Audio classification with Dilated Convolution with Learnable Spacings

Ismail Khalfaoui-Hassani, Timothée Masquelier, Thomas Pellegrini

Audio classification with Dilated Convolution with Learnable Spacings

Ismail Khalfaoui-Hassani*
CerCo UMR 5549, CNRS
Université Toulouse III
Toulouse, France
ismail.khalfaoui-hassani@univ-tlse3.fr

Timothée Masquelier
CerCo UMR 5549, CNRS
Université Toulouse III
Toulouse, France
timothee.masquelier@cnrs.fr

Thomas Pellegrini
IRIT, CNRS, Toulouse INP
Université Toulouse III
Toulouse, France
thomas.pellegrini@irit.fr

Abstract

Dilated convolution with learnable spacings (DCLS) is a recent convolution method in which the positions of the kernel elements are learned throughout training by backpropagation. Its interest has recently been demonstrated in computer vision (ImageNet classification and downstream tasks). Here, we show that DCLS is also useful for audio tagging using the AudioSet classification benchmark. We took two state-of-the-art convolutional architectures using depthwise separable convolutions (DSC), ConvNeXt and ConvFormer, and a hybrid one using attention in addition, FastViT, and drop-in replaced all the DSC layers by DCLS ones. This significantly improved the mean average precision (mAP) with the three architectures without increasing the number of parameters and with only a low cost on the throughput. The method code is based on PyTorch and is available at https://github.com/K-H-Ismail/DCLS-Audio.

1 Introduction

The very popular ConvNeXt model [16], a fully convolutional model designed for vision tasks, has been successfully adapted to audio classification on AudioSet [21] by transforming audio samples to log-mel spectrograms and adapting the stem of the ConvNeXt model to fit the input audio extracts. This has improved the state of the art of audio classification using convolutional neural networks by achieving better accuracy than PANN-type models [13], while having fewer learnable parameters. Furthermore, when used as a backbone for downstream tasks, the ConvNeXt-audio model has achieved positive, if not state-of-the-art, results for the audio captioning and audio retrieval tasks.

Separately, the Dilated Convolution with Learnable Spacings (DCLS) method has already proven itself in several computer vision tasks [10]. Through a simple drop-in replacement of the model's DSC with DCLS (which can be done automatically for all layers of a model via this script 7), the DCLS convolution method has empirically proven its effectiveness for several computer vision tasks using ImagNet1k [3] trained models as backbones. This resulted in a ConvNeXt-dcls model [10] and a ConvFormer-dcls model [11], depending on the model chosen in the study, by performing the replacement in the ConvNeXt and ConvFormer models, respectively.

*Corresponding author

NeurIPS 2023 Workshop on Machine Learning for Audio

Our aim in the present article is to show empirically that a drop-in replacement of the DCLS method in the same fully convolutional models can improve their accuracy for the task of audio classification on the AudioSet dataset without much effort, demonstrating once again the interest of the method not only on the reference benchmark for image classification but also on the reference benchmark for audio classification (AudioSet [5]). Furthermore, we add a third test model that differs slightly from the other two in that it's a hybrid model (having both DSC layers and multi-head self-attention layers, depending on the stage to which the layer belongs): FastVit [26], and again, replacing the DSC layers by DCLS improves results.

This article does not claim to be the absolute state of the art on the task of classification on AudioSet, but rather tries to provide an objective comparison between known and proven convolutional models and those equipped with a DCLS convolution that would make them more efficient.

2 Related work

Audio tagging systems were mainly based on convolutional neural networks until recently, with the adaptation of vision transformers to audio processing. The PANN-based models (*e.g.*, CNN14), in particular, comprise blocks of plain 3×3 kernel convolution layers [13]. In [28], PANN-like models were enhanced, in terms of accuracy, model size and inference speed, by adding residual connections, and by modifying the kernel sizes, the stride and padding, using a "decreasing temporal size parameter". Other efficient CNN architectures, such as EfficientNet [7], were also tested in audio tagging. In [23], efficient PANNs (E-PANN) were obtained by using filter pruning. In [4], DSC layers were used, which resulted in large reductions in model complexity, together with performance gains. In [21], doing so in PANN's CNN14 also yielded significant model size reduction (about 60% relative), whilst observing a gain in performance. In this last study, ConvNeXt was adapted to perform the audio tagging task in AudioSet. It performed better or on par with the transformer-based architectures AST [6] and PaSST-S [14].

3 Methods

3.1 Dataset and configuration

Dataset. In all the experiments in this article, we used AudioSet [5], the reference dataset in audio classification. It contains about 2 million video clips downloaded from the YouTube platform. We are only interested in the audio portion of these clips and are not using the video dataset. The audio clips available in AudioSet can vary in size, but most are 10 seconds long. If a sample is longer than that, we truncate it; if shorter, we pad it with zeros. The classification task in AudioSet consists of assigning each sample to the class or classes to which it belongs among the 527 available labels. It is thus a multi-label classification task. The majority of the excerpts correspond to one of the two classes "speech" and "music" (often both), due to their predominance on the aforementioned video hosting site. This latter fact leads to an imbalance in the dataset, with several classes being poorly represented while a few classes account for most of the dataset. We downloaded the data in 2018, and some of the YouTube links have been broken since then. Our AudioSet data contains 1,921,982 clips (unbalanced train), 21,022 clips (balanced train), and 19,393 clips (evaluation).

Metrics. We report the usual evaluation metric for AudioSet tagging: mean average precision (mAP) which is typically the metric of interest in audio tagging. All DCLS-equipped models studied here outperform their respective baselines using this metric.

No weighted sampler. Given the unbalanced nature of the dataset, many state-of-the-art models make good use of a weighted random sampler [13, 8, 9], where each class in the dataset is weighted by its frequency of occurrence in the dataset. This is a classic machine learning approach to mitigate data imbalance. However, as pointed out by [19], these approaches based on a weighted sampler whose oversampling rate is adjusted as a training hyperparameter seem to overfit the dataset more than anything else and do not favor the rarest classes. Since in this article, we are only interested in the comparative study between baseline models and the same models augmented with the DCLS method, we have chosen not to include weighted samplers in our training phases, even if this means losing a few points in mAP, thereby allowing a comparison that is less noisy due to the effects of sampling. Furthermore, the naive use of Mixup augmentation [31] in conjunction with a weighted sampler may turn out to be a source of undesirable behavior, since proceeding in this way could

destroy the weighted sampling that was originally intended, as the Mixup acts randomly by drawing two samples without taking balancing into account. Weighting-aware approaches to Mixup such as [22] should be better investigated and implemented in order to better take advantage of both methods.

Spectrogram resolution. Many audio classification models use raw audio signals [20, 2, 1], while a growing number of state-of-the-art models use spectrograms, taking advantage of the signal's periodic aspect by using the Short-time Fourier transform [13, 14, 9]. We prefer this second choice in order to use computer vision baselines. Additionally, the obtained spectrograms are often filtered using the mel psychoacoustic scale [24]. We use the latter filtering to obtain mel-frequency spectrograms, which we transform from the power/amplitude scale to the decibel scale. A comprehensive enumeration of the hyperparameters used to perform these transformations is given in Table 2.

The final spectrogram size obtained is ($F = 128$, $T = 1001$). In the course of our experiments, we noticed that the larger the size of the spectrograms (in both frequency and time), the greater the mAP of the models, but to the detriment of their throughput. This is a well-known phenomenon in computer vision, where higher input image resolution often leads to better vision model accuracy but also to higher computational time costs. We believe that this resolution provides a good trade-off in mAP-throughput and argue that there is a sweet spot between resolution and stem size that offers optimal performance regarding mAP-throughput.

Adapting the stem. The three neural networks used in this study all come from the world of computer vision. It is therefore essential to adapt the stem of these models in order to process no longer natural images made up of three channels corresponding to the RGB colors, but instead a spectrogram with a single channel and a size different from the crop images initially designed for vision tasks. To this end, we used a basic stem, common to all three studied models, namely a convolution layer with a kernel size = (2, 16) and a stride = (2, 16). This stem produces maps of size (64, 62) from input spectrograms of size (128, 1001). This type of stem is similar to that originally found in the ConvNeXt model, while the ConvFormer model featured a slightly more sophisticated stem with a kernel size larger than the stride size. The FastVit model, on the other hand, came with a much more complex stem consisting of several layers based on the MobileOne block [27].

Imposing a common stem on all the models in the study means that, on the one hand, we can compare the models more accurately, knowing that the input resolution will be the same for all of them. On the other hand, in the absence of an optimal stem adapted to audio spectrograms, we use a coarse stem with which we can conduct our study. Note that the search for an ideal stem is a study in itself and that the stem presented here can always be refined and improved.

Pretraining on ImageNet1k. Using pre-trained models on ImageNet [3] as a better initialization to solve the tagging task on AudioSet is common practice [6, 14, 21]. In the first few epochs, models initialized in this way have a clear advantage over those initialized randomly. However, this advantage is quickly regained over the course of training, and randomly initialized models often end up performing similarly to or slightly worse than pre-trained ones. We only use pre-trained models on ImageNet1k when they are available and do not cost us anything to train. Therefore, we use the symbol ‡ to designate models that have not been pre-trained on ImageNet.

Configuration. In Table 2 is a comprehensive list of the hyperparameters and augmentations used in this study. These are largely similar to the hyperparameters used in [16]. Note that a high drop path (0.4) is used in this work to overcome the overfitting problem encountered with the tagging task on AudioSet and that large effective batch sizes were used (4096) to speed up training. However, some instabilities during training were noted, particularly for the ConvFormer model. These instabilities are known from [30] and were resolved by using the LAMB optimizer [29], while we used AdamW [18] for the other two models.

3.2 Models

We used three different models from computer vision that we adapted to audio inputs to corroborate our results. The first one is a fully convolutional model: ConvNeXt-tiny [16]. The second one is also a purely convolutional model that outperforms the ConvNext model on the ImageNet classification task: ConvFormer-S18 [30]. The DCLS method as a replacement for DSC has already been successfully used in the latter two models for various vision tasks, including image classification on ImageNet1k [10, 11]. The third model we used is more recent and achieves the current state-of-the-art throughput-accuracy trade-off in image classification on the ImageNet dataset: FastVit-SA24 [26]. The latter is a so-called hybrid model, i.e., it contains DSC layers as well as and multi-head self-attention layers.

3.3 DCLS substitution

Considering the baseline models discussed in the previous section, we carried out the following study: we trained the baselines on the concatenation of the unbalanced train and the balanced train sets of AudioSet, then evaluated them on the evaluation subset. We repeated the same process with the same models, except that this time we replaced all DSC layers having a kernel size equal to 7 with a DCLS convolution layer. In all test cases, we used exactly the same training configuration to avoid attributing performance gains to any reason other than the replacement of the DSC layers by DCLS ones. Also, to learn the positions (and standard deviations for DCLS-Gauss) of each kernel element, we followed the same training techniques as those listed in [11]. This gave us 6 test cases to examine in total, for which we measure the mAP metric mentioned in Section 3.1 averaged over three different seeds (seeds 0, 1, and 2).

4 Results

The results presented in Table 1 demonstrate the performance of the three models mentioned in Section 3.2 on the 128×1001 spectrograms, where the convolution method used varies. Notably, we observe that when comparing each baseline model with its DCLS-equipped counterpart, the use of DCLS-Gauss with a kernel size of 23^2 and a kernel count of 26 stands out, achieving a higher mAP (+0.6 on average) with an equal or lower number of parameters. This result highlights the effectiveness of DCLS-Gauss in enhancing classification performance. DCLS does, however, introduce a reduction in throughput (13% for ConvNeXt-T and FastVit-SA24 and 23% for ConvFormer-S18) due to the use of larger kernels. The results of a previous study on ConvNeXt [21] show that an mAP of 47.1 can be achieved, but here we only reach 44.8 for the baseline; this is due to the fact that in that previous study, a higher spectrogram resolution was used (224×1001 versus 128×1001 in this work) and that a stem size of 4×4 instead of 2×16 here was used to produce larger feature maps, which is reflected both in the large memory required to run this model and in the model's throughput.

model	ker. size / count	method	# param.	mAP	throughput (sample / s)
CNN14 [13]		Conv.	80.7M	43.1	378.2
PaSST-S [14]		MHS. Attention.	87M	47.1	88.7
ConvNeXt-T [21]	7^2 / 49	Depth. Conv.	28.2M	47.1	153.6
ConvFormer-S18†	7^2 / 49	Depth. Conv.	26.8M	43.14 ± 0.03	513.3
ConvFormer-S18†	23^2 / 26	DCLS-Gauss	26.8M	43.68 ± 0.02	396.8
FastVIT-SA24‡	7^2 / 49	Depth. Conv.	**21.5**M	43.82 ± 0.05	**633.6**
FastVIT-SA24‡	23^2 / 26	DCLS-Gauss	**21.5**M	44.4 ± 0.07	551.7
ConvNeXt-T	7^2 / 49	Depth. Conv.	28.6M	44.83 ± 0.14	591.4
ConvNeXt-T	23^2 / 26	DCLS-Gauss	28.6M	**45.52** ± 0.05	509.4

Table 1: **Classification mean average precision (mAP) on the evaluation set of AudioSet.** For the baselines using DSC and the DCLS-Gaussian cases, the results have been averaged over 3 distinct seeds and presented in the format mean $\pm$ standard deviation. † : trained using LAMB, ‡ : no ImageNet pretraining. The throughputs were calculated with a single NIVIDIA V100 32-GB gpu.

5 Conclusion

In conclusion, this article has demonstrated the efficacy of Dilated Convolution with Learnable Spacings (DCLS) as a method with promising applications beyond the computer vision field. By exploiting DCLS in the audio tagging task on AudioSet, we have demonstrated tangible improvements in accuracy when compared to models employing traditional DSC methods. While this work does not claim to establish an absolute state-of-the-art benchmark, it does contribute valuable insights into the potential of DCLS convolution in audio classification. This research underscores the significance of exploring novel convolutional techniques, like DCLS, and adapting them to various domains beyond their initial design. As the field of deep learning continues to evolve, such methods pave the way for broader and more efficient applications, thereby advancing the state of the art in deep learning.

Acknowledgments

This work was performed using HPC resources from GENCI–IDRIS (Grant 2023-[AD011013219R1]). Support from the ANR-3IA Artificial and Natural Intelligence Toulouse Institute is gratefully acknowledged. We would also like to thank the region of Toulouse Occitanie.

References

[1] H. Akbari, L. Yuan, R. Qian, W.-H. Chuang, S.-F. Chang, Y. Cui, and B. Gong. Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text. *Advances in Neural Information Processing Systems*, 34:24206–24221, 2021.

[2] W. Dai, C. Dai, S. Qu, J. Li, and S. Das. Very deep convolutional neural networks for raw waveforms. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 421–425. IEEE, 2017.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 248–255. IEEE, 2009.

[4] K. Drossos, S. I. Mimilakis, S. Gharib, Y. Li, and T. Virtanen. Sound event detection with depthwise separable and dilated convolutions. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2020.

[5] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *Proc. IEEE ICASSP*, New Orleans, LA, 2017.

[6] Y. Gong, Y.-A. Chung, and J. Glass. AST: Audio Spectrogram Transformer. In *Proc. Interspeech*, pages 571–575, Brno, 2021.

[7] Y. Gong, Y.-A. Chung, and J. Glass. PSLA: Improving audio tagging with pretraining, sampling, labeling, and aggregation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3292–3306, 2021.

[8] Y. Gong, A. Rouditchenko, A. H. Liu, D. Harwath, L. Karlinsky, H. Kuehne, and J. R. Glass. Contrastive audio-visual masked autoencoder. In *The Eleventh International Conference on Learning Representations*, 2022.

[9] P.-Y. Huang, V. Sharma, H. Xu, C. Ryali, H. Fan, Y. Li, S.-W. Li, G. Ghosh, J. Malik, and C. Feichtenhofer. Mavil: Masked audio-video learners. *arXiv preprint arXiv:2212.08071*, 2022.

[10] I. Khalfaoui-Hassani, T. Pellegrini, and T. Masquelier. Dilated convolution with learnable spacings. In *The Eleventh International Conference on Learning Representations*, 2023.

[11] I. Khalfaoui-Hassani, T. Pellegrini, and T. Masquelier. Dilated convolution with learnable spacings: beyond bilinear interpolation. In *ICML 2023 Workshop on Differentiable Almost Everything: Differentiable Relaxations, Algorithms, Operators, and Simulators*, 2023.

[12] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur. Audio augmentation for speech recognition. In *Proc. Interspeech 2015*, pages 3586–3589, 2015.

[13] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

[14] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer. Efficient training of audio transformers with patchout. In *Proc. Interspeech*, pages 2753–2757, Incheon, 2022.

[15] G. Larsson, M. Maire, and G. Shakhnarovich. Fractalnet: Ultra-deep neural networks without residuals. In *International Conference on Learning Representations*, 2016.

[16] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 11976–11986, 2022.

[17] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2016.

[18] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

[19] R. C. Moore, D. P. Ellis, E. Fonseca, S. Hershey, A. Jansen, and M. Plakal. Dataset balancing can hurt model performance. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[20] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[21] T. Pellegrini, I. Khalfaoui-Hassani, E. Labbé, and T. Masquelier. Adapting a ConvNeXt Model to Audio Classification on AudioSet. In *Proc. INTERSPEECH 2023*, pages 4169–4173, 2023.

[22] S. Ramasubramanian, H. Rangwani, S. Takemori, K. Samanta, Y. Umeda, and R. Venkatesh Babu. Selmix: Selective mixup fine tuning for optimizing non-decomposable metrics. In *The Differentiable Almost Everything Workshop of the 40 th International Conference on Machine Learning*, 2023.

[23] A. Singh, H. Liu, and M. D. Plumbley. E-PANNs: Sound Recognition Using Efficient Pretrained Audio Neural Networks. In *Proc. Inter Noise*, Chiba, 2023.

[24] S. S. Stevens, J. Volkmann, and E. B. Newman. A scale for the measurement of the psychological magnitude pitch. *The journal of the acoustical society of america*, 8(3):185–190, 1937.

[25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[26] P. K. A. Vasu, J. Gabriel, J. Zhu, O. Tuzel, and A. Ranjan. Fastvit: A fast hybrid vision transformer using structural reparameterization. *arXiv preprint arXiv:2303.14189*, 2023.

[27] P. K. A. Vasu, J. Gabriel, J. Zhu, O. Tuzel, and A. Ranjan. Mobileone: An improved one millisecond mobile backbone. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7907–7917, 2023.

[28] S. Verbitskiy, V. Berikov, and V. Vyshegorodtsev. Eranns: Efficient residual audio neural networks for audio pattern recognition. *Pattern Recognition Letters*, 161:38–44, 2022.

[29] Y. You, J. Li, S. Reddi, J. Hseu, S. Kumar, S. Bhojanapalli, X. Song, J. Demmel, K. Keutzer, and C.-J. Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

[30] W. Yu, C. Si, P. Zhou, M. Luo, Y. Zhou, J. Feng, S. Yan, and X. Wang. Metaformer baselines for vision. *arXiv preprint arXiv:2210.13452*, 2022.

[31] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[32] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020.

6 Appendix A: Training hyper-parameters

Configuration	AudioSet2M
Optimizer	AdamW [18] | LAMB [29]†
Optimizer momentum	$\beta_1 = 0.9, \beta_2 = 0.999$
Weight decay	0.05
Base learning rate	$4e-3$
Learning rate schedule	half-cycle cosine decay [17]
Gradient clipping	None
Epochs	60
Warm-up epochs	20
Batch size	4096
GPUs size	32
Weighted sampling	False
Drop path [15]	0.4
Mixup [31]	0.8
Multilabel	True
Label smoothing [25]	0.1
Loss Function	Binary Cross-Entropy
Dataset Mean for Normalization	-18.2696
Dataset Std for Normalization	30.5735
Spectrogram configuration	
Number of fft	1024
Hop length	320
Power	2
Mel scale configuration	
Number of mels	128
Sample rate	32 000
Min frequency	50
Max frequency	14 000
Amplitude to dB	True
Augmentations	
PadOrTruncate	$10 \times$ sample rate
RandomRoll	$[-$sample size, sample size$]$, p $= 1$
SpeedPerturbation [12]	rates $= (0.5, 1.5)$, p $= 0.5$
RandomErasing [32]	p $= 0.25$

Table 2: **Training hyper-parameters.**

7 Appendix B: How to replace all model's DSC by DCLS ones ?

```python
import copy
from torch import nn
from DCLS.construct.modules import Dcls2d

# Helper function that replaces all ".int." patterns
# by "[int]" in a character string
def replace_dots_brackets(name):
    name_split = name.split(".")
    name_split = [
        "[" + i + "]" if i.isdigit() else "." + i for i in name_split
    ]
    return "".join(name_split[:-1]), name_split[-1][1:]

# Helper function that replaces all the
# 2D depthwise separable convolution in
# a model by synchronized Dcls2d ones
def replace_depthwise_dcls(
    model, dilated_kernel_size=23, kernel_count=26, version="gauss"):
    in_channels, P, SIG = 0, None, None
    # Loop over all model modules
    for name, module in model.named_modules():
        # if the module is a depthwise separable Conv2d module
        if (isinstance(module, nn.Conv2d)
            and module.groups == module.in_channels == module.
    out_channels
            and module.kernel_size == (7, 7)
        ):
            name_eval, last_layer = replace_dots_brackets(name)
            dcls_conv = Dcls2d(
                module.in_channels,
                module.out_channels,
                kernel_count=kernel_count,
                stride=module.stride,
                dilated_kernel_size=dilated_kernel_size,
                padding=dilated_kernel_size // 2,
                groups=module.in_channels,
                version=version,
                bias=module.bias is not None,
            )
            nn.init.normal_(dcls_conv.weight, std=0.02)
            if module.bias is not None:
                nn.init.constant_(dcls_conv.bias, 0)

            # Synchronise positions and standard
            # deviations belonging to the same stage
            if in_channels < module.in_channels:
                in_channels = module.in_channels
                P, SIG = dcls_conv.P, dcls_conv.SIG

            dcls_conv.P, dcls_conv.SIG = P, SIG
            setattr(eval("model" + name_eval), last_layer, dcls_conv)
    return model

model = nn.Conv2d(96, 96, 7, padding=3, groups=96)
# Replace all the 2D depthwise separable convolutions
# in the model by synchronized Dcls2d ones.
model = replace_depthwise_dcls(
    copy.deepcopy(model),
    dilated_kernel_size=23,
    kernel_count=26,
    version="gauss",
)
print(model)
```