inspired by recent advances in neural architecture searching [7], we adopt the 'swap-expand' strategy in the neural architecture adaption phase. The 'swap' operation allows exchanging the adjacent convolutional layers for finding a better downsampling location. The 'expand' operation performs small increments of convolutional layers to enhance the backbone's capacity. The best modification among all nodes is selected as the initial model for the next round of active learning. For efficient adaption, we also design several network morphism modifications [3] to avoid re-pretraining the modified model at each round, where the weights of models can be well-inherited before and after modification. This 'swap-expand' strategy works well and keeps increasing the performance of active learning efficiently.

To calibrate the predictive probabilities from the classifier, we introduce Dirichlet calibration which considers the distribution of prediction vectors separately on the instances of each class. Based on the equivalence between generative parametrisation and linear parametrisation [11], Dirichlet calibration just learns a single corrective layer on log-transformed class probabilities, followed by softmax as multinomial logistic regression. It increases the confidence-reliability of predictive probabilities, which is benefit for estimating the uncertainty on unlabeled data.

For sample selection diversity, we present an effective clustering scheme to store samples according to their distribution. The features for clustering can be restored during the inference for unlabeled data, with no additional calculation. Meanwhile, to save the storage space, we perform dimension reduction on these features, it also facilitates the subsequent clustering to be more efficient. As such, the uncertainty samples can be selected proportion-

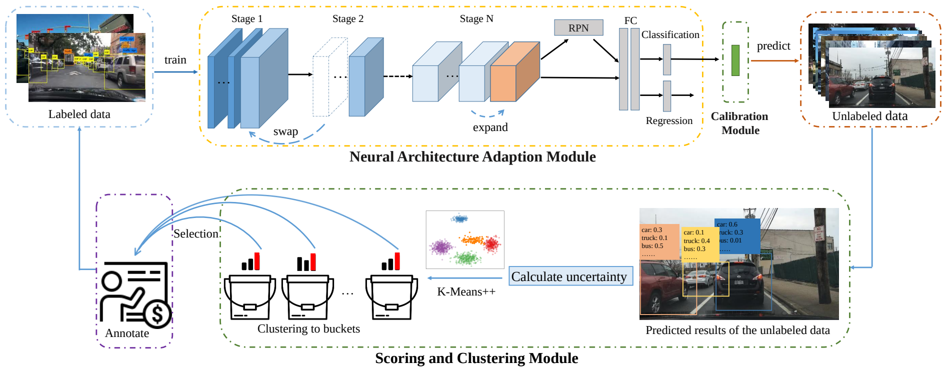
Labeled data
train
Stage 1
Stage 2
Stage N
RPN
FC
Classification
Regression
swap
expand
Neural Architecture Adaption Module
Calibration Module
predict
Unlabeled data
Selection
Annotate
Clustering to buckets
K-Means++
Calculate uncertainty
car: 0.3 truck: 0.1 bus: 0.5
car: 0.1 truck: 0.4 bus: 0.3
car: 0.6 truck: 0.3 bus: 0.01
Predicted results of the unlabeled data
Scoring and Clustering Module

Figure 1: The whole pipeline of our scalable active learning method with neural architecture adaption. At each round of active learning, we first perform the neural architecture adaptation module with 'swap-expand' strategy. Then the best model is proceeded through the Dirichlet calibration module before predicting the uncertainty on unlabeled data. After that, we design a clustering technique to distribute predicted samples into different buckets, and select the samples with the most uncertainty proportionally to update the labeled data.

ally from different clusters, keeping the diversity.

Evaluative results on two widely used detection benchmarks, i.e., COCO [14] and BDD [27], show that the proposed method achieves superior performance, significantly improves known active learning algorithms. In particular, our method achieves an accuracy of 32.8% with Resnet-18 as initial backbone architecture on the BDD dataset, even outperforms the fixed Resnet-50 (32.2%) by 0.6%, and our final architecture has absolute advantages with smaller memory, parameters and faster FPS. It is worth mentioning that the proposed method is a flexible and general framework and can be integrated with other active learning strategies to improve the performance consistently.

The main contributions of our paper can be summarized as below:

- We propose a scalable active learning approach with dynamic neural architecture adaption for object detection, generating a more powerful model in each round.
- We present Dirichlet calibration to correct the prediction probabilities, providing the high-quality uncertainty estimates on unlabeled data.
- We introduce an effective clustering scheme for selecting samples, making the chosen examples informative without losing diversity.

2 The Proposed Method

We propose scalable active learning with dynamic neural architecture adaption framework for object detection. The whole pipeline is shown in Figure 1. The core idea of our method is to integrate a neural architecture adaption module in every active learning round, aiming to adapt to the variation of the incoming data stream. Based on this purpose, we introduce

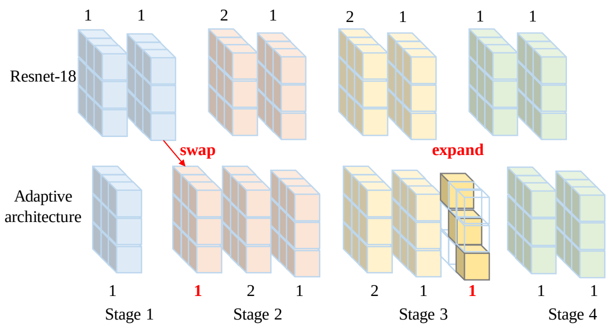

Figure 2: Illustration of the 'swap' and 'expand' operations on Resnet-18 backbone, without repeatedly pretraining from scratch.

a 'swap-expand' searching strategy to find a suitable depth and resolution in each stage. Meanwhile, we design a calibration layer to correct probabilities and present a clustering sampling scheme to collect diverse samples.

2.1 Dynamic Neural Architecture Adaption

The backbone architecture [6, 10, 22, 24] served as a feature extractor, is critical for object detection networks and active learning. Previous work in active learning has typically adopted a fixed architecture throughout the iterative process, which leads to sub-optimal performance in two main folds. Firstly, if a large backbone is selected, it is prone to overfit in the early period which lacks enough data, and fails in selecting the most informative unlabeled samples. Secondly, if a small backbone is selected, it may not have sufficient capacity to fully exploit a large amount of labeled data as iteration selection proceeds.

Therefore, we propose a neural architecture adaption scheme for the active learning process, which takes modifications and extensions into account, aims at extending the capacity of the network architecture. We also introduce the network morphism to enable an efficient adaption that avoids repeated training after changing the network architecture in each round.

Search Space: Our neural architecture adaption is designed to find a best updated model by the most significant performance improvement for the input data stream. We use the classical Resnet as the basic structure and adjust the depth, receptive fields, output channels of each stage while maintaining the number of stages. Each stage contains several basic blocks (for the Resnet-18) or bottleneck blocks(for the Resnet-50), and the resolution of the feature map remains unchanged through each block. We downsample the feature map for each stage except for the first one. Besides, to limit the complexity of the entire network, we allow a different number of blocks in each stage and adjust the order of downsampling in each stage.

For simplicity, we encode our adaptive architecture like 11-21-121-21, where '-' separates each stage with a different resolution, '1' indicates a regular block like the basic block of Resnet-18, and '2' means the number of base channel is doubled. According to this definition pattern, we encode Resnet-18 as 11-21-21-21, for an example.

'Swap-Expand' Strategy: We present a set of operations including 'swap' and 'expand' to the current model. The brilliance is that the pre-trained model can be reused by network morphism [3], avoiding the repeated training during the adaptive progress. As illustrated in Figure 2, the 'swap' operation interchanges the stride of the neighboring block while keeping the weight unchanged. For the 'expand' operation, a new block is inserted into any neighboring blocks, and the weight parameters are initialized as an identity matrix.

In summary, our neural architecture adaption method starts with a small base model (e.g., Resnet-18 pre-trained on ImageNet dataset), generates several new models with the proposed 'swap-expand' operations through the training of several epochs (i.e. 8). Then the best performing model is selected as the base model for the next iteration of active learning (i.e. pre-arch model). The cycle continues until the end condition for active learning reaches.

2.2 Dirichlet Calibration for Uncertainty

Uncertainty is a simple yet effective active learning method, it measures distances between samples and the decision boundary based on the predicted probabilities. However, deep neural networks usually yield poor probability estimates for unlabeled data whose distribution may shift from that of original data. To solve this problem, we present a Dirichlet calibration method to offer high-quality predictive probabilities for uncertainty estimates.

Uncertainty: For detection task, the uncertainty methods of active learning need to consider the uncertainty for each bounding box in an image. Given a predicted bounding box B, the uncertainty [12] is defined as:

$$U_1(B) = 1 - \hat{p}_{max}(B) \tag{1}$$

where $\hat{p}_{max}(B)$ is the highest class probability for this box. The higher the value, the smaller the probabilities of other categories, which means a relatively accurate predicted result with low uncertainty.

Another way to use the probability uncertainty is to query the margin between the posteriors [8]:

$$U_2(B) = -(\hat{p}_{max1}(B) - \hat{p}_{max2}(B)) \tag{2}$$

where $\hat{p}_{max1}$ and $\hat{p}_{max2}$ represent the first and second maximal probabilities, respectively. The minus sign in front is simply to ensure that $U(B)$ acts a maximizer.

The entropy strategy [16] is an upgrade of the above two methods. It encodes the discrete distribution of all probabilities of a predicted box and measures the variable:

$$U_3(B) = -\sum_{j=1}^{c} \hat{p}_i log(\hat{p}_j) \tag{3}$$

where c is the number of classes, a higher value of the entropy indicates greater uncertainty in the probability. The final uncertainty of the image I can be obtained by the average or maximal the uncertainty of all detected boxes within.

Dirichlet Calibration: In order to better evaluate the uncertainty on unlabeled data, we implement Dirichlet calibration for the probability $\hat{p}$. Define the canonical calibration map [25] as:

$$\mu(q) = (P(Y = 1 \mid \hat{p}(B) = q), ..., P(Y = k \mid \hat{p}(B) = q)) \tag{4}$$

where $\hat{p}(B)$ is the predicted probability of all classes for the bounding box B. These k distributions are Dirichlet distributions with different parameters:

$$\begin{gathered} Y \sim \text{Categorical}(\pi) \\ \hat{p}(B) \mid Y = j \sim \text{Dir}(\alpha^{(j)}) \text{ for } j \in \{1, ..., k\} \end{gathered} \tag{5}$$

where $\pi \in \Delta_k$ is the parameter of the categorical prior distribution and $\alpha^{(j)} = (\alpha_1^{(j)}, \dots, \alpha_k^{(j)}) \in (0, \infty)^k$ are the Dirichlet parameters for class j. Then, we use Bayes' rule to express the calibration function as:

$$\begin{aligned} P(Y = j \mid \hat{p}(B) = q) &\propto P(\hat{p}(B) = q \mid Y = j) P(Y = j) \\ &= f(q; \alpha^{(j)}) \pi_j \end{aligned} \tag{6}$$

where f is the probability density function of Dirichlet distribution. Let $z = \sum_{j=1}^{k} \pi_j f(q; \alpha^{(j)})$ be the normaliser, we can obtain:

$$\hat{\mu}_{\mathrm{DirGen}}(q; \alpha, \pi) = (f(q; \alpha^{(1)}) \pi_1, \dots, f(q; \alpha^{(k)}) \pi_k)/z \tag{7}$$

as [11] shows that $\hat{\mu}_{\mathrm{DirGen}}(q; \alpha, \pi)$ is equivalent to:

$$\hat{\mu}_{\mathrm{DirLin}}(q; W, b) = \sigma(W \ln q + b) \tag{8}$$

where $W \in \mathbb{R}^{k \times k}$ is a $k \times k$ parameter matrix, ln is a vector function that calculates the natural logarithm component-wise and $b \in \mathbb{R}^k$ is a parameter vector of length k. In practice, we only have to learn W and b by the loss function:

$$L = \frac{1}{n} \sum_{i=1}^{n} \ln[\hat{\mu}_{\mathrm{DirLin}}(\hat{p}(x_i); W, b)]_{y_i} + \lambda ||W||_2^2 \tag{9}$$

where $(x_i, y_i)_{i \in \{1,\dots,n\}}$ are validation datapoints, and the bias b is non-regularized in order to reduce variance [5].

In training stage, we learn a Dirichlet calibration layer by Eq.(9) after the classification layer, as shown in Figure 1. For training these calibration parameters, we randomly adopt 1K data from the selected unlabeled samples of the previous round, and hold the other network parameters fixed. In prediction stage, the calibration layer calibrates the predicted probabilities for each bounding box, and then the calibrated probabilities are used to estimate uncertainty by the uncertainty methods.

2.3 Clustering to Buckets

Diversity is an effective way to avoid sampling homogenization, but it is neglected in recent state-of-the-art active learning methods [1, 4, 9, 26]. Especially when the samples come from continuous scenarios, the over-sampling of similar consecutive frames not only wastes extensive manual annotation resources but also easily causes the overfitting of the model.

To overcome this challenge, we introduce a clustering method to guarantee the diversity of sampling. More specifically, we perform dimension reduction on the features of unlabeled data and then divide all unlabeled data into different buckets based on their clustering distribution. The samples can be selected proportionally within each bucket. Detailed steps of our proposed approach are as follows.

Clustering Feature: As illustrated in Figure 1, during the inference process of unlabeled data, we preserve the intermediate features which are the output of the last convolutional layers of the last stage, these features are most informative for clustering. As this step needs no deliberate feature extraction, the process is very effective.

Dimension Reduction: Before saving these features directly, it is essential to downscale them. Because their volume would be around 24 times greater than the original input image, which increases a gross IO and storage overhead. Such large features also present challenges for understanding and exploring the relationships. An intuitive solution is downsampling with a 1*1 convolutional layer or simply adding max or mean pooling layer. However, there are no direct and reasonable labels to train that convolutional layer and learn reasonable parameters, while the pooling layer with a very large stride could lose a majority of significant information. Instead, the classical machine learning methods such as Principal Component Analysis (PCA) [21] or t-Distributed Stochastic Neighbor Embedding (t-SNE) [17] can be adopted to perform on these features.
Clustering to Buckets: After obtaining the dimension-reduced features of all unlabeled data, we aggregate them to K clusters with K-Means++ [23]. As shown in Eq.(10), we first randomly select K objects as the initial clustering centers C, and then calculate the distance E between each object and each center:

$$E = \sum_{i=1}^{K} \sum_{x \in C_i} \|x - u_i\|^2 \tag{10}$$

where $\mu_\mathrm{i} = \frac{1}{|c_i|} \sum_{x \in C_i} x$ is the mean vector of C_i.

This method takes advantage of the clustering results that spread different samples into different buckets, maintaining the diversity of sample selection.

3 Experiments

In this section, we perform extensive experiments on two widely used detection benchmarks: COCO [14] and BDD [27]. We first provide an ablation study for a better understanding of our method and then compare it with other state-of-the-art active learning algorithms.

3.1 Datasets and Evaluation Protocol

We use the COCO dataset [14] to evaluate universal object detection and Berkeley Deep Drive (BDD) dataset [27] to evaluate domain-specific object detection. COCO dataset contains 80 classes with 118K images for training, 5K images for validation. BDD is a large-scale autonomous driving dataset. It contains 10 classes with 70K images sampled from 100K video clips. Since the training images of the 'train' category are too few, we ignore this category and only report results of the other 9 categories. For COCO and BDD datasets, we follow the standard evaluation metrics from COCO style, and calculate mean Average Precision (mAP) across IoU thresholds from 0.5 to 0.95 with an interval of 0.05.

3.2 Implementation Details

We employ representative Faster RCNN [18] with FPN [15] to implement our method, with mmdetection toolbox [2]. The Resnet-18 pre-trained on ImageNet [10] are used for backbone architectures of the detectors. We adopt 8 NVIDIA Tesla V100 GPUS cards only for parallel acceleration, with a batch size of 16 and 2 images per GPU. During each round of active learning training, we optimize the objective function by the Stochastic Gradient Descend(SGD) method for 8 epochs. The learning rate is set to 0.02 and it is reduced with a

Methods					Data				
Random	ENT[16]	Clu	Cal	NAS	14K	28K	42K	56K	70K
√					23.8	27.0	28.8	29.9	30.8
	√				23.8	27.7	29.7	30.3	30.9
	√	√			23.8	27.9	29.8	30.6	31.0
	√	√	√		23.8	28.0	30.1	30.9	31.2
	√	√	√	√	**23.8**	**28.6**	**30.8**	**31.8**	**32.8**

Table 1: Ablation study of different components for active learning on BDD dataset (mAP). ‘Clu’ denotes Clustering, ‘Cal’ denotes Calibration, ‘NAS’ is the proposed neural architecture adaption. Resnet-18 is employed as backbone.

cosine learning rate schedule until 0.0001 in the last iteration. In all experiments, we randomly select 20K images from COCO training set as initial labeled data, and the remaining 100K images as unlabeled data. At every round of active learning, another 20K images are selected from unlabeled data for labeling and then sent to the labeled pool. Similarly, the BDD training set is divided into 14K and 56K as labeled data and unlabeled data, respectively, and 14K images are selected from unlabeled data for labeling in each round. The resolutions for COCO and BDD are set as (1333, 800) and (1280, 720), respectively, following the traditional implementation of each dataset. The clustering number K in Eq.(10) is set to 6 for BDD, 60 for COCO. The NAS strategy searches for 8 models at each round of active learning.

3.3 Ablation Study

We evaluate each module of our active learning method to demonstrate its effectiveness. As shown in Table 1, the performance of the random method continues to improve as the labeled data increases, while ENT[16] has a more obvious advantage, especially at the rounds of 28K and 42K. This implies that using a proper small model on a small dataset can better reflect the effect of active learning. Our ‘Clu’ and ‘Cal’ schemes further improve the performance in every round, demonstrating that sample diversity and probability calibration are conducive to learning a generalized and robust model. There are two points worth noting: (1) Although all approaches adopt the same complete 70K data in the final round, active learning can still improve the model. This can be interpreted as a better pre-trained model from the previous round that can bring better initial weights for the training of the current round. (2) The growth trend brought by active learning seems to decrease with the increase of data. It is mainly due to that the fixed model architecture is unable to adapt to the incoming data flow, so it is particularly necessary to equip incremental data with a neural architecture adaption module, and our approach emerges. The proposed NAS module consistently and significantly improves the performance by a large margin, from 0.6% to 1.6%, compared with ‘ENT + Clu + Cal’. It can find a more powerful model architecture in each round of active learning, including depth, resolution, and receptive fields.

3.4 Comparison with State-of-the-Art

Comparison on BDD Dataset: Figure 3 shows the performance curves of our method and other state-of-the-art methods on BDD dataset. The state-of-the-art methods have approximate performance that are superior to those of random selection, especially in the second and third rounds. However, the advantage is less obvious in the final round. Compared

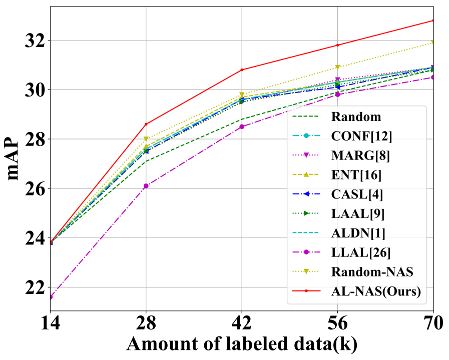

Figure 3: The detection performance (mAP) of our method and other state-of-the-art methods on BDD dataset.

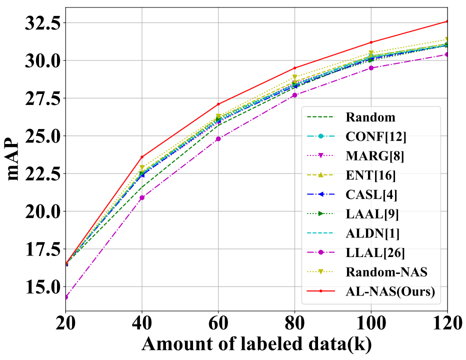

Figure 4: The detection performance (mAP) our method and other state-of-the-art methods on COCO dataset.

Data	Ours	Pre_Arch	Arch	Memory (MB)	Params (M)	FPS	mAP
14K		Resnet-18	Fixed	2176	28.68	20.5	23.8
		Resnet-50	Fixed	4338	43.98	13.4	26.4
28K	√	Resnet-18	11-211-21-21	2353	29.05	20	28.6
		Resnet-50	Fixed	4338	43.98	13.4	30.1
42K	√	11-211-21-21	112-112-12-1	2700	29.43	17.4	30.8
		Resnet-50	Fixed	4338	43.98	13.4	31.4
56K	√	112-112-12-1	11211-1121-12-1	3309	31.21	14.7	31.8
		Resnet-50	Fixed	4338	43.98	13.4	31.9
70K	√	11211-1121-12-1	11211-1121-121-1	**3440**$^{-898}$	**35.93**$^{-8.05}$	**14.0**$^{+0.6}$	**32.8**$^{+0.6}$
		Resnet-50	Fixed	4338	43.98	13.4	32.2

Table 2: The detailed neural architectures at each round on BDD dataset. 'Pre_arch' denotes that the backbone architecture of previous round is served as the initial architecture in the current round. 'Arch' denotes the new backbone architecture obtained by our method.

with these active learning approaches with fixed architectures, the proposed scalable active learning with neural architecture adaption method (AL-NAS) consistently produces superior results in each round. It is most striking that our method still drives the improvement by a large margin in the final round (i.e., 32.8%), breaking the bottleneck that active learning methods have little effect on the full data. What's more, we note that Random-NAS also performs better than other state-of-the-art methods, demonstrating that our NAS module is very versatile to be integrated into other approaches.

Comparison on COCO Dataset: To further illustrate the effectiveness of our method, we also provide the results on COCO dataset in Figure 4. Without any bells and whistles, our AL-NAS achieves the best results. Equipped with neural architecture adaption module, Random-NAS is able to adjust the model as the amount of training data changes, so it ranks second. LLAL[26] jointly trains the target and prediction modules, the gradient backpropagation affects the original network, resulting in a lower performance in the initial round. Other methods are superior to random selection. However, with continuous iterations, the models of these methods tend to be saturated and cannot further improve random selection.

Comparison with Larger Model: In addition, it is interesting to inspect the final model that has been selected by the neural architecture adaption module. As shown in Table 2, we

Methods	Data				
	14K	28K	42K	56K	70K
AL-NAS ($K = 4$)	23.8	28.5	30.6	31.7	32.6
AL-NAS ($K = 6$)	23.8	28.6	30.8	31.8	32.8
AL-NAS ($K = 10$)	23.8	28.6	30.7	31.9	32.7

Table 3: Parameter analysis of clustering number on BDD dataset (mAP) to explore the relationship between sample diversity and uncertainty.

embed ENT[16] into a fixed model with Resnet-50 as the backbone and our adaptive model with Resnet-18 as the starting backbone, respectively. In the initial active learning round, it is evident that there is a big mAP gap (2.6%) between Resnet-18 and Resnet-50. Later on, the gap is getting smaller deriving from the continuous improvement of our approach to Resnet-18. In the fourth round, the performance of our architecture (31.8%) is comparable to that (31.9%) of Resnet-50. It is worth noting that our final architecture (32.8%) significantly outperforms Resnet-50 (32.2%) in the fifth round, while our method only adds a few layers to original Resnet-18, eventually increasing to 28 layers, far less than Resnet-50. Meanwhile, our model has a huge advantage over Resnet-50 in terms of Memory, Params, and FPS.

3.5 Parameter Analysis

We adjust the clustering number K on our method to explore the relationship between sample diversity and uncertainty. When K is 1, it is equivalent to no clustering. As K increases, the sample distribution becomes tighter and the aggregation degree of each cluster gradually grows. However, a very large K might be not necessarily a better option for active learning. As shown in Table 3. a larger K can only guarantee the diversity of the samples, while may lose the uncertainty. For example, various samples with low uncertainty are clustered into the same bucket, but these low-informative samples will be selected proportionally at every active learning round, which reduces the constraint of selecting samples based on uncertainty. In conclusion, the cluster number K moderates the trade-off between uncertainty and diversity.

4 Conclusion

In this paper, we propose a novel active learning pipeline with neural architecture adaption for object detection. The adaption module consistently modifies and expands the current architecture to offer a more powerful detector for the real-time data instream. The morphism design allows the detector to inherit previous weights well during each round, making the training more efficient. The Dirichlet calibration provides more accurate prediction probabilities, and the clustering scheme shows effective performance improvement from sample diversity. Extensive experimental results on multiple detection benchmarks show that the proposed method achieves state-of-the-art performance. It is worth mentioning that our framework is highly versatile and can be integrated with other active learning methods to advance them.

Acknowledgements: This work is partially supported by NSFC (No. 62176155, 61772330) and Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102).

References

[1] Hamed H Aghdam, Abel Gonzalez-Garcia, Joost van de Weijer, and Antonio M López. Active learning for deep detection neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3672–3680, 2019.

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[3] Thomas Elsken, Jan-Hendrik Metzen, and Frank Hutter. Simple and efficient architecture search for convolutional neural networks. *arXiv preprint arXiv:1711.04528*, 2017.

[4] Mingfei Gao, Zizhao Zhang, Guo Yu, Sercan Ö Arık, Larry S Davis, and Tomas Pfister. Consistency-based semi-supervised active learning: Towards minimizing labeling cost. In *European Conference on Computer Vision*, pages 510–526. Springer, 2020.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] Chenhan Jiang, Hang Xu, Wei Zhang, Xiaodan Liang, and Zhenguo Li. Sp-nas: Serial-to-parallel backbone search for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11863–11872, 2020.

[8] Ajay J Joshi, Fatih Porikli, and Nikolaos Papanikolopoulos. Multi-class active learning for image classification. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2372–2379. IEEE, 2009.

[9] Chieh-Chi Kao, Teng-Yok Lee, Pradeep Sen, and Ming-Yu Liu. Localization-aware active learning for object detection. In *Asian Conference on Computer Vision*, pages 506–522. Springer, 2018.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[11] Meelis Kull, Miquel Perello-Nieto, Markus Kängsepp, Hao Song, Peter Flach, et al. Beyond temperature scaling: Obtaining well-calibrated multiclass probabilities with dirichlet calibration. *arXiv preprint arXiv:1910.12656*, 2019.

[12] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*, pages 148–156. Elsevier, 1994.

[13] David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *SIGIR'94*, pages 3–12. Springer, 1994.

[14] 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 *European conference on computer vision*, pages 740–755. Springer, 2014.

[15] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[16] Wenjie Luo, Alex Schwing, and Raquel Urtasun. Latent structured active learning. In *Advances in Neural Information Processing Systems*, pages 728–736, 2013.

[17] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[18] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[19] Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In *European Conference on Machine Learning*, pages 413–424. Springer, 2006.

[20] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079, 2008.

[21] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[23] MA Syakur, BK Khotimah, EMS Rochman, and BD Satoto. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. In *IOP Conference Series: Materials Science and Engineering*, volume 336, page 012017. IOP Publishing, 2018.

[24] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[25] Juozas Vaicenavicius, David Widmann, Carl Andersson, Fredrik Lindsten, Jacob Roll, and Thomas Schön. Evaluating model calibration in classification. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3459–3467. PMLR, 2019.

[26] Donggeun Yoo and In So Kweon. Learning loss for active learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 93–102, 2019.

[27] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018.