ShapeAug++: More Realistic Shape Augmentation for Event Data

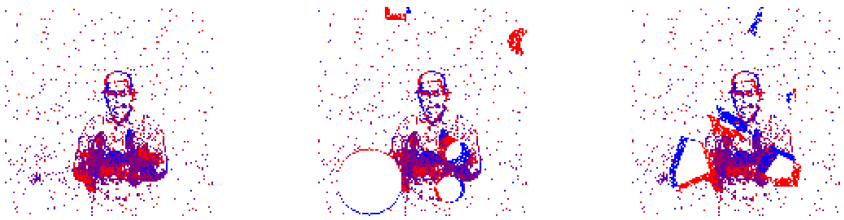
Katharina Bendig[1], René Schuster[1,2], and Didier Stricker[1,2]

[1] RPTU - University Kaiserslautern-Landau, Germany
[2] DFKI - German Research Institute for Artifical Intelligence, Germany
firstname.lastname@dfki.de

Abstract. The novel Dynamic Vision Sensors (DVSs) gained a great a mount of attention recently as they are superior compared to RGB cameras in terms of latency, dynamic range and energy consumption. This is particularly of interest for autonomous applications since event cameras are able to alleviate motion blur and allow for night vision. One challenge in real-world autonomous settings is occlusion where foreground objects hinder the view on traffic participants in the background. The ShapeAug method addresses this problem by using simulated events resulting from objects moving on linear paths for event data augmentation. However, the shapes and movements lack complexity, making the simulation fail to resemble the behavior of objects in the real world. Therefore in this paper, we propose ShapeAug++, an extended version of ShapeAug which involves randomly generated polygons as well as curved movements. We show the superiority of our method on multiple DVS classification datasets, improving the top-1 accuracy by up to 3.7% compared to ShapeAug.

Keywords: Event Camera Data · Augmentation · Classification.

1 INTRODUCTION



(a) Original (b) ShapeAug (c) ShapeAug++

Fig. 1: Visualization of the ShapeAug and ShapeAug++ augmentation methods on *DVS-Gesture* [1].

The field of autonomous driving advances rapidly and has the potential to vastly change everyday life for a great amount of people. Therefore, it is imperative that the robustness and safety of this technology has to be a priority. One promising progression is the development of Dynamic Vision Sensors (DVSs), also known as event cameras. These vision sensors, unlike unconventional RGB cameras, register brightness changes asynchronously instead of absolute intensity values at a pre-defined frame rate. This enables them to record visual input with an exceedingly low latency, in the range of milliseconds. DVSs therefore allow a crucially fast detection of other traffic participants, which are able to move multiple meters in mere seconds. In addition, event cameras have a high dynamic range (in the range of 140 dB) and can thus record motion information even at night time and during poor lighting conditions. Due to their asynchronous pixels, DVSs moreover have a low energy consumption, allowing for their utilization in any mobile application and leading to a comparably low carbon footprint.

However, given that this is a relatively recent technology, the amount of data available for Deep Learning (DL) approaches is quite limited. In comparison to RGB datasets like ImageNet [4], which contains 14 million images, event datasets such as N-CARS [20] have only a few thousand labeled images. Consequently, data augmentation becomes crucial to prevent overfitting and increase the robustness of neural networks. Even with larger datasets like the Gen1 Automotive Event Dataset [3], other works have shown that simple geometric augmentation methods can significantly boost the performance by up to 25% [9].

Especially in the context of autonomous driving systems, occlusion is another great challenge for DL detection methods. Occlusion describes the (partial) covering of background objects by other foreground items. In order to avoid dangerous accidents, detection methods must be capable to detect these occluded objects nevertheless. Since even the labeling of occluded items is challenging, data augmentation methods become crucial to ensure the robustness and sufficient detection by neural networks. Many augmentation methods for handling occlusion like [10,21] typically remove data over time or in specific areas. However, this can only simulate an occluding object moving synchronously with the camera, which is not representative of real-world scenarios. Automotive settings are inherently dynamic, meaning most objects within the scene are in motion independent of the camera's ego-motion, which cannot be accurately represented by simply dropping events at a fixed location.

For this reason, we have introduced a method for the more realistic simulation of occluding objects which move in the foreground in our work ShapeAug [2]. To do so, random objects in the form of squares and circles are generated, which move along linear paths into a random direction. Moreover, the resulting events from this movement are simulated and applied into the foreground of the scene. Our experiments show, that ShapeAug is able to increase the performance for classification tasks as well as the robustness against various challenging validation data.

However, perfect squares and linear movements are rarely encountered in real-world scenarios. To further improve performance and realism in simulated

occlusions, we introduce **ShapeAug++**. This advanced method significantly increases the complexity of simulated objects and their movements. **ShapeAug++** can generate various polygons by using the convex hull of randomly distributed points. It also incorporates arbitrary Bézier curves for movement and includes object rotations around their own axes. We show that our improved augmentation technique is able to outperform ShapeAug on the most common event datasets for classification.

2 RELATED WORK

2.1 Occlusion-aware RGB Image Augmentation

One common approach to augmenting RGB images, which can be viewed as occlusion simulation, involves removing (zeroing out) specific regions of the image. Hide-and-Seek [19] is one such method, which divides the image into a fixed number of patches and assigns each patch a probability of being removed. Alternatively, the Cutout method [5] abandons the rigid grid structure by randomly selecting a fixed number of center points in the image and removing squares of a predefined side length around these points. Building on these methods, the work by [8] incorporates a gradient-based saliency approach along with Batch Augmentation [12]. However, these methods do not effectively replicate real-world event occlusion, where occlusion in consecutive frames needs temporal correlation, and moving foreground objects would generate events themselves.

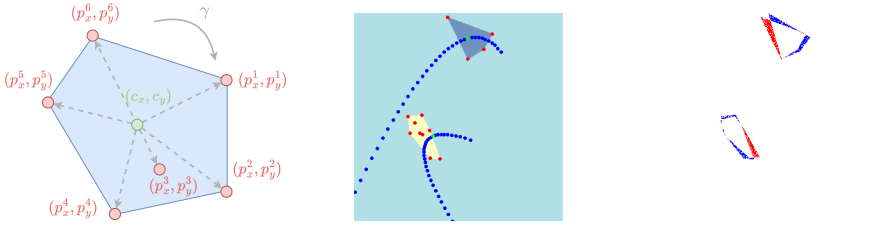
2.2 Event Data Augmentation

Event data augmentation techniques often stem from methods used for RGB images. For example, [15] applies various geometric augmentations – such as horizontal flipping, shifting, rotation as well as Cutout [5] and CutMix [23]. As in our previous work ShapeAug [2], we apply geometric augmentation during all our experiments.

CutMix combines two samples and their labels through linear interpolation. EventMix [18] applies this idea to event data but falls short in realistically modeling occlusion, as it does not account for situations where a foreground object fully covers a background object.

Inspired by Dropout [21], the work of [10] introduces EventDrop, which drops events randomly based on time and area. EventRPG [22] extends this idea by computing saliency maps for Spiking Neural Networks allowing for a relevance based application of EventDrop and EventMix. These approaches, however, are unable to adequately simulate occlusion in dynamic real-world scenes because moving objects would still generate events unless perfectly synchronized with the camera.

Our previous method ShapeAug [2] overcomes this limitation by simulating both the occlusion caused by foreground objects and the events resulting from their movement. However, the simplistic nature of the objects (squares and circles) and their linear movements does not capture the complexity of natural



(a) Shape Generation (b) Example Frame (c) Resulting Events

Fig. 2: Visualization of (a) the shape generation process, (b) an example generated frames for ShapeAug++ and (b) the resulting events. The path along the Bézier curves is illustrated with blue points, the center point is highlighted in green, and the offset points are marked in red. These points are displayed here for visualization purposes only.

shapes and motion. In this paper, we build upon ShapeAug by introducing more complex shapes based on polygons as well as curved movements and rotations.

3 Method

3.1 Event Data Handling

Following the example of ShapeAug, we construct event histograms E based on the asynchronous events $e_i = (x_i, y_i, t_i, p_i)$ consisting of the pixel position (x_i, y_i) and time t_i of an event as well as its polarity p_i. The event histogram E has the shape $(T, 2, H, W)$ with (H, W) as the height and width of the event sensor and T as the number of timesteps one event sample is split into. Mathematically, E can be constructed in the following manner:

$$\mathcal{E}(\tau, p, x, y) = \sum_{e_i \in \mathcal{E}} \delta(\tau - \tau_i)\delta(p - p_i)\delta(x - x_i)\delta(y - y_i), \tag{1}$$

$$\tau_i = \left\lfloor \frac{t_i - t_a}{t_b - t_a} \cdot T \right\rfloor, \tag{2}$$

with $\delta(\cdot)$ as the Kronecker delta function.

In our classification experiments, the timesteps are fed sequentially into the network, as we are utilizing a Spiking Neural Network (SNN).

While we use event histograms due to their popularity for densifying events and simplifying neural network handling, our method can theoretically work with any event representation.

3.2 Object Generation

To ensure a wide variety of complex shapes, we employ a method to generate objects as random polygons, which is visualized in Figure 2. The number of

shapes n_s is a random integer sampled from a uniform distribution between 1 and the maximum number of shapes N_s. Then for each shape a random center point (c_x, c_y) within the frame with dimensions W, H is selected. The coordinates of the center point are drawn from uniform distributions:

$$(c_x, c_y) = (\mathcal{U}(0, W-1), \mathcal{U}(0, H-1)). \tag{3}$$

For each shape, we generate a random number of vertices N_p, which is sampled uniformly between 6 and 10. The position of each vertex is defined relative to the center point and is calculated as:

$$(p_x, p_y) = (c_x + \mathcal{U}(-s, s), c_y + \mathcal{U}(-s, s)), \tag{4}$$

with $s = \mathcal{U}(s_{min}, s_{max})$ as the size for the shape.

The polygons are constructed by computing the convex hull of the set of vertices so that the shapes are still big enough to cause occlusion. Moreover each polygon as well as the background are assigned a random color, contributing to the diversity of the generated shapes and introducing more randomness to the polarity of events.

3.3 Object Movement

To simulate more complex and natural movements, we generate the paths of the objects using random quadratic Bézier curves. Each curve is defined by three points: The starting point $P_0 = (c_x, c_y)$, the control point $P_1 = (\mathcal{U}(-\frac{W}{2}, \frac{W}{2}), \mathcal{U}(-\frac{H}{2}, \frac{H}{2}))$, and the end point P_2, which is a random point outside the frame by a margin of s to ensure the shape moves out of the frame. To compute n points on the quadratic Bézier curve, we can utilize the following equations:

$$t_i = \frac{i}{n-1}, \quad \text{for each } i \in [0, n-1], \tag{5}$$

$$\mathbf{B}(t_i) = (1-t_i)^2\mathbf{P}_0 + 2(1-t_i)t_i\mathbf{P}_1 + t_i^2\mathbf{P}_2. \tag{6}$$

For each frame, the center point of the object is set to the next point $\mathbf{B}(t_i)$ on the curve. Additionally, for each shape, we randomly choose an angle $\gamma = \mathcal{U}(-10, 10)^\circ$ and rotate it by this angle at each step.

These complex movements result in significant variety, especially in the events which are formed on different sides of the objects depending on their direction of movement. Moreover, due to the non-uniform distribution of points along a Bézier curve, the objects can also accelerate or decelerate along their paths.

3.4 Frame-based Event Simulation

To keep the computational overhead of our augmentation approach low, we use the same method as ShapeAug to simulate events based on frames. The first step involves generating frames with various moving objects, which were detailed

in the previous sections. We then compute the difference between consecutive frames, as DVSs register changes in lighting rather than absolute intensity values. Positive and negative differences correspond to positive and negative events, respectively.

To enhance realism, we simulate noise by randomly varying the number of events per pixel and setting some to zero. We also clip all values to 0.9 times the maximum of all non-zero values times a small random value. Additionally, pixels behind generated foreground objects are masked out to simulate occlusion.

4 EXPERIMENTS AND RESULTS

4.1 Datasets

For validation, we utilize the same datasets and settings as the original ShapeAug, which include two image-based event datasets and two original event camera datasets for classification. The converted datasets were created by recording RGB images from existing RGB datasets using an event camera. One such dataset is *DVS-CIFAR10* [14], the event-based version of CIFAR-10 [13], containing 10,000 event streams at a resolution of $128px \times 128px$. Similarly, *N-Caltech101* [17], derived from Caltech101 [7], consists of 8,709 images of varying sizes. Among the real event datasets, *N-CARS* [20] is designed for vehicle classification, containing 15,422 training samples and 8,607 test samples at a resolution of $120px \times 100px$ pixels. Another real-world dataset, *DVS-Gesture* [1], focuses on gesture recognition and includes samples of 11 hand gestures performed by 29 subjects, resulting in 1,342 samples at a resolution of $128px \times 128px$. Following the example of [18], we resize all event streams to $80px \times 80px$ pixels using bi-linear interpolation and divide them into 10 timesteps. For datasets without a predefined training-validation split, we utilize the same split as [18]. We compare our method (Shape++) against ShapeAug [2] (Shape) and the two other leading state-of-the-art event augmentation techniques: EventDrop [10] (Drop), which randomly masks out events in specific areas or time intervals, and EventMix [18] (Mix), which combines multiple samples into a single augmented sample.

4.2 Implementation

Following the approach in ShapeAug [2], we use the same training settings as [18] for all our classification experiments on DVS-CIFAR10, N-Caltech101, N-CARS, and DVS-Gesture. Specifically, we employ a preactivated spiking ResNet34 [11] with PLIF neurons [6], and the AdamW optimizer [16] with a learning rate of 1.56×10^{-4} and a weight decay of 1×10^{-4}. The model is trained using a batch size of 32 for 200 epochs with a cosine decay applied to the learning rate. For our baseline, we use geometric augmentation, which includes cropping to $80px \times 80px$ pixels after padding with 7 pixels, random horizontal flipping, and random rotation up to 15°.

Table 1: Comparison of classification results using different max shape sizes s_{max} on all four classification datasets. We report the top-1 accuracy as well as the top-5 accuracy in parantheses, except for datasets with less than 5 classes. The best and second best results are shown in **bold** and underlined respectively.

Method	s_{max} [px]	DVS-CIFAR10	N-Caltech101	N-CARS	DVS-Gesture
Geo	-	73.8 (95.5)	62.2 (81.5)	97.1	89.8 (99.6)
Geo + ShapeAug	10	74.3 (95.1)	68.0 (83.9)	97.3	90.9 (99.6)
Geo + ShapeAug	30	73.9 (94.7)	68.7 (86.9)	96.9	91.7 (100)
Geo + ShapeAug	50	**75.7** (96.7)	68.2 (85.2)	96.9	90.5 (99.2)
Geo + ShapeAug++	10	74.1 (95.8)	**72.4** (88.1)	**97.5**	90.2 (99.6)
Geo + ShapeAug++	30	75.1 (96.7)	68.3 (86.3)	97.4	91.7 (100)
Geo + ShapeAug++	50	75.4 (96.6)	70.6 (87.7)	97.4	**92.4** (100)

Table 2: Comparison of the robustness of current event augmentation approaches on various augmented versions of *DVS Gesture* [1]. The best and second best results are shown in **bold** and underlined respectively.

Train Valid	-	Geo	Drop	Shape	**Shape++**
Geo (Baseline)	89.8	87.5	58.0	63.6	59.8
Geo + Drop	89.8	87.9	**86.0**	73.9	50.8
Geo + Mix	**93.2**	**92.4**	82.6	76.9	63.6
Geo + Shape	91.7	90.5	84.1	**87.9**	56.1
Geo + **Shape++**	92.4	92.0	72.3	84.8	**85.6**

4.3 Event Data Classification

The comparison of ShapeAug++ to the original ShapeAug on multiple event classification datasets is presented in Table 1 for various maximum shape sizes s_{max}. ShapeAug++ outperforms ShapeAug on most datasets, with improvements of up to 3.7% or at least maintaining equivalent performance. The difference in performance is particularly notable on more complex datasets, such as those involving real-world DVS recordings or a large number of classes. This demonstrates that the increased complexity of ShapeAug++ is especially beneficial for handling complex input data and provides a greater challenge for the prediction network. Similar to ShapeAug, we observe that the optimal maximum shape size still depends on the specific dataset, which is reasonable given that different datasets contain objects of varying sizes.

4.4 Comparison on Robustness

We evaluate ShapeAug++ in comparison to other state-of-the-art methods using five challenging validation datasets based on the *DVS Gesture* dataset [1]. These

Table 3: Evaluation of the robustness of the combination of different event augmentation methods on various augmented versions of *DVS Gesture* [1]. The best and second best results are shown in **bold** and underlined respectively.

Train Valid	-	Geo	Drop	Shape	**Shape++**
Geo (Baseline)	89.8	87.5	58.0	63.6	59.8
Geo + Drop + Mix	92.8	89.8	89.8	75.4	64.0
Geo + Drop + Shape	91.7	90.2	88.6	87.5	52.3
Geo + Mix + Shape	94.7	91.7	87.5	**91.3**	72.3
Geo + Drop + Mix + Shape	**95.8**	**94.7**	**92.8**	89.8	71.2
Geo + Drop + **Shape++**	90.2	86.7	86.4	83.3	84.8
Geo + Mix + **Shape++**	93.9	90.2	86.4	87.9	**85.2**
Geo + Drop + Mix + **Shape++**	92.8	89.4	87.9	87.1	84.8

validation datasets were generated by applying the following augmentation techniques to each sample: Geometric transformations (horizontal flipping, rotation, cropping), EventDrop [10], ShapeAug [2], and ShapeAug++ with $s_{max} = 30$. This allows us to assess if these augmentation techniques lead to an increased robustness of the trained neural networks. The results of our experiments are shown in Table 2.

Our results indicate that ShapeAug++ significantly enhances robustness across all validation datasets compared to the baseline. It outperforms the original ShapeAug method on the original and geometrically transformed validation data. ShapeAug++ also demonstrates robustness against shape-augmented data. However, it shows a slight decrease in robustness against drop-augmentation, as its complex shapes differ from the masked-out squares used in EventDrop. The EventMix method achieves slightly higher performance than ShapeAug++ on some validation datasets. Nonetheless, these methods are not directly comparable, as EventMix introduces a multilabel classification problem.

Notably, all other augmentation methods show significantly lower robustness against validation data augmented with ShapeAug++, proving the necessity for more complex occlusion augmentation that closely resembles real-world scenarios.

In Table 3 we further explore the performance of combined augmentation strategies on the different validation datasets. The results demonstrate that incorporating EventMix into ShapeAug++ generally enhances performance across most data. Conversely, adding EventDrop reduces performance, even when combined with both EventMix and ShapeAug++. This might suggest that, due to the increased complexity of ShapeAug++, the network is too weak to handle further augmentation. Therefore, to effectively combine ShapeAug++ with other augmentation techniques, a different training schedule or a larger network may be necessary. Additionally, the results indicate that models trained without ShapeAug++ fail to perform well on ShapeAug++ augmented validation

data, showing the importance of ShapeAug++ for handling complex real-world occlusions.

5 Conclusion

Augmentation techniques play a crucial role in enhancing the robustness and accuracy of neural networks, particularly for challenging tasks. ShapeAug introduced event data augmentation by simulating occlusions through the movement of square and circular objects along linear paths. In this work, we have extended this approach with ShapeAug++, introducing more complex, randomly shaped polygons and more realistic curved movements using Bézier curves. This improvement enables ShapeAug++ to more accurately model real-world scenarios, leading to greater robustness against occlusions.

Our experiments on the most commonly used event classification datasets demonstrate that ShapeAug++ achieves significantly higher accuracy than the baseline and outperforms the original ShapeAug method across most datasets. ShapeAug++ also exhibits strong performance on challenging validation datasets, surpassing not only ShapeAug but also the EventDrop method. Moreover, no current state-of-the-art augmentation technique is capable to achieve a high robustness against ShapeAug++ augmented validation data, showing the need for a more realistic occlusion simulation during the training. Despite its complexity, which may require more capable networks and extended training schedules, ShapeAug++ often delivers exceptional performance on its own, reducing the need for combining it with other augmentation techniques.

ShapeAug++ currently applies occlusions randomly, without specifically targeting important objects in the data. Future research could explore integrating ShapeAug++ with saliency methods or other techniques to guide occlusions in a data-dependent manner, potentially enhancing its effectiveness even further.

Acknowledgments. This work was funded by the Carl Zeiss Stiftung, Germany under the Sustainable Embedded AI project (P2021-02-009).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Amir, A., Taba, B., Berg, D., Melano, T., McKinstry, J., Di Nolfo, C., Nayak, T., Andreopoulos, A., Garreau, G., Mendoza, M., Kusnitz, J., Debole, M., Esser, S., Delbruck, T., Flickner, M., Modha, D.: A low power, fully event-based gesture recognition system. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
2. Bendig, K., Schuster, R., Stricker, D.: Shapeaug: Occlusion augmentation for event camera data. In: International Conference on Pattern Recognition Applications and Methods (ICPRAM) (2024)

3. De Tournemire, P., Nitti, D., Perot, E., Migliore, D., Sironi, A.: A large scale event-based detection dataset for automotive. arXiv preprint arXiv:2001.08499 (2020)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2009)
5. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)
6. Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., Tian, Y.: Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In: International Conference on Computer Vision (ICCV) (2021)
7. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: Conference on Computer Vision and Pattern Recognition Workshop (CVPRW) (2004)
8. Fong, R., Vedaldi, A.: Occlusions for effective data augmentation in image classification. In: International Conference on Computer Vision Workshop (ICCVW) (2019)
9. Gehrig, M., Scaramuzza, D.: Recurrent vision transformers for object detection with event cameras. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2023)
10. Gu, F., Sng, W., Hu, X., Yu, F.: Eventdrop: data augmentation for event-based learning. In: International Joint Conferences on Artificial Intelligence (IIJCAI) (2021)
11. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European Conference on Computer Vision (ECCV) (2016)
12. Hoffer, E., Ben-Nun, T., Hubara, I., Giladi, N., Hoefler, T., Soudry, D.: Augment your batch: better training with larger batches. In: arXiv (2019)
13. Krizhevsky, A.: Learning multiple layers of features from tiny images. University of Toronto (2012)
14. Li, H., Liu, H., Ji, X., Li, G., Shi, L.: CIFAR10-DVS: An event-stream dataset for object classification. Frontiers in Neuroscience (2017)
15. Li, Y., Kim, Y., Park, H., Geller, T., Panda, P.: Neuromorphic data augmentation for training spiking neural networks. In: European Conference on Computer Vision (ECCV) (2022)
16. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (ICLR) (2017)
17. Orchard, G., Jayawant, A., Cohen, G.K., Thakor, N.: Converting static image datasets to spiking neuromorphic datasets using saccades. Frontiers in Neuroscience (2015)
18. Shen, G., Zhao, D., Zeng, Y.: Eventmix: An efficient data augmentation strategy for event-based learning. Information Sciences (2023)
19. Singh, K.K., Lee, Y.J.: Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In: International Conference on Computer Vision (ICCV) (2017)
20. Sironi, A., Brambilla, M., Bourdis, N., Lagorce, X., Benosman, R.: Hats: Histograms of averaged time surfaces for robust event-based object classification. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
21. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research (JMLR) (2014)

22. Sun, M., Zhang, D., Ge, Z., Wang, J., Li, J., Fang, Z., Xu, R.: Eventrpg: Event data augmentation with relevance propagation guidance. International Conference on Learning Representations (ICLR) (2024)
23. Yun, S., Han, D., Chun, S., Oh, S.J., Yoo, Y., Choe, J.: CutMix: Regularization strategy to train strong classifiers with localizable features. In: International Conference on Computer Vision (ICCV) (2019)