Avoiding Overfitting: new algorithms to improve generalization in Convolutional Neural Networks

Claudio F. G. Santos
Department of Computer Science
Federal University of São Carlos - UFSCar
São Carlos - SP - Brazil
E-mail: cfsantos@ufscar.br
claudio.santos@eldorado.org.br

João Paulo Papa
Department of Computing
State University of Sao Paulo - UNESP
Bauru - SP - Brazil
E-mail: joao.papa@unesp.br

***Abstract*—Deep Learning has achieved state-of-the-art results in several domains, such as im- age processing, natural language processing, and audio processing. To accomplish such results, it uses neural networks with several processing layers along with a massive amount of labeled information. One particular family of Deep Learning is the Convolutional Neural Networks (CNNs), which work using convolutional layers de- rived from the digital signal processing area, being very helpful to detect relevant features in unstructured data, such as audio and pictures. One way to improve results on CNN is to use regularization algorithms, which aim to make the training process harder but generate models that generalize better for inference when used in applications. The present work contributes to the area of regularization methods for CNNs, proposing more methods for use in different image processing tasks. This thesis presents a collection of works developed by the author during the research period, which were published or submitted until the present time, presenting: (i) a survey, listing recent regularization works and highlighting the solutions and problems of the area; (ii) a neuron dropping method to use in the tensors generated during CNNs training; (iii) a variation of the mentioned method, changing the dropping rules, targeting different features of the tensor; and (iv) a label regularization algorithm used in different image processing problems.**[1]

I. Introduction

Machine Learning is a field of study that aims to generate computer software capable of learning some tasks based on data examples. Supervised Learning is one of the methods in this family, which creates models that rely on labeled data to learn to execute some jobs. Deep Learning can use this learning method to generate really powerful models that are state-of-the-art in several tasks related to natural language processing (NLP) [1], [2] and image processing [3], [4]. To achieve such nice results, the process of training depends on several points, such as model architecture, amount and quality of training data, optimization methods, and others.

One family of algorithms can improve these results by applying changes to different parts of the training process. These algorithms are called regularization, which aims to generate difficulties in the training procedure but creates models which perform better in execution time [5]. This study targeted generating new regularization algorithms to improve the results of Convolutional Neural Networks.

[1]This work relates to Claudio F. G. Santos' Ph.D. thesis.

A. Hypothesis

The hypothesis and contributions of the present thesis regard answering the following questions: a) could the Dropout logic be changed, i.e., instead of randomly dropping neurons, using another dropping police, improve neural networks results? And b) could a random change in the labels generate neural networks with better results? Three new algorithms are proposed to answer the questions. i) a new logic to drop neurons during training time, based on the maximum output tensor values, ii) an adaption of the previous method that considers the tensor structure for convolutional neural networks, and iii) a label level regularization. The results shown in the following sections support the hypothesis. The thesis is formed from a collection of works published and submitted by the authors during the research period.

B. Scope of this work

This work is segmented as follows: Section II discusses the evolution of CNNs in the last years and presents a list of regularization algorithms studied. Section III illustrates the algorithms developed during the period of the presented research. Section IV shows the results of the developed methods, compared with other studies. Finally, Section V, presents the conclusion, including the list of the articles developed during the research period.

II. Related Works

This section presents a brief historical analysis of Convolutional Neural Networks, the methodology used for defining the most relevant works in the area, and the list of these selected works we considered important in this field.

A. Convolutional Neural Networks

Convolutional neural networks (CNNs) was conceived by adapting convolutional processing from the digital signal process area. Once the backpropagation algorithm [6] was successfully adapted to fit the kernel values, it proved it

could be used for image classification, more specifically for classifying digits from the MNIST dataset [7] by the LeNet-5 architecture [8]. Another historically relevant work is the AlexNet architecture [9] developed to classify images from the ImageNet competition [10], achieving the first place in this contest, overcoming the runner up by more than 10%. Ths work is important because it was one of the first ones to show the CNNs could be used in pratical solutions.

Nowadays, CNNs are considered state-of-the-art in several different image processing related problems, such as image classification [3], image super-resolution [11], and object detection [12]. One may argue that Transformers [13] has overcome CNNs, however, it is possible to say that both method are in draw.

B. Regularization for Convolutional Neural Networks

A serious analysis of recent regularization works were performed to identify the main characteristics of the area. The works were selected according to some rules:

- the work should be recent, not older than 4 years, with the exception of the Dropout;
- the source code should be available in some way, preferably in a public repository, and;
- the regularization method should improve the baseline results.

For a better understanding of the methods, a classification method system is proposed, differentiating each method in the way they work during the training of a given neural network:

- **Input**: the method changes the data in the input of the neural network. In most of the cases it can be seen as a data augmentation technique;
- **Internal**: the algorithm changes information among layers of the neural network during training;
- **Label**: the procedure changes information in the label during training.

Although this categorization of methods is proposed, some methods can act change information in more than one place during training (ex: changing input and label information together).

Table I enumerates all methods studied fot this research.

Ref.	Short Name	Description	Where
[14]	Bag of Tricks	Combines several regularizers to show how it improves CNN	Input
[15]	Batch Augment	Increases the size of the mini-batch	Input
[16]	FixRes	Performs train and test with different image sizes	Input
[17]	Cutout	Removes part of the image	Input
[18]	CutMix	Replaces part of the image using other parts of other images	Input / Label
[19]	RandomErasing	Replaces part of the image by noise or paint the region	Input
[20]	Mixup	Mixes two images from different classes	Input / Label
[21]	AutoAugment	Learns how to provide better data augmentation based on information from the training data set	Input
[22]	Fast Autoaugment	Reduces the training time of the agent from [21]	Input
[23]	RandAugment	Learns augmentation policies during training	Input
[24]	PBA	Population Based algorithm for data augmentation	Input
[25]	CutBlur	Replaces regions from high-resolution images with low resolution pieces	Input / Label
[26]	Dropout	Drops random neurons	Internal
[28]	GradAug	Trains sub-networks from the original CNN	Internal
[29]	Local Drop	Dropout and DropBlock based on the Radamacher complexity	Internal
[30]	Shake-Shake	Gives different weights to each branch of the residual connection	Internal
[31]	ShakeDrop	Improves Shake-Shake by generalizing to other models	Internal
[32]	Manifold Mixup	Act like Mixup, however, in the middle layers of a CNN	Internal / Label
[33]	DropBlock	Drops entire regions from a tensor	Internal
[34]	AutoDrop	Learns drop pattern	Internal
[35]	Label Smoothing	Replaces one-hot encoding vectors to smoothed labels	Label
[36]	TSLA	Two-stage algorithm for label smoothing	Label
[37]	SLS	Quantifies label smoothing based on feature space	Label
[38]	JoCoR	Co-relates labels for label smoothing	Label

TABLE I: Summarization of the approaches considered in this research.

III. Proposed Algorithms

As a result of this research, three new algorithms were proposed. This section describes more details about the implementation, basic concepts and more details about each one.

A. MaxDropout

The first proposed algorithm during the research is the MaxDropout [27]. While Dropout [26] randomly drops neurons during training, MaxDropout uses neuron activation value to define if a given neuron will be dropped or not. In training time, MaxDropout removes all neurons that has a value higher than a threshold. Algorithm 1 shows how it works.

After training, neural networks work by generating features maps based on the most active (with higher values) neurons. During training, MaxDropout forces the neural architecture to learn without the most activate neurons, generating models that learns from features that are not necessarily generated by the most activated neurons.

B. MaxDropout V2

An improvement on MaxDropout, called MaxDropout V2 [39] is proposed in the next work. It kept the concept of dropping neurons based on their values, however, it relies on a space correlation between the neurons.

Instead of performing a direct comparison between neuron values, it first calculate the sum of the values of a given tensor

Algorithm 1 Pseudocode for MaxDropout training algorithm.

```
while training do
    for each layer do
        rate ← U(0, r)
        normTensor ← L2Normalize(Tensor)
        max ← Max(normTensor)
        keptIdx ← IdxOf(normTensor, (1 − rate) ∗ max)
        returnTensor ← Tensor ∗ KeptIdx
    end for
end while=0
```

on the axis 1, generating a matrix of values. Based on this new matrix, it generates a dropping mask by detecting the most activate (with higher values) positions on this matrix, which indicates the most relevant feature maps, and then removes the neurons that belongs to this segment. Algorithm 2 shows how it works.

Algorithm 2 Pseudocode for MaxDropout V2 training algorithm.

```
while training do
    for each layer do
        rate ← U(0, r)
        matrix_tensor ← sum_axis1(Tensor)
        normMatrix ← L2Normalize(matrix_tensor)
        max ← Max(normMatrix)
        keptIdx ← IdxOf(normMatrix, (1 − rate) ∗ max)
        keptIdx ← keptIdx.expand()
        returnTensor ← Tensor ∗ KeptIdx
    end for
end while=0
```

Results comparing MaxDropout and MaxDropout V2 shows their performance on image classification are roughly the same, as it is possible to see in Section IV, however, MaxDropout V2 is faster for training. It took around 10% less time for training, as it is possible to see in Table II.

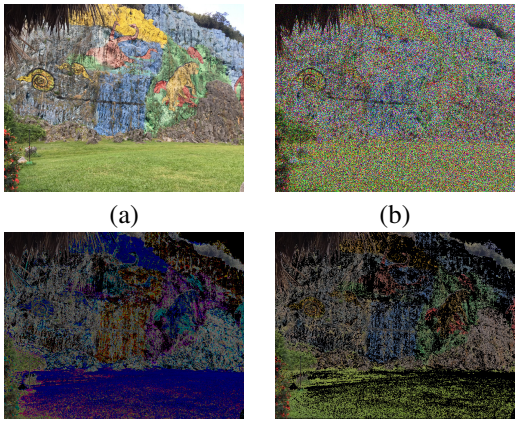
TABLE II: Time consumed in seconds for training ResNet-18 in CIFAR-100 dataset.

	Seconds per Epoch	Total time
MaxDropout [27]	33.1	6,621
MaxDropoutV2 [39]	**30.2**	**6,038**

For better understanding, Figure 1 demonstrates, in an RGB image, a simulation of what each algorithm executes during training. In this sequence of images, the first image can be pretended as a 3D tensor, and the subsequent images are the execution of the Dropout, MaxDropout and MaxDropout V2.

C. Random Label Smoothing - RLS

Regularization at the label level may even sound something unsafe because for tasks that strongly rely on supervised training such as image classification, the place one may argue not to change is the information on the label level. However, some studies [25], [35] demonstrates that it is a place that can be used to perform modifications and still improve the results of machine learning systems.



(a) (b) (c) (d)

Fig. 1: Simulation using colored images. The original colored image is presented in (a), and its outcomes are bestowed after (b) Dropout, (c) MaxDropout, and (d) MaxDropoutV2 transformations using a dropout rate of 50%.

One of the methods proposed by this thesis is Random Label Smoothing, which is known as RLS. It works by randomly changing the label information among some thresholds, depending on the problem the neural network is trying to solve.

The first problem studied is image classification. For this case, RLS changes the value of the label during training by setting the active label with higher values and allowing the other places of the vector to have any value smaller than 0.5. Figure 2 illustrates how RLS works for image classification.

For image processing-related tasks that aim to generate an RGB image as a label, RLS works by randomly changing the value of the pixels in the same amount. First, it was proposed to unsystematically change the values of the label without any restricted rules, however, as it is possible to see in our results (tagged as RLS-gaussian because the random distribution used was the Gaussian distribution) it did not perform well, decreasing the results of the baseline. One calculation that worked well was to change all the pixel values in the same amount. For instance, if the value chosen is 10, all pixels of the image should receive 10 more in the final value. The fundamental problem here is to set the boundaries that the label can be safely changed to improve the results of the neural network.

The first method for setting up the boundaries that work was to reverse engineering the values of the output of the existing neural networks, based on the Peak Signal-to-noise Ration (PSNR) metric. The PSNR can be directly calculated based on the mean square error (MSE) of the output, i.e., based on the PSNR values, one can set the average pixel value error of an output. By reversing the PSNR error into MSE value for EDSR [40] and PyNet [41], we show that the CNNs training using RLS relying on this approach could improve the final results.

The second approach that worked well for the RGB reconstruction-related problems is an adaptation of the pre-

viously mentioned method. By using half of the values of the boundaries, we could slightly improve results. In Section IV, the results related to this method are listed as RLS-half.

To show how innovative this process is, as far as it was researched, this is the first regularization method that worked well for the Image Signal Processing (ISP) problem, also known as Software ISP.

IV. Results

This section presents the results of the developed methods and compares other methods of the literature. First, it shows the results of the image classification task. After that, a comparison of the RLS against other regularization techniques is applied to the image super-resolution task. Finally, it presents the results for the software ISP task.

A. Image Classification

All methods developed in this work were tested on the image classification task. Table III shows their results and compares them with other methods relevant to this research.

The most important result is the RLS method applied to the ResNet-18 architecture [42]. When working by itself, the average result is beaten just by the Mixup regularization [20], however, it is important to remember that Mixup works in two parts: by mixing the input and the values on the label. One may argue that it might be necessary to compare Mixup with RLS if the RLS used some other regularization along in the training, so it might be fairer if the Mixup results are compared to the outcomes of the RLS + Cutout training, which in this case changes values in the input and the label. In this case, it can be considered that we achieved the best result for ResNet-18 applied to the CIFAR-100 data set.

B. Image Super-Resolution

RLS was evaluated for the image super-resolution task too. Table IV shows its results. For the Div2K data set, the EDSR architecture using RLS-Half achieved the best result as far as we could check in the literature, overcoming Cutblur [25] and all other regularization techniques working together, showing it could be a reliable alternative to improve results in this particular task.

For the RealSR data set, RLS-Half has the runner-up result, just being worse than the training with all regularization working along, however, when directly compared to each method, it reaches the best result.

C. Software ISP

Last but not least, Table V shows the results of RLS for the software ISP task. The most relevant information here is that, as far as we searched, RLS is the first regularization method to work in this particular job, that is why there is no comparison with other techniques, only against the baseline PyNet [41] architecture.

Method	CIFAR-10	CIFAR-100
ResNet18 [17]	4.72 ± 0.21	22.46 ± 0.31
+ Cutout [17]	3.99 ± 0.13	21.96 ± 0.24
+ RandomErasing [19]	4.31 ± 0.07	24.03 ± 0.19
+ Mixup [20]	4.2	21.1
+ MM [32]	2.95 ± 0.04	20.34 ± 0.52
+ TSLA [36]	-	21.45 ± 0.28
+ TargetDrop [43]	4.41	21.37
+ TargetDrop + Cutout [43]	3.67	21.25
+ LocalDrop [29]	4.3	22.2
+ MaxDropout [27]	4.66 ± 0.14	21.93 ± 0.07
+ MaxDropoutV2 [39]	4.63 ± 0.03	21.92 ± 0.23
+ MaxDropout + Cutout [27]	3.76 ± 0.08	21.82 ± 0.13
+ RLS	-	21.18 ± 0.35
+ RLS + Cutout [17]	-	20.6 ± 0.16
WRN [44]	4.00	19.25
+ Dropout [44]	3.89	18.85
+ TargetDrop [43]	3.68	-
+ GradAug [28]		16.02
+ Dropout + Cutout [17]	3.08 ± 0.16	18.41 ± 0.27
+ Dropout + PBA [24]	2.58 ± 0.06	16.73 ± 0.15
+ Dropout + RE [19]	3.08 ± 0.05	17.73 ± 0.15
+ Dropout + BA + Cutout [15]	2.85	19.87
+ ShakeDrop [31]	4.37	19.47
+ Dropout + RE [19]	3.08 ± 0.05	17.73 ± 0.15
+ Dropout + Mixup [20]	2.7	17.5
+ Dropout + MM [32]	2.55 ± 0.02	18.04 ± 0.17
+ Dropout + Fast AA [22]	2.7	17.3
+ Dropout + RA [23]	2.7	16.7
+ AutoDrop [34]	3.1	
+ AutoDrop + RE [34]	2.1	-
+ MaxDropout [27]	3.84	18.81
PyramidNet [45]	3.48 ± 0.20	17.01 ± 0.39
+ GradAug [28]		13.76
+ ShaekDrop [31]	3.08	16.22
+ ShaekDrop + Cutout [21]	2.3	12.2
+ ShaekDrop + AA [21]	1.5 ± 0.1	10.7 ± 0.2
+ ShaekDrop + Fast AA [22]	1.8	11.9
+ ShaekDrop + RA [23]	1.5	-
+ ShaekDrop + PBA [24]	1.46 ± 0.07	
+ ShaekDrop [31] + RLS	-	16.12 ± 0.14

TABLE III: Error (in %) for each classification dataset using different methods and models. The following acronyms were used: MM = ManifoldMixup, PBA = Population Based Augmentation, RE = RandomErasing, BA = BatchAugmentation, AA = AutoAugment, Fast AA = Fast AutoAugment, and RA = RandAugment.

	Div2K	RealSR
EDSR	29.21	28.89
Cutout	29.22 ± 0.01	28.95 ± 0.06
Cutmix	29.22 ± 0.01	28.89 ± 0.00
Mixup	29.26 ± 0.05	28.98 ± 0.09
CutMixup	29.27 ± 0.06	29.03 ± 0.14
RGB Permutation	29.30 ± 0.09	29.02 ± 0.13
Blend	29.23 ± 0.02	29.03 ± 0.14
Cutblur	29.26 ± 0.05	29.12 ± 0.23
All	29.30 ± 0.30	$\mathbf{29.16 \pm 0.27}$
RLS-Gaussian	28.03 ± 0.07	27.97 ± 0.04
RLS-Full	29.31 ± 0.01	29.05 ± 0.04
RLS-Half	$\mathbf{29.32 \pm 0.01}$	29.15 ± 0.03

TABLE IV: PSNR results on Div2K and RealSR datasets for EDSR using regulization methods.

	Batch - Regular Training			Batch - Label Smoothing			Batch - Random Label Smoothing		
Class 0	0	0	0	0.022	0.022	0.022	0.1	0.05	0.02
Class 1	0	0	0	0.022	0.022	0.022	0.05	0	0.15
Class 2	0	0	0	0.022	0.022	0.022	0.03	0.03	0.05
Class 3	0	**1**	0	0.022	**0.8**	0.022	0.17	**0.75**	0.01
Class 4	0	0	0	0.022	0.022	0.022	0.02	0.02	0
Class 5	**1**	0	0	**0.8**	0.022	0.022	**0.5**	0.06	0.03
Class 6	0	0	0	0.022	0.022	0.022	0.03	0.01	0.01
Class 7	0	0	0	0.022	0.022	0.022	0.04	0.03	0.03
Class 8	0	0	**1**	0.022	0.022	**0.8**	0.01	0.04	**0.68**
Class 9	0	0	0	0.022	0.022	0.022	0.05	0.01	0.02

Fig. 2: Simulation of Label Smoothing and Random Label Smoothing over a batch of labels during training for a classification model (active label in bold). Traditionally, the active label is set to"1" while all other classes' indices are set to '0". In Label Smoothing, the active class is set to a constant (and higher) value, while the inactive classes are set to a smaller invariant value. In Random Label Smoothing, the active label receives a random (and higher) value (e.g., greater than 0.5) while the inactive labels receive a random value that, summed with the active label value, reaches 1.

Method	PSNR	MS-SSIM
PyNet [41]	21.19	0.862
Pynet + Gaussian-RLS	20.86	0.850
Pynet + Full-RLS	21.21	0.863
Pynet + Half-RLS	$\mathbf{21.22 \pm 0.01}$	$\mathbf{0.867}$

TABLE V: Results on Zurich RAW to RGB Dataset for PyNet.

V. Conclusion

This sections presents the most pertinent works of this research, emphasizing the three regularization methods developed which had relevant results in the area: MaxDropout, MaxDropout V2 and RLS. Other significant result is the related works section which has been published as a survey, displaying the recent historical development on the regularization algorithms and other aspects, such as the definition of an essential evaluation protocol and some other areas that does not have tailored made regulatization to help improving the models.

A. Publications related to the thesis

As a result of this research, one survey [46] and two conference papers [27], [39] were published, and one other paper is under evaluation in a journal. Table VI shows these papers.

B. Other publications as first author

The first author of this work has also worked in other research areas, such as biopsy cancer classification [47] and biometry identification by gait recognition [48], [49]. Table VII shows other papers the author proposed, however, they are not related directly to the main problem of the thesis.

C. Other publications as co-author

Finally, a list of works that the author has contributed as a co-author of the research. Table VIII shows these works.

Name	Type	Year	Status
MaxDropout: Deep Neural Network Regularization Based on Maximum Output Values [27]	Conference	2021	Published
MaxDropoutV2: An Improved Method to Drop Out Neurons in Convolutional Neural Networks [39]	Conference	2022	Published
Avoiding Overfitting: A Survey on Regularization Methods for Convolutional Neural Networks [46]	Journal	2022	Published
Rethinking Regularization with Random Label Smoothing	Journal	2022	Submited

TABLE VI: List of publications developed by the author related to the thesis.

Name	Type	Year	Status
Does pooling really matter? an evaluation on gait recognition [49]	Conference	2019	Published
BreastNet: breast cancer categorization using convolutional neural networks [47]	Conference	2020	Published
Does Removing Pooling Layers from Convolutional Neural Networks Improve Results? [50]	Journal	2020	Published
Normalizing images is good to improve computer-assisted COVID-19 diagnosis [51]	Book Chapter	2021	Published
Gait Recognition Based on Deep Learning: A Survey [48]	Journal	2022	Published
Rethinking Regularization with Random Label Smoothing	Journal	2022	Submited

TABLE VII: List of publications developed by the author.

Acknowledgment

This work has been supported by the following Brazilian research agencies: Eldorado Research Institute, FAPESP (grants 2013/07375-0, 2014/12236-1, and 2019/07665-4), CNPq (grant 308529/2021-9), and Petrobras (grant 2017/00285-6)

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv*

Name	Type	Year	Status
A hybrid approach for breast mass categorization [52]	Conference	2019	Published
Image Denoising using Attention-Residual Convolutional Neural Networks [53]	Conference	2020	Published
Improving Pre-Trained Weights through Meta-Heuristics Fine-Tuning [54]	Conference	2021	Published
ISP meets Deep Learning: A Survey on Deep Learning Methods for Image Signal Processing	Journal	2022	Submitted

TABLE VIII: List of publications developed by the author.

preprint arXiv:1810.04805, 2018.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[3] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint arXiv:1905.11946*, 2019.

[4] A. Sauer, K. Schwarz, and A. Geiger, "Stylegan-xl: Scaling stylegan to large diverse datasets," *arXiv preprint arXiv:2202.00273*, vol. 1, 2022.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[6] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[7] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[11] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2018, pp. 0–0.

[12] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 781–10 790.

[13] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, 2020, pp. 38–45.

[14] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of tricks for image classification with convolutional neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 558–567.

[15] E. Hoffer, T. Ben-Nun, I. Hubara, N. Giladi, T. Hoefler, and D. Soudry, "Augment your batch: better training with larger batches," *arXiv preprint arXiv:1901.09335*, 2019.

[16] H. Touvron, A. Vedaldi, M. Douze, and H. Jégou, "Fixing the train-test resolution discrepancy," *arXiv preprint arXiv:1906.06423*, 2019.

[17] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[18] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6023–6032.

[19] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation." in *AAAI*, 2020, pp. 13 001–13 008.

[20] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[21] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation policies from data," *arXiv preprint arXiv:1805.09501*, 2018.

[22] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, "Fast autoaugment," in *Advances in Neural Information Processing Systems*, 2019, pp. 6665–6675.

[23] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 702–703.

[24] D. Ho, E. Liang, X. Chen, I. Stoica, and P. Abbeel, "Population based augmentation: Efficient learning of augmentation policy schedules," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2731–2741.

[25] J. Yoo, N. Ahn, and K.-A. Sohn, "Rethinking data augmentation for image super-resolution: A comprehensive analysis and a new strategy," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8375–8384.

[26] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[27] C. F. G. d. Santos, D. Colombo, M. Roder, and J. P. Papa, "Maxdropout: Deep neural network regularization based on maximum output values," in *Proceedings of 25th International Conference on Pattern Recognition, ICPR 2020, Milan, Italy, 10-15 January, 2021*. IEEE Computer Society, 2020, pp. 2671–2676.

[28] T. Yang, S. Zhu, and C. Chen, "Gradaug: A new regularization method for deep neural networks," *arXiv preprint arXiv:2006.07989*, 2020.

[29] Z. Lu, C. Xu, B. Du, T. Ishida, L. Zhang, and M. Sugiyama, "Localdrop: A hybrid regularization for deep neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[30] X. Gastaldi, "Shake-shake regularization," *arXiv preprint arXiv:1705.07485*, 2017.

[31] Y. Yamada, M. Iwamura, T. Akiba, and K. Kise, "Shakedrop regularization for deep residual learning," *IEEE Access*, vol. 7, pp. 186 126–186 136, 2019.

[32] V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, D. Lopez-Paz, and Y. Bengio, "Manifold mixup: Better representations by interpolating hidden states," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6438–6447.

[33] G. Ghiasi, T.-Y. Lin, and Q. V. Le, "Dropblock: A regularization method for convolutional networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 727–10 737.

[34] H. Pham and Q. V. Le, "Autodropout: Learning dropout patterns to regularize deep networks," *arXiv preprint arXiv:2101.01761*, 2021.

[35] 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*, 2016, pp. 2818–2826.

[36] Y. Xu, Y. Xu, Q. Qian, H. Li, and R. Jin, "Towards understanding label smoothing," *arXiv preprint arXiv:2006.11653*, 2020.

[37] W. Li, G. Dasarathy, and V. Berisha, "Regularization via structural label smoothing," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, S. Chiappa and R. Calandra, Eds., vol. 108. PMLR, 26–28 Aug 2020, pp. 1453–1463. [Online]. Available: https://proceedings.mlr.press/v108/li20e.html

[38] H. Wei, L. Feng, X. Chen, and B. An, "Combating noisy labels by agreement: A joint training method with co-regularization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[39] C. F. G. d. Santos, M. Roder, L. A. Passos, and J. P. Papa, "Maxdropoutv2: An improved method to drop out neurons in convolutional neural networks," in *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 2022, pp. 271–282.

[40] B. Lim, S. Son, H. Kim, S. Nah, and K. Mu Lee, "Enhanced deep residual networks for single image super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 136–144.

[41] A. Ignatov, L. Van Gool, and R. Timofte, "Replacing mobile camera isp with a single deep learning model," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 536–537.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[43] H. Zhu and X. Zhao, "Targetdrop: A targeted regularization method for convolutional neural networks," *arXiv preprint arXiv:2010.10716*, 2020.
[44] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proceedings of the British Machine Vision Conference (BMVC)*, E. R. H. Richard C. Wilson and W. A. P. Smith, Eds. BMVA Press, September 2016, pp. 87.1–87.12. [Online]. Available: https://dx.doi.org/10.5244/C.30.87
[45] D. Han, J. Kim, and J. Kim, "Deep pyramidal residual networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5927–5935.
[46] C. F. G. d. Santos and J. a. P. Papa, "Avoiding overfitting: A survey on regularization methods for convolutional neural networks," *ACM Comput. Surv.*, jan 2022, just Accepted. [Online]. Available: https://doi.org/10.1145/3510413
[47] C. Santos, L. Afonso, C. Pereira, and J. Papa, "Breastnet: Breast cancer categorization using convolutional neural networks," in *2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS)*, 2020, pp. 463–468.
[48] C. Filipi Gonçalves dos Santos, D. d. S. Oliveira, L. A. Passos, R. Gonçalves Pires, D. Felipe Silva Santos, L. Pascotti Valem, T. P. Moreira, M. Cleison S. Santana, M. Roder, J. Paulo Papa *et al.*, "Gait recognition based on deep learning: a survey," *ACM Computing Surveys (CSUR)*, vol. 55, no. 2, pp. 1–34, 2022.
[49] C. F. G. dos Santos, T. P. Moreira, D. Colombo, and J. P. Papa, "Does pooling really matter? an evaluation on gait recognition," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, I. Nyström, Y. Hernández Heredia, and V. Milián Núñez, Eds. Cham: Springer International Publishing, 2019, pp. 751–760.
[50] C. F. G. d. Santos, T. P. Moreira, D. Colombo, and J. P. Papa, "Does removing pooling layers from convolutional neural networks improve results?" *SN Computer Science*, vol. 1, no. 5, pp. 1–10, 2020.
[51] C. F. G. dos Santos, L. A. Passos, M. C. de Santana, and J. P. Papa, "Normalizing images is good to improve computer-assisted covid-19 diagnosis," in *Data Science for COVID-19*, U. Kose, D. Gupta, V. H. C. de Albuquerque, and A. Khanna, Eds. Academic Press, 2021, pp. 51–62. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128245361000332
[52] L. A. Passos, C. Santos, C. R. Pereira, L. C. S. Afonso, and J. P. Papa, "A hybrid approach for breast mass categorization," in *VipIMAGE 2019*, J. M. R. S. Tavares and R. M. Natal Jorge, Eds. Cham: Springer International Publishing, 2019, pp. 159–168.
[53] R. G. Pires, D. F. S. Santos, C. F. Santos, M. C. Santana, and J. P. Papa, "Image denoising using attention-residual convolutional neural networks," in *2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, 2020, pp. 101–107.
[54] G. H. De Rosa, M. Roder, J. P. Papa, and C. F. Dos Santos, "Improving pre-trained weights through meta-heuristics fine-tuning," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2021, pp. 1–8.