RETHINKING DYNAMIC SCALE TRAINING WITH REGULARIZATION EFFECT OF DATA AUGMENTATION AND DATA POOL

Yuxuan Cheng

Guangdong Provincial/Zhuhai Key Laboratory of Interdisciplinary Research and Application for Data Science, Beijing Normal-Hong Kong Baptist University, Zhuhai, China

ABSTRACT

Natural data collected from the real world often exhibit the scale imbalance problem. A large object can produce much more loss values than a small object, causing the detector to favour large objects more, even though small objects dominant the dataset. This inclination inside detectors results in the performance degradation of small objects. To alleviate this problem, this paper proposes a new patch-level collage fashion data augmentation technique and a new global scheduler based on existing dynamic scale training paradigm. Our new data augmentation can generate collage images with uniform object scales for better augmentation effects. Additionally, our new global scheduler can adjust the strength between different data augmentations to adapt to different stages of the training process. Experiments demonstrate the effectiveness of our techniques. Codes at https://github.com/Andisyc/DataPool.

KEYWORDS

Scale Imbalance, Data Augmentation, Model Brittleness

1. INTRODUCTION

Scale imbalance problem, which can be described as the performance gap between small and large objects [1], has always been a challenge for object detectors. Beside the lack of semantic information in small objects, some researchers believe that there existing different level of imbalance [2] within the scale imbalance problem. When collecting natural images directly from the real world, there may be some over or under representation of certain scales [3] scale existing. Moreover, large objects often generate more loss value than small objects, resulting in suboptimal optimization for small objects [4]. Furthermore, natural data collected directly from the real world do not guarantee a continuous semantic space on the scale level, while neural networks lack scale invariance. All these factors contribute to the performance degradation of detectors.

To address these problems, it is necessary to make some modifications to natural images. One existing approach is collage fashion, for example, *Mosaic* [5]. The collage fashion data augmentation can introduce certain multi-scale effects to training, effectively expanding the feature space. Meanwhile, since modern neural networks have high brittleness, prolonged observation of the same scale can overwrite the previously learned knowledge. The multi-scale effect introduced by collage fashion data augmentations can effectively disrupt the overfitting tendency toward single scale, thereby improving models' generalization ability. In this case, *Mosaic* serves as a random scale generator, allowing the network to observe images with fluctuating scales within a certain range. This ensures that the parameters related to scale within the network remain at reasonable proportions, resulting in minimal overfitting and optimal generalization ability.

However, due to the existence of feature-level imbalance, which means the optimization performance of large objects is always stronger than that of small objects, only relying on Mosaic still exist certain scale imbalances. Therefore, *Dynamic Scale Training* [4] insert a small object batch right after a large object batch. Disrupting the formation of continuous large object batches. This will offset the gradually accumulated large object optimization effects, keeping the scale parameters within the network at the optimal balanced proportion for generalization performance. As shown in Fig. 1, although the green line indicate that *Stitcher* cannot directly generalize to test set, the blue line (*Mosaic* and *DST* simultaneously) can achieve higher performance than the red line (*Mosaic* only) and the orange line (*DST* only). Although as mentioned in [6] while the over-parameterized network is easier to optimize but also more prone to overfitting the distribution of natural datasets. *Mosaic* and *DST* tackle this problem by providing regularization training with ability to reduce the overfitting of certain scales without reducing the representation ability of models. So, the model can maintain the scale balance state, where the generalization ability reaches the peak.

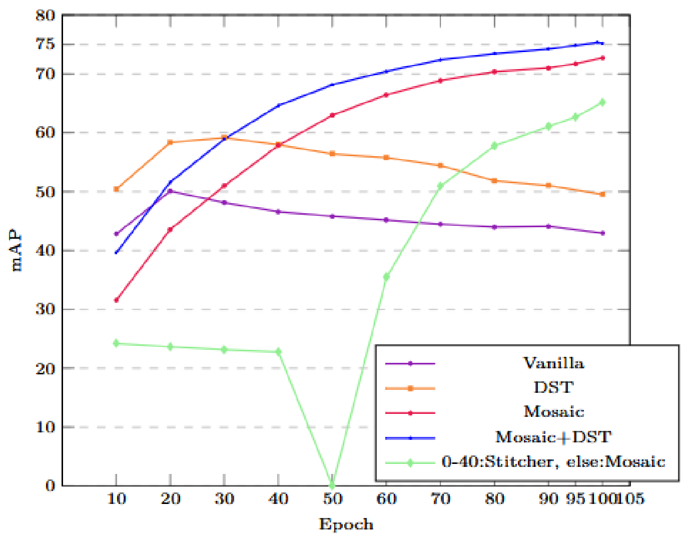

Figure 1. Results on the VOC 2007 test set using different data augmentation techniques on YOLOX-S. *Vanilla* means without any data augmentation. *DST* means adopt both scheduler and data augmentation from *Dynamic Scale Training*. *Stitcher* is the data augmentation component from *Dynamic Scale Training*

However, multi-scale data augmentations, like *Mosaic* and *Stitcher*, is not perfect. Firstly, as shown in Fig. 2, *Stitcher* directly resizes and collages four images together, resulting in small objects being resized too small. Even without recognition significance. Secondly, directly resizing images cannot guarantee small objects. Some large objects still have large scales after resize. Resulting in backward optimization effect. Thirdly, many studies [7], [8] have proved that sufficient, high-quality context information can help models recognize objects. But simply resizing images cannot create the context-object ratio small objects have. As shown in Fig. 3, resizing the train in the left image to match the size of the train in the right image resulting the context far less than the right image.

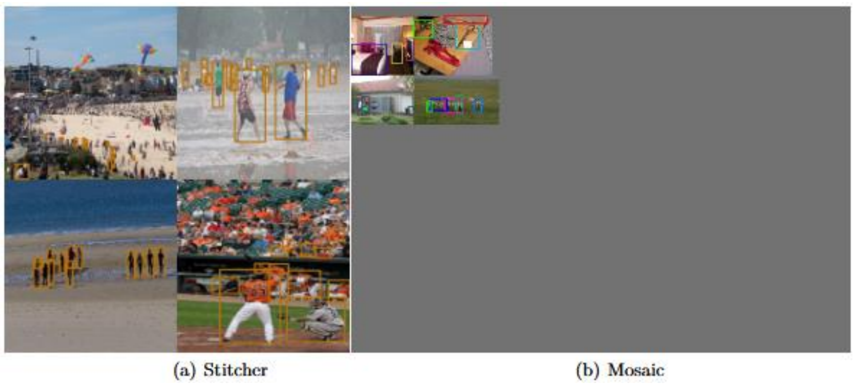

Figure 2. (a): Stitcher from [4]. Notices that the tiny objects in the top-left image no longer have recognition significance. (b): Mosaic from [5]. The multi-scale range is (0.1, 2). Notices that this image have a lot of grey area, which is a waste for training.

To address these issues, we propose DataPool, a new collage fashion data augmentation. Many studies [9] [10] [11] have improved performance by leveraging object-object relationships. These relationships include geometric, semantic, and interaction, etc. But we consider this point from the scale perspective, where objects that appear together often have the same scale. For example, small objects often appear in groups. By decoupling the relationship between images and objects, we can construct a collage image consisting entirely of small objects, as shown in Fig. 8. Compared to Mosaic, DataPool better utilizes the entire spatial area of the image, leaving fewer grey areas. Compared to Stitcher, DataPool avoids excessive scaling, preserving the semantic features of the objects and appropriate amount of context.

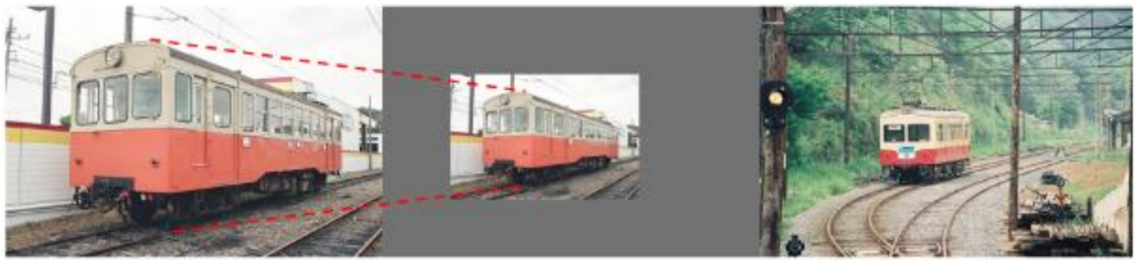

Figure 3.: The left image and the right image are original size. The middle image is obtained by resizing the train in left image to the same size of the train in the left image. Notices that the middle image lacks a significant amount of background, while background information also contributes to the recognition of objects.

In addition, we also modified the existing dynamic scale training scheduler. As the learning rate gradually decreases, the influence disparities between different scales also reduce. Therefore, gradually reducing the frequency of data augmentations can prevent the model moving from overfitting large objects to overfitting small objects. In this way, the model can learn a more balanced scale information, leading to optimal generalization performance.
Our contributions are two-folds:

- Designed a new data augmentation technique for dynamic scale training paradigm - DataPool, as a complement for Stitcher as the data augmentation component for Dynamic Scale Training.
- Improved the existing dynamic scale training scheduler to adjust strength of data augmentation. This new scheduler can automatically modify the training distribution so the model can stay at scale balance state.

2. RELATED WORKS

2.1. Collage Fashion Data Augmentation

The *Dynamic Scale Training* (DST) [4] is a feedback-driven data augmentation aimed at alleviating the scale imbalance problem. Dynamic scale training consists of a scheduler and a data augmentation component. Specifically, When the scheduler detects that the loss value ratio of small objects in the previous batch is below the threshold, data augmentation is triggered. The *Stitcher* will resize four images to one-fourth of their original size and then collage them together to form a single image, which is then used for the next iteration of training.

Mosaic [5] and *RICAP* [12] are both data augmentation techniques belonging to the collage fashion. Both Mosaic and RICAP randomly extract patches from images. These patches are then subject to certain data augmentation processes such as resizing, translation, flipping, and other colour and geometric manipulation. The augmented patches are reassembled based on a selected centre point, and the resulting image is then resized to create a multi-scale image. This approach introduces fluctuations in the object scale, forcing the model to pay attention to secondary regions, increasing object density of images, reducing the model's overfitting to scale and semantics.

2.2. Regularization Effect of Data Augmentation

Hern´andez-Garc´ıa & K¨onig [13] defined explicit regularization techniques as techniques that can reduce the network representation ability, while implicit regularization techniques as techniques that can reduce generalization error or prevent overfitting without reducing the network representation ability. [14] showed that explicit regularization techniques can be completely replaced by data augmentations. Furthermore, [15] found that data augmentation techniques can more easily adapt to different architectures and different datasets, while dropout [16] or weight decay [17] may require more delicate hyper-parameter fine-tuning to adapt into different architectures and data size.

2.3. Model Brittleness & Continue Learning

Many studies have attempted to address the catastrophic forgetting problem [18] during continual learning and imbalanced learning. Replay-based methods consciously select a portion of data through some algorithm for replay based on the adjustment of scheduler. iCaRL [19] conducts class-incremental learning by periodically replaying data stored in memory that best represents a certain class. DST [4] determines whether to initiate data augmentation based on loss values. GSS [20] selects samples for training based on the diversity of gradients. RSS [21] calculates the similarity of feature map representations to choose new samples for training, stabilizing changing rate in feature maps.

Methods based on explicit regularization aim to address catastrophic forgetting by constraining the persistence of network parameters. EWC [22] uses the Fisher information matrix to estimate the importance of weight parameters and guides gradient updates accordingly. OGD [23] projects the gradients from new tasks onto the solutions of old tasks to preserve previously learned knowledge. There are also approaches that apply regularization through a Bayesian perspective, such as [24], using Gaussian processes to impose regularization during training. Additionally, some methods like [25] and [26] employ parameter isolation to fix previously learned knowledge while learning new tasks.

[27] also considered the issue of catastrophic forgetting from a training perspective. It is discussed different lower points caused by hyper-parameter tuning and reducing model capacity to mitigate overfitting can alleviate catastrophic forgetting. However, we consider this oscillation descent process from scale perspective, while utilizing data augmentation as an implicit regularization technique that does not reduce model capacity.

3. METHODS

3.1. Data Pool

The design ethos of *DataPool* is analyse and extract the small object cluster existed on images. With reasonable calculation, object clusters with background can be cut out from images without hurting the integrity of objects. By doing so, an image can be divided into patches, then any kind images can be formed by leveraging these patches.

To do that, we leverage the analysis strategy to decouple the relationship between images and objects, dividing images into patches. Then we leverage the search strategy filters out suitable patches. Finally, we utilize the synthesis strategy, incorporating sensible cuts and minimizing resizing, to handle patches in a way that prevents excessive resizing and distortion of objects. The overall pipeline of the DataPool is displayed in Fig. 4.

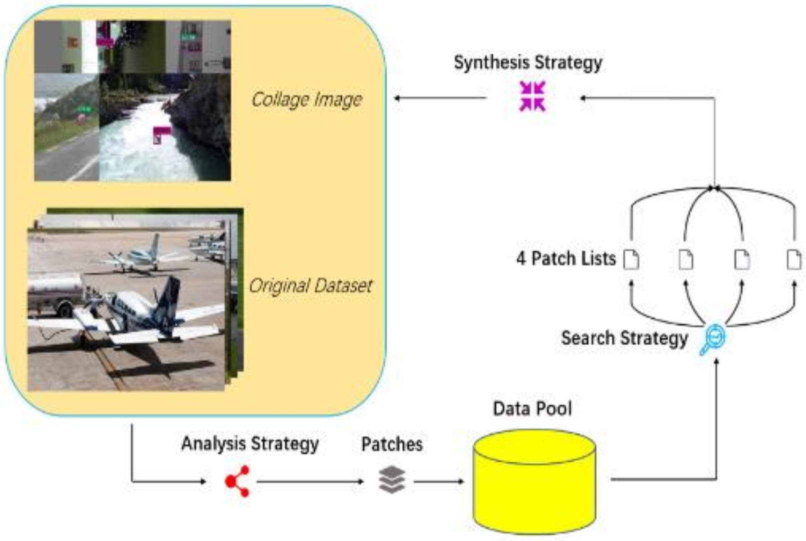

Figure 4.: DataPool Pipeline. First, divide images into patches. Then, extract patches based on size requirements. Finally, a synthesized image with a specific object scale is generated.

First, we attempt to redefine the object scale. We set the object scale ratio to the square root of the ratio between the object area and the image area. This is because we aim to obtain a linear scale space, while the object area actually is quadratic growth with respect to width and height. Using a ratio and linear metric space makes the variations smoother and reduces the impact of specific thresholds. Based on experience, we divide objects into five scales, as shown in following Equation:

$$\text{ObjectScale} = \begin{cases} \text{Tiny}, & height\ \&\ width \leq 32 \\ \text{Small}, & 0.00 < (\frac{obj\ area}{img\ area})^{\frac{1}{2}} \leq 0.15 \\ \text{LowerMiddle}, & 0.15 < (\frac{obj\ area}{img\ area})^{\frac{1}{2}} \leq 0.30 \\ \text{UpperMiddle}, & 0.30 < (\frac{obj\ area}{img\ area})^{\frac{1}{2}} < 0.59 \\ \text{Large}, & 0.59 \leq (\frac{obj\ area}{img\ area})^{\frac{1}{2}} < \infty \end{cases}$$

We further divide the "Middle" scale into "LowerMiddle," which is closer to the small objects, and "UpperMiddle," which is closer to the large objects.

[28] found that although the class space is discrete, the information learned by neural networks is actually continuous due to the similarity between neighbouring samples. We consider this from a scale perspective. As shown in Fig. 5, the accuracy is lower when there are objects only on the left-hand side of the "Small" scale compared to when there are objects on both sides. Therefore, we set the maximum scale of the objects on the synthesized DataPool image to "LowerMiddle" instead of "Small," ensuring that the average scale remains close to "Small."

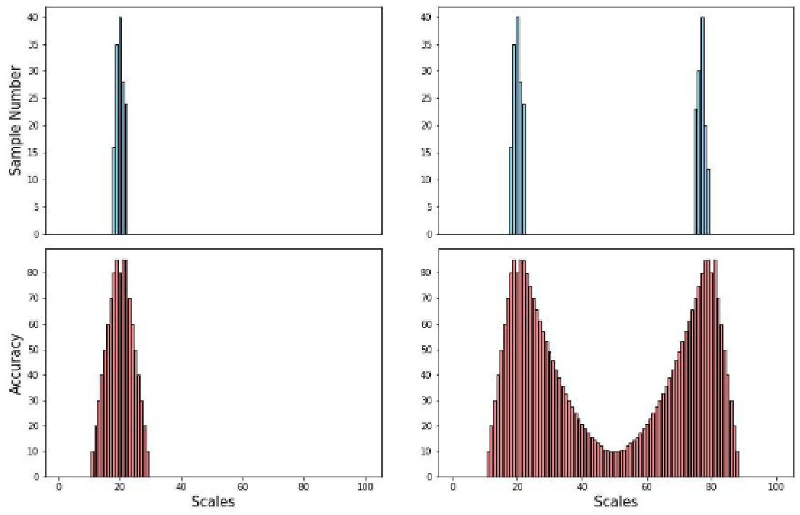

Figure 5. A concept figure of cross-scale object similarity. Left: the accuracy of detectors that trained with samples only exist within a small range. Right: the accuracy of detectors that trained with samples exist within two scale ranges.

For the object belonging to Tiny, Small, or LowerMiddle, object clusters will be estimated by merging objects outward based on it as the centre, as shown in Fig. 6. When an object is classified as a small or relatively small object, we first check if it overlaps with other objects. If the object does overlap with other objects, we then determine if the other objects have additional overlapping objects. By gradually identifying all overlapping objects, outliers and excessively large objects can be excluded. Finally, the coordinates of the composite large object formed by all the merged objects will be calculated and start merging an appropriately amount of background, until encountering distant non-overlapping objects. This process generates image patches only consisted of small objects. At this point, patches can still expand outward to merge other nearby non-overlapping objects.

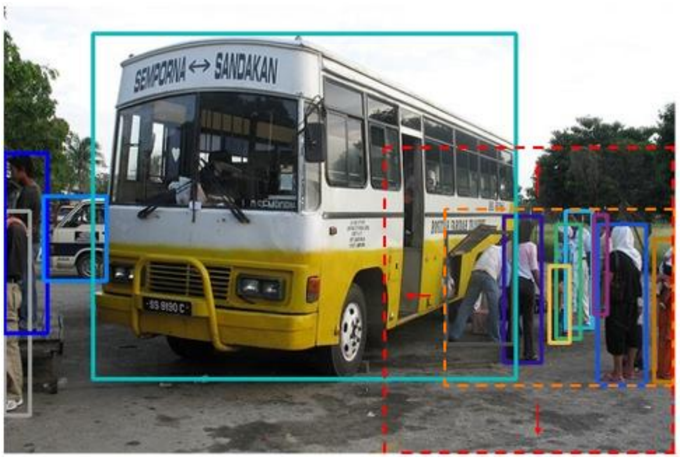

Figure 6. The process of forming an object cluster. The orange dashed box is the object cluster obtained by the analysis process. The cluster area is formed by the four farthest object edges. The red dashed box is obtained by appropriately expanding the cluster area outward to include a certain amount of background. The coordinates of red dashed box will be recorded as a patch for the synthesis process.

Although some objects do not overlap with each other, they are still closely clustered together. In these cases, expanding patches outward from a central object and merge other objects that are close in scale and distance. Starting from the four vertices of the current object cluster, patches expand in the opposite direction up to one-third of the image width and one-half of the image height, merging all the small objects (Tiny, Small, LowerMiddle) encountered along the way, as shown in Fig. 7. The expansion stops when encountering large objects (UpperMiddle, Large). This process generates an image patch consisting of small objects. The final synthesis result is shown in Fig. 8.

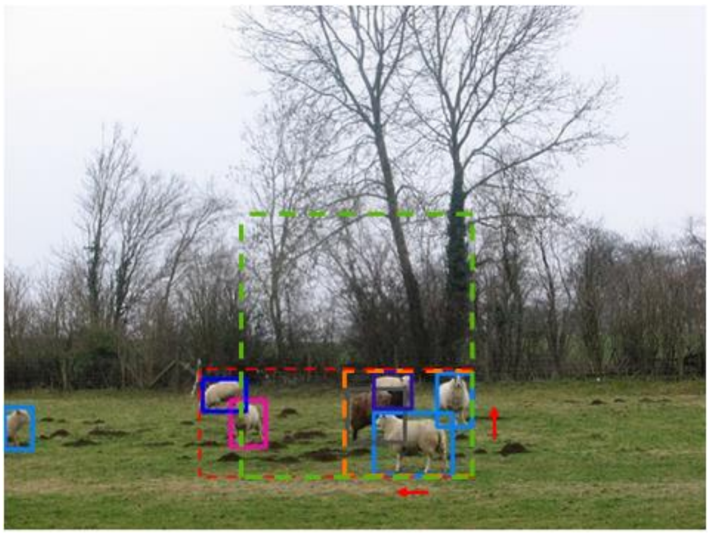

Figure 7. The process of merging additional objects. The orange dashed box is the already analyzed object cluster. The green dashed box is the expansion area start from the right-bottom corner. The red dashed box is the final object cluster.

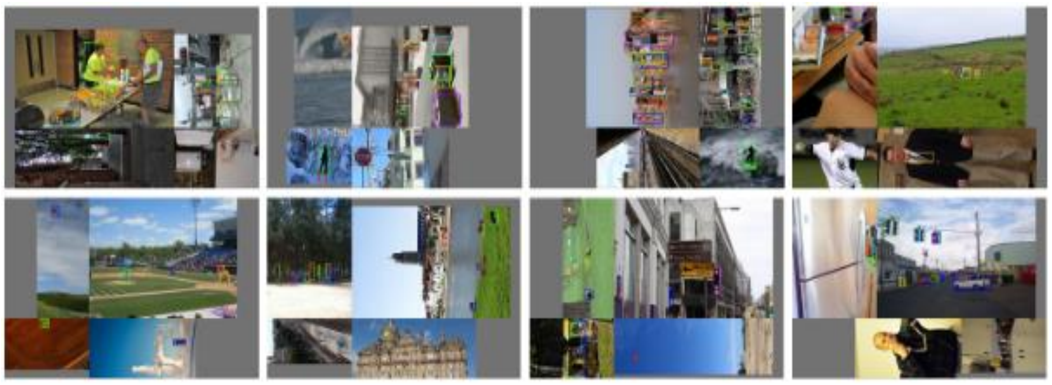

Figure 8. DataPool Collage Images. Object scale of first row images is [0, 64×64]. Object scale of second row images is [0, 32×32].

3.2. Hybrid Scheduler

Catastrophic forgetting [29], which also known as stability-plasticity dilemma [27], has long troubled researchers. [30] suggests that neural networks are highly brittleness yet inefficient because they require extensive training data while struggling to learn new knowledge without forgetting old knowledge. This is related to the learning behaviour of network, as depicted by the peculiar green curve in Fig. 1. Although the parameters learned within the network can generalize to small objects when trained with *Stitcher*, switching to *Mosaic* destroy the existing parameter distribution, as the gradient simultaneously affects all parameters. The scale information contained in the gradient can also counteract and alter the neurons with other scale information, resulting in oscillate descent when training with small batches. From a scale perspective, by controlling the scale distribution within a certain training period, the direction of oscillate descent will not deviate, thereby ensuring a path towards a lower point.

[31] states that neurons within the network can be divided into four categories, as shown in Fig. 9. For a given class, the red neurons are the main generalizing neurons, while the orange neurons are overfitting neurons that hinder the model's generalization to the test set. The yellow neurons are generalizing neurons that may perform poorly on the training set but contribute to the network's generalization to the test set, and the blue neurons are neurons that learn information from other classes. During normal training, due to the stronger optimization effect on large objects, the large objects neurons gradually overfit, leading to an increase in the orange region. When using scheduler-guided data augmentation, the tendency to form continuous batches of large objects is interrupted, reducing the excessive accumulation of optimization effects on large objects. This helps maintain a balanced state of scale-related parameters within the network, reducing the overfitting neurons, and increasing the generalization ability of other scales.

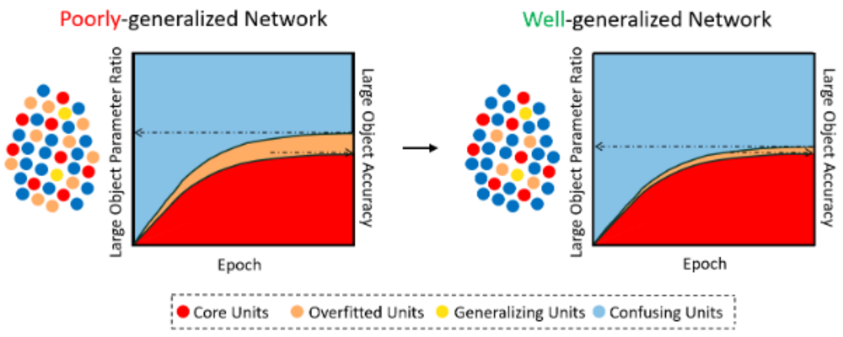
Poorly-generalized Network
Well-generalized Network
Large Object Parameter Ratio
Large Object Accuracy
Epoch
Large Object Parameter Ratio
Large Object Accuracy
Epoch
● Core Units ● Overfitted Units ● Generalizing Units ● Confusing Units

Figure 9. A concept figure of network generalization. Left: a normally trained network. Since large objects have stronger optimization effect, the overfitting neurons (orange colour) of large objects gradually increase during training. Although the parameters related to large objects (orange area + red area) occupy a relatively large proportion within the network, the generalization ability is only contributed by the red area. Right: a better generalize network. There are relatively fewer orange neurons, and the overfitting parameters of large objects (orange area) increase less compared to the normally trained network.

With training going on, the learning rate gradually decays, and the influence of large and small objects becomes more balanced, as shown in following Equation:

$$\Delta = \alpha \cdot (\frac{\partial L_{Large}}{\partial \theta} - \frac{\partial L_{Small}}{\partial \theta}),$$

where Δ is the influence difference, L_{Large} and L_{Small} are the loss values of one large object and one small object, θ is the weights of the network. If the frequency of data augmentation is not reduced, the network will gradually shift from overfitting large objects to overfitting small objects, as exhibited in *Section 4.2*. Therefore, we directly adopt the cosine anneal learning rate scheduler [33] to control the data augmentation, as shown in following Equation:

$$\mathrm{LR} = 0.5 \cdot prob \cdot (1 + \cos(\pi \cdot \frac{current\_epoch}{total\_epoch})).$$

We directly set the prob value as 0.1 from YoloX [32]. This allows the scheduler to be easily integrated into different architectures without the need to fine-tune any hyper-parameters.

4. EXPERIMENTS

4.1. Experiment Setup

The first part experiments are conducted with YOLOX-S on Pascal VOC [33]. The input image size is 512×512. The Mosaic prob is 1 and the range is (0.1, 2). Mosaic will be closed in the last 5 epochs. The optimizer is yoloxwarmcos with weight decay as 5e-4 and momentum as 0.9.

The second part experiments are conducted with RetinaNet [34] on COCO [35]. The input image size is 640 × 1024. Mosaic was closed in this part of experiments. The optimizer is WarmupCosineLR with weight decay as 1e-4 and momentum as 0.9.

The object scale follows the MSCOCO definition: [0, 32×32], [32×32, 96×96], [96×96, ∞]. The GPU used for training and testing is a Tesla V100-SXM2-32GB.

4.2. The influence of Batch Size & Object Range

First, we tested the effectiveness of DataPool on VOC dataset. We employed the original dynamic scale training strategy, but replaced the data augmentation from Stitcher to DataPool. Based on [36] and limited computation resources, we conducted small batch training, in which batch size equal to 8, 16, and 32. Additionally, we tested two DataPool object scales, [0, 32×32], [0, 64×64], to find out which one is better.

Table 1. Across techniques and batch sizes results on the VOC 2007 test set with YOLOX-S. AP_{50} comes from evaluation codes cross-class. AP_s, AP_m and AP_l come from evaluation codes cross-scale. DataPool 32

means that object scale range is [0, 32×32]. DataPool 64 means that object scale range is [0, 64×64]. The scheduler in this table is the original dynamic scale training scheduler.

Augment	**Batch Size**	**AP**	**AP_{50}**	**AP_{75}**	**AP_s**	**AP_m**	**AP_l**
Stitcher	8	55.10	78.09	60.16	43.05	67.13	85.62
DataPool 32	8	53.76	77.00	58.55	43.07	66.96	84.53
DataPool 64	8	54.91	77.81	59.78	43.13	68.03	85.40
Stitcher	16	54.85	77.94	59.64	40.64	67.28	85.32
DataPool 32	16	54.69	77.94	59.57	42.24	66.64	85.09
DataPool 64	16	55.07	78.23	60.24	40.38	67.70	85.92
Stitcher	32	55.10	78.34	60.35	41.11	66.95	85.32
DataPool 32	32	54.32	77.48	59.64	42.56	66.70	85.14
DataPool 64	32	55.32	78.72	60.97	42.48	67.21	83.41

From Table 1, It can be observed that compared to the other two configurations, DataPool 32 performs lower. This is because, compared to Stitcher and DataPool 64, DataPool 32 has a lower average scale and cannot benefit from slightly larger adjacent-scale objects for small objects, as shown in Fig. 5. DataPool 32 has relatively higher AP_s and lower AP_m, which means that overfitting to small objects has led to a decrease in overall generalization capability.

Additionally, as the batch size increases, the performance of DataPool 64 gradually surpasses Stitcher. When the batch size is 32, compared to Stitcher, DataPool 64 exhibits lower AP_l(-1.91%) but higher AP_s(+1.37%) and AP_m(+0.26%), resulting overall higher generalization capability. This suggests that after eliminating large objects from images, a more focused optimization effect allows the network to achieve a more balanced state, resulting in higher generalization performance.

4.3. New Automatic Scheduler

In order to further investigate the effect of the scheduler during training, we fixed the ratio between the normal batch and the augmented batch, as the fixed ratio scheduler. We also directly adopt the YoloX cosine anneal learning rate scheduler as the decay scheduler to compare with the fixed ratio scheduler. This comparison aims to verify the concept of maintaining a balanced state as the learning rate decreases.

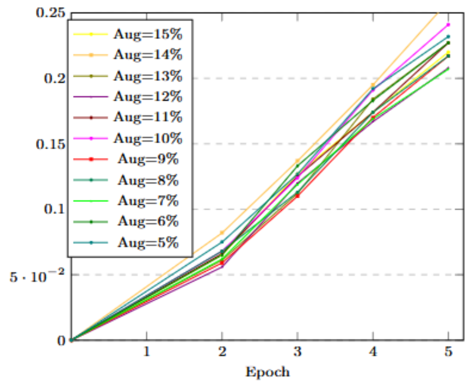

Figure 10. Initial value search with batch size 8. The data for the first six epochs is displayed, and each experiment is run three times.

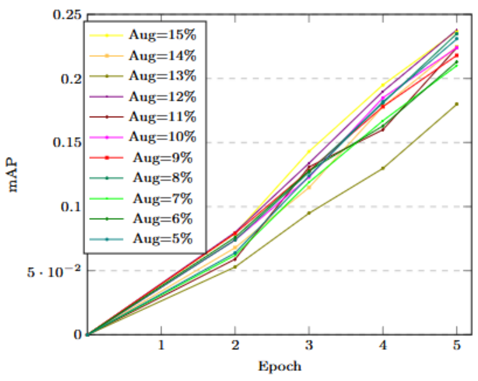

Figure 11. Initial value search with batch size 16. The data for the first six epochs is displayed, and each experiment is run three times.

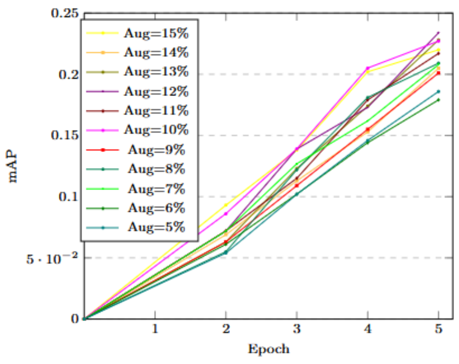

Figure 12. Initial value search with batch size 32. The data for the first six epochs is displayed, and each experiment is run three times.

Since data augmentation like Stitcher, which will reduce the average scale of the image, cannot directly generalize, as shown by the green line in Fig. 1, it is crucial to choose the initial ratio between normal batches and augmented batches. To determine the initial ratio, we employed the initial value search technique from [36]. Based on the experiences, we set the initial ratio search range to [85, 95]. As shown in Fig.12, when the batch size is 32 (the right figure), the optimal initial value is 90. When the batch size is 16 (Fig.11) and 8 (Fig.10), 90 remains in a competitive position. Therefore, we set the initial value to 90.

Table 2. Scheduler comparison results on the VOC 2007 test set with YOLOX-S. AP_{50} comes from evaluation codes across-class. AP_s, AP_m and AP_l come from evaluation codes across-scale. DataPool 32 means that object scale range is [0, 32×32]. DataPool 64 means that object scale range is [0, 64×64].

Method	Batch Size	AP_{50}	AP_s	AP_m	AP_l
Ratio / Cosine Stitcher	8	77.30 / 78.06	41.18 / 41.04	66.29 / 67.70	84.68 / 85.25
Ratio / Cosine Stitcher	16	78.10 / 78.29	41.01 / 43.30	67.11 / 67.48	84.96 / 85.27
Ratio / Cosine Stitcher	32	78.25 / 78.26	43.04 / 43.25	66.85 / 67.56	85.53 / 85.42
Ratio / Cosine DataPool 64	8	76.90 / 77.23	41.68 / 39.42	66.25 / 66.38	83.52 / 84.76
Ratio / Cosine DataPool 64	16	76.57 / 78.17	42.39 / 41.49	66.87 / 67.84	83.50 / 85.42
Ratio / Cosine DataPool	32	77.96 /	41.81 /	68.16 /	84.92 /

64		78.52	41.97	67.23	85.20

Table 2 shows that compared to the fixed ratio scheduler, the accuracy is generally higher when using the decay scheduler. This is because as the learning rate decays, the influence differences between small and large objects gradually decrease. Maintaining the same data augmentation frequency in the later stages of training as in the earlier stages can cause the network to gradually overfit small objects. Therefore, a decay scheduler that gradually reduces the data augmentation frequency can help maintain a scale balance state.

4.4. Across Model & Dataset Type & Dataset Size

In this section, we tested the performance of our methods on the COCO dataset with RetinaNet. We employed the original setup of *Dynamic Scale Training*}, using RetinaNet-50 and the full-size COCO dataset, without *Mosaic*. The only modification we made was switching *the Multi Step learning rate scheduler* to *the Cosine Anneal learning rate scheduler* to accommodate the hybrid scheduler. The batch size is 16, and the DataPool object scale is [0, 64×64], with 640 × 1024 image size.

Table 3. Across dataset size results on the COCO test set with RetinaNet-50. *Full* means full training dataset. *Half* means half training dataset. *Quar* means quarter training dataset. DataPool 64 means that object scale range is [0, 64×64].

Method	**Dataset Size**	**AP**	**AP_{50}**	**AP_{75}**	**AP_s**	**AP_m**	**AP_l**
Baseline (Vanilla)	Full	34.87	54.10	36.29	21.54	38.63	43.75
DST + Stitcher	Full	36.30	55.36	37.19	23.17	40.71	45.18
DST + DataPool 64	Full	36.24	55.12	36.96	22.85	40.38	45.17
Hybrid + Stitcher	Full	37.26	55.45	37.89	24.45	41.69	46.71
Baseline (Vanilla)	Half	36.64	55.71	37.24	23.03	40.61	47.00
DST + Stitcher	Half	36.48	55.38	36.99	23.62	40.65	46.00
DST + DataPool 64	Half	34.11	53.77	36.21	22.76	37.51	42.02
Hybrid + Stitcher	Half	37.34	55.65	37.93	23.64	41.56	47.25
Baseline (Vanilla)	Quar	28.22	45.27	28.74	14.33	30.35	36.35
DST + Stitcher	Quar	36.35	55.24	36.87	23.18	40.43	45.11
DST + DataPool 64	Quar	33.95	54.84	35.91	21.32	37.43	42.39
Hybrid + Stitcher	Quar	37.27	55.54	37.99	24.27	41.74	46.77

From Table 3, it can be observed that when using datasets of full, half, and quarter sizes, the Hybrid Scheduler improves the performance by AP+0.96%, AP+0.86%, and AP+0.92%, respectively, compared to the original dynamic scale training scheduler. This demonstrates that maintaining a scale balanced state within the network and minimizing overfitting neurons are essential for achieving optimal generalization.

Table 4. Overall: iteration number of overall optimal performance. SmallObj: iteration number of small objects optimal performance. AP, AP_s are obtained at the small objects optimal generalization iteration.

Method	**Dataset Size**	**Overall**	**SmallObj**	**AP**	**AP_s**
Baseline (Vanilla)	Full	88k	67k	33.56	22.32
DST + Stitcher	Full	87k	87k	36.30	23.02
Hybrid + Stitcher	Full	86k	86k	37.26	24.33
Baseline (Vanilla)	Half	88k	77k	36.43	23.19
DST + Stitcher	Half	90k	81k	36.37	23.88
Hybrid + Stitcher	Half	89k	74k	36.69	24.22

Baseline (Vanilla)	Quar	56k	41k	27.76	15.22
DST + Stitcher	Quar	86k	63k	35.27	23.11
Hybrid + Stitcher	Quar	90k	73k	36.54	24.13

From Table 4, it can be observed that the overall best performance point is different from the best small objects performance point. The overall performance at best small objects performance point is lower compared to the optimal overall performance. This is because the network becomes more biased towards small objects, squeezing the parameter space of other objects. When overfitting occurs on a single scale, whether it is for large or small objects, it will lead to a decrease in overall generalization performance. Only when the parameter proportions for each scale are balanced can the overall generalization performance reach its best.

5. CONCLUSIONS

In this paper, we propose a new data augmentation technique called DataPool and improve the existing dynamic scale training scheduler to alleviate the scale imbalance problem. By decoupling the relationship between images and objects, DataPool can construct more suitable images for training. These images have objects within a limited scale range, preserving the context-object ratio as much as possible and avoiding excessive resizing objects. The improved dynamic scale training scheduler gradually reduces the frequency of data augmentation as the learning rate decreases, maintaining the network in a scale balance state. Experiment results demonstrate that using the new data augmentation technique and the improved scheduler can further improve the performance across different network architectures, different datasets, and different dataset sizes.

ACKNOWLEDGEMENTS

The authors would like to thank everyone, just everyone!

REFERENCES

[1] Singh, B., & Davis, L. S. (2018). An analysis of scale invariance in object detection snip. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3578-3587).

[2] Oksuz, K., Cam, B. C., Kalkan, S., & Akbas, E. (2020). Imbalance problems in object detection: A review. *IEEE transactions on pattern analysis and machine intelligence*, *43*(10), 3388-3415.

[3] Kisantal, M., Wojna, Z., Murawski, J., Naruniec, J., & Cho, K. (2019). Augmentation for small object detection. *arXiv preprint arXiv:1902.07296*.

[4] Chen, Y., Zhang, P., Li, Z., Li, Y., Zhang, X., Qi, L., ... & Jia, J. (2020). Dynamic scale training for object detection. *arXiv preprint arXiv:2004.12432*.

[5] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.

[6] Frankle, J., & Carbin, M. (2018). The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

[7] Divvala, S. K., Hoiem, D., Hays, J. H., Efros, A. A., & Hebert, M. (2009, June). An empirical study of context in object detection. In *2009 IEEE Conference on computer vision and Pattern Recognition* (pp. 1271-1278). IEEE.

[8] Zhang, M., Tseng, C., & Kreiman, G. (2020). Putting visual object recognition in context. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 12985-12994).

[9] Liu, Y., Wang, R., Shan, S., & Chen, X. (2018). Structure inference net: Object detection using scene-level context and instance-level relationships. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6985-6994).

[10] Yu, R., Chen, X., Morariu, V. I., & Davis, L. S. (2016). The role of context selection in object detection. *arXiv preprint arXiv:1609.02948*.

[11] Dvornik, N., Mairal, J., & Schmid, C. (2018). Modeling visual context is key to augmenting object detection datasets. In *Proceedings of the european conference on computer vision (ECCV)* (pp. 364-380).

[12] Takahashi, R., Matsubara, T., & Uehara, K. (2019). Data augmentation using random image cropping and patching for deep CNNs. *IEEE Transactions on Circuits and Systems for Video Technology*, *30*(9), 2917-2931.

[13] Hernández-García, A., & König, P. (2018). Data augmentation instead of explicit regularization. *arXiv preprint arXiv:1806.03852*.

[14] Hernández-García, A., & König, P. (2018). Do deep nets really need weight decay and dropout?. *arXiv preprint arXiv:1802.07042*.

[15] Hernández-García, A., & König, P. (2018). Further advantages of data augmentation on convolutional neural networks. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27* (pp. 95-103). Springer International Publishing.

[16] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929-1958.

[17] Hanson, S., & Pratt, L. (1988). Comparing biases for minimal network construction with back-propagation. *Advances in neural information processing systems*, *1*.

[18] McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation* (Vol. 24, pp. 109-165). Academic Press.

[19] Rebuffi, S. A., Kolesnikov, A., Sperl, G., & Lampert, C. H. (2017). icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (pp. 2001-2010).

[20] Aljundi, R., Lin, M., Goujaud, B., & Bengio, Y. (2019). Gradient based sample selection for online continual learning. *Advances in neural information processing systems*, 32.

[21] Kalb, T., Mauthe, B., & Beyerer, J. (2022, October). Improving replay-based continual semantic segmentation with smart data selection. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)* (pp. 1114-1121). IEEE.

[22] Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., ... & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, *114*(13), 3521-3526.

[23] Farajtabar, M., Azizan, N., Mott, A., & Li, A. (2020, June). Orthogonal gradient descent for continual learning. In *International conference on artificial intelligence and statistics* (pp. 3762-3773). PMLR.

[24] Titsias, M. K., Schwarz, J., Matthews, A. G. D. G., Pascanu, R., & Teh, Y. W. (2019). Functional regularisation for continual learning with gaussian processes. *arXiv preprint arXiv:1901.11356*.

[25] Jerfel, G., Grant, E., Griffiths, T., & Heller, K. A. (2019). Reconciling meta-learning and continual learning with online mixtures of tasks. *Advances in neural information processing systems*, *32*.

[26] Li, X., Zhou, Y., Wu, T., Socher, R., & Xiong, C. (2019, May). Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International conference on machine learning* (pp. 3925-3934). PMLR.

[27] Mirzadeh, S. I., Farajtabar, M., Pascanu, R., & Ghasemzadeh, H. (2020). Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems*, *33*, 7308-7320.

[28] Yang, Y., Zha, K., Chen, Y., Wang, H., & Katabi, D. (2021, July). Delving into deep imbalanced regression. In *International conference on machine learning* (pp. 11842-11851). PMLR.

[29] Zhao, Y., Zhang, H., & Hu, X. (2024, May). Role Taxonomy of Units in Deep Neural Networks. In *Proceedings of UniReps: the First Workshop on Unifying Representations in Neural Models* (pp. 291-301). PMLR.

[30] Loshchilov, I., & Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

[31] Ge, Z., Liu, S., Wang, F., Li, Z., & Sun, J. (2021). Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*.

[32] Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision*, *88*, 303-338.

[33] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980-2988).

[34] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13* (pp. 740-755). Springer International Publishing.

[35] Masters, D., & Luschi, C. (2018). Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*.

[36] Smith, L. N. (2017, March). Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)* (pp. 464-472). IEEE.

AUTHORS

Yuxuan Cheng is currently an MPhil student at UIC.