IIEEE International Conference on Image Processing (ICIP),
Bordeaux, France, 2022

DOI: https://doi.org/10.1109/ICIP46576.2022.9897273

CCL: CLASS-WISE CURRICULUM LEARNING FOR CLASS IMBALANCE PROBLEMS.

Marcos Escudero-Viñolo, Alejandro López-Cifuentes

Video Processing & Understanding Lab. Universidad Autónoma de Madrid.

ABSTRACT

Computer vision datasets usually present long-tailed training distributions where the classes are not represented with the same number of training samples. This so-called class imbalance problem hinders the proper learning of inference models, biasing them towards over-represented classes and decreasing their generalization. Adopted solutions to tackle the effect of class imbalance are based on weighting the training loss according to the number of class samples, leading to regimes where low-represented classes guide the learning just accounting for their cardinal number. To also incorporate class complexity in the process, we propose a novel training scheme called CCL: Class-wise Curriculum Learning. Classes are first sorted based on a difficulty criterion which not only accounts for the number of training samples but also for their training outcomes. The curriculum is then used to guide the training: easy classes are fed first and—incrementally, the more difficult ones are added. The proposed approach is validated for image classification using long-tailed datasets. Results show that when the proposed Class-wise Curriculum Learning scheme is used, trained models outperform specific state-of-the-art methods devoted to handle the class imbalance problem. The code, data and reported models described along this paper are publicly available at https://github.com/vpulab/CCL.

Index Terms— Class imbalance, Curriculum learning, Sample scoring, Training pace, Image Classification.

1. INTRODUCTION AND RELATED WORK

The advent of deep learning architectures, pushed by enormous amounts of visual data, have led to the blossom of computer vision. Deep Learning (DL) solutions and—in particular, Convolutional Neural Networks (CNNs) have proven to significantly outperform traditional approaches on several computer vision tasks. These solutions generally rely on a learning stage to propagate training knowledge towards unobserved data. The performance of the learned models is highly impacted by the training stage; hence, learning biases have a key impact on their whole operation. Common biases in learning include the class imbalance and the difficulty-imbalance problems.

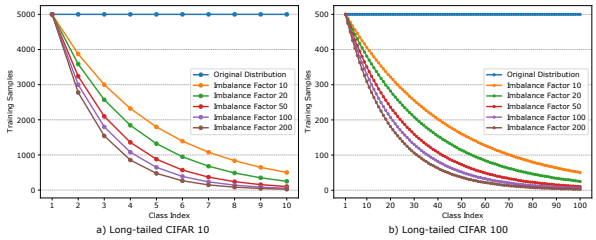

Fig. 1. Long-tailed CIFAR-10 and CIFAR-100 training distributions with different Imbalance Factors.

Imbalance Problems. In imbalanced scenarios, the class priors—the number of per-class samples, usually depict a long-tailed distribution (see examples in Figure 1), resulting in the high-populated classes dominating the learning process, hindering the learning of low represented classes [1]. Class imbalance situations may be caused by impaired, biased or erroneous data sampling methods, where some training classes are heavily populated whereas some others are just exemplified by few samples, or can arise naturally, as they are intrinsic to the task. Existing approaches to tackle the effect of imbalance can be broadly organized into four categories: adding more new training data, re-sampling techniques, weighting the training loss and feature-driven.

Adding more new training data to the under represented classes is usually a good option for maximizing performance: the availability of larger data sets is a key factor for success, but usually entails considerable costs in terms of human resources, energy and time. Alternatively, using augmented versions of the available data is usually preferred, e.g., by combining samples [2], [3]. **Re-sampling techniques** are mainly based on two regimes: under-sampling (removing training samples from high represented classes until class-balance is achieved) and over-sampling (adding repeated training samples for under represented classes). Whereas under-sampling is prone to suppress relevant training samples, generally decreasing overall performance, over-sampling may lead to a decrease of intra-class variation, inducing the model to high over-fitting and to learn features with a lower generalization power for the under represented classes. Alternatively, there

Partially founded by the Comunidad de Madrid through the SI1/PJI/2019-00414 AISEEME project

are solutions proposing to increase the number of samples by interpolating existing ones [4], or by creating new synthetic images [5]. **Loss weighting** techniques tackle the class imbalance problem at the training stage by weighing the class contributions to the loss. Generally, most of the methods are based on assigning class weights depending on the number of training samples per class. Common approaches usually define weighting by inverse class frequency strategies [6, 7] or smoothed versions of these strategies relying on square roots of class frequency [8, 9]. These methods use the whole training set for learning and force CNNs to learn features from under-represented classes by adjusting the contribution of samples from well-represented training classes in the loss. However, the features learnt from under-represented classes aren't necessarily representative ones, harming the overall learning process. To overcome this issue, *Cui et al* [10] proposed to measure data overlap and to define an effective number of samples for each class; hence, reducing the imbalance ratio. Differently, instead of relying on fixed weights, Focal Loss [11] defines a schedule that allows to dynamically weight hard examples during training, inducing the network to adapt the learnt features towards them. Loss weighting strategies are able to deal with imbalanced problems. However, this promotes an over-influence in the learning process of under-represented or difficult to learn, samples. Finally, **feature-driven** techniques define frameworks for handling imbalance by separately training features and classification [12] and for scaling low-represented features and promoting the learning of hard inter-class boundaries samples [13], yielding current top performance in imbalanced problems.

Curriculum Learning. The difficulty-imbalance problem is a natural—yet sometimes underestimated, one in multi-class deep learning tasks. During training, the features describing some classes' domains are harder to be learned than others. This may be caused by a combination of large intra-class variations, large inter-class overlapping and class imbalance problems. Even though class population and class difficulty are not always correlated, if easier classes are more populated, the training process may naturally evolve towards learning easy, high populated classes, as their learning helps to minimize the global cost function. The use of curriculum learning approaches is an accepted choice for incorporating estimates on the difficulty of samples in the training.

Humans and animals learn better when the learning examples are not randomly presented but organized in a meaningful order that illustrates gradually more concepts, and gradually more complex ones [14]. Curriculum learning paradigm is the formalization of this idea into a training strategy. Training CNNs is usually done by providing a sequence of random mini-batches sampled uniformly from the entire training data. Curriculum learning involves the non-uniform sampling of mini-batches. The learning starts by training only easy samples of a task and then gradually and incrementally increase the task difficulty. *Hacohen et al* [15] suggested that Curriculum Learning improves classification results in a broad set of datasets such as CIFAR-10 [16] and CIFAR-100 [16] when samples are sorted with respect to their difficulty. *Soviany et al* [17] proposed to use a predefined curriculum to ease Generative Adversarial Networks learning by feeding them first with easy samples. Finally, *Graves et al* [18] proposed an automated curriculum learning that increased the learning efficiency. Given the power of Curriculum Learning to obtain features that generalize better for unseen samples, we propose to use this idea to tackle class imbalance by defining a Class-wise Curriculum Learning (CCL) that instead of sorting the samples, sorts classes by difficulty.

In this paper we describe our contribution to both problems: CCL aims to use prior knowledge about the difficulty of the target classes in order to define class-wise selection strategies to consider non-uniformly sampled incremental subsets of training examples during learning. The intuition behind CCL is that the learning process is boosted when simpler classes are fed first. In particular, we hypothesize that for highly imbalanced distributions, the prior learning of easier classes may benefit the learning of more complex or less populated classes.

2. CLASS-WISE CURRICULUM LEARNING

Definition. Let $\mathbb{X} = \{(x_i, y_i)\}_{i=1}^{N}$ denote the training data, with each $x_i \in \mathbb{R}^d$ being a sample and $y_i \in \mathbb{C} = \{1, 2, ...C\}$ its label out of C classes. In classification problems, the aim is to learn the parameters θ of a classifier $\phi(\theta, \mathbb{X}) : \mathbb{R}^d \rightarrow \mathbb{C}$.

The general process is to train $\phi(\theta, \mathbb{X})$ sequentially, using uniformly sampled mini-batches of $\mathbb{X}$. The learning is guided by the optimization of a loss function, a common choice for the loss function in classification problems is the cross-entropy loss. A so-trained model is a *Baseline* one.

Sample-wise curriculum learning strategies first divide $\mathbb{X}$ into subsets of increasing complexity—measured by a scoring function on the training samples $s(x_i, y_i)$. From these subsets, mini-batches are uniformly sampled, resulting in a non-uniform sampling of $\mathbb{X}$. These mini-batches are then sequentially fed to the learning process according to a pacing function $h_\phi(e, \tau)$. The pacing function defines the subsets to be sampled at each epoch e of the learning process according to an update hyper-parameter τ.

The proposed CCL strategy follows the same idea but defines the scoring and pacing functions on a per-class basis. $\mathbb{X}$ is first divided into class subsets: $[\mathbb{X}_1 ... \mathbb{X}_c ... \mathbb{X}_C] \subseteq \mathbb{X}$, with $\mathbb{X}_c = \{(x_i, y_i) \mid y_i = c\}_{i=1}^{N_c}$, and N_c the number of training samples for class c. The subsets are ordered according to a scoring function $s(\mathbb{X}_c)$ that accounts for all the training samples of class c. The pacing function $h_\phi(e, \tau)$ determines how and when incorporating the subsets in the training process.

Scoring and Pacing Functions. We evaluate two scoring functions: *Self-taught* and *Self-paced*. For the *Self-taught* scoring function, we train the network using uniformly sampled mini-batches without curriculum, following the *Baseline* model approach. Then, the per-class losses $\mathcal{L}_c$ of this trained model on the training set are extracted and the $\mathbb{X}_c$ subsets are sorted in increasing complexity order according to these loss values—the lower the loss, the simpler the subset: $s\left(\mathbb{X}_c\right) = \mathcal{L}_c$. This order is fixed during the training of the CCL model. The *Self-paced* scoring function does not require a prior model as the $\mathbb{X}_c$ subsets are ordered according to the classes' losses after a warm-up period. This order is updated at each epoch e based on the corresponding loss values at its end: $s_e\left(\mathbb{X}_c\right) = \mathcal{L}_c^e$, with $\mathcal{L}_c^e$ the loss for class c at epoch e.

The pacing function, $h_\phi(e, \tau)$ is defined as a monotonically increasing function, i.e., at each updating stage of CCL, all the samples for a specific number of classes are included. The process is guided by a ρ parameter ($\rho \in \mathbb{R} \mid 0 < \rho \leq 1$), that defines the relative classes increment at each updating stage, depicting a stair-case pacing function. We use a regular updating strategy by including $\lfloor \rho \cdot C \rfloor$ additional classes in the learning process every τ epochs.

CCL Method. The combination of the *Self-taught* scoring function and the pacing function is as follows:

(i) The class subsets are ordered according to their loss estimates at the output of a learnt *Baseline* model:

$$[\mathbb{X}_1 ... \mathbb{X}_c ... \mathbb{X}_C] \xrightarrow{s(\mathbb{X}_c), \forall c} [\mathbb{X}_{(1)} ... \mathbb{X}_{(j)} ... \mathbb{X}_{(C)}], \quad (1)$$

with $s\left(X_{(j)}\right) \leq s\left(X_{(j+1)}\right)$ and $X_{(j)}$ the jth scored subset. **(ii)** The first $\lfloor \rho \cdot C \rfloor$ classes in the ordered set of subsets are included in the training by defining an initial set of subsets:

$$\mathbb{X}^0 = \bigcup_{j=1}^{\lfloor \rho \cdot C \rfloor} \mathbb{X}_{(j)}, \quad (2)$$

and the model is trained considering only uniformly sampled mini-batches from the training samples of the classes in $\mathbb{X}^0$. **(iii)** Every τ epochs—$\{e = k\tau, k \in [1, C-1]\}$, the set of considered classes is updated by incorporating $\lfloor \rho \cdot C \rfloor$ new classes to the training set:

$$\mathbb{X}^e = \bigcup_{j=1}^{k\lfloor \rho \cdot C \rfloor} \mathbb{X}_{(j)} \iff e = k\tau. \quad (3)$$

At epoch $e = C\tau$, with all the classes included, the learning rate is decayed following a predefined decay policy.

When the *Self-paced* scoring function is used, the process is equivalent with slight modifications: step (i) is performed after the warm-up stage and then at every epoch, step (ii) starts after the warm-up stage, and step (iii) incorporates a gap equal to the number of warm-up epochs in its updating policy.

3. EXPERIMENTAL RESULTS

Datasets and Training Details. We conduct a set of experiments on CIFAR-10 [16] and CIFAR-100 [16] datasets. Both datasets are class-balanced and are composed of 50.000 training and 10.000 validation images. CIFAR-10 is made up of 10 different classes whereas CIFAR-100 extends it to 100 classes. To create long-tailed versions for both CIFAR-10 and CIFAR-100, we follow an existing unbalancing process [10]. The number of samples per class is reduced by randomly sampling N samples for each class according to an exponential function: $N = N_c \mu^c$, where c is the class index, N_c is the original number of training images and $\mu \in (0, 1)$ is a parameter controlling the imbalance[1]. Validation sets remain unchanged. The imbalance factor is defined as the ratio between the number of training samples in the highest populated class divided by that of the lowest populated one. Figure 1 depicts long-tailed training distributions from Imbalance Factors ranging from 10 to 200 for CIFAR-10 and CIFAR-100.

We use the same training procedure defined by *Cui et al.* [10]. The ResNet-32 architecture [19] is used as backbone for image classification. All models in the experiments ahead are fully trained from scratch using regular Stochastic Gradient Descend with Momentum (SGD) as the optimizer function and 128 samples batches. We use an initial linear warm-up stage for the first 5 epochs with a starting learning rate of 0.001 and ending in 0.1, the learning rate for training that is decayed every 200 epochs by 0.01. For CCL the decay policy is only applied when all the class subsets have been fed to the learning process (see Section 2). For data augmentation we use padding, random crop, horizontal flips and normalization operations. The models design, training and evaluation has been implemented using PyTorch 1.7.0 DL framework ([20]) running on a PC using a 12 Cores CPU, 30 GB of RAM and a NVIDIA TITAN Xp 12GB GPU.

Parameter Setup. We have conducted two experiments to asses the effect of the hyper-parameters of the pacing function $h_\phi(\tau)$: the effect of relative class increment ρ (see top of Figure 2) and the number of τ epochs (see bottom of Figure 2). All the results have been obtained using an Imbalance Factor of 50 for both CIFAR datasets. For comparison, *Baseline* (see section 2) performance is included. Results in Figure 2 (top) indicate that there is a range (between 0.1 and 0.6) of ρ in which the CCL method clearly outperforms the *Baseline*'s decreasing error rates for both CIFAR-10 and CIFAR-100. From there, as ρ increases the performance of CCL method linearly converges towards *Baseline's* performance, until $\rho = 1$ for which both approaches are equivalent. Regarding the influence on the performance of increasing τ, i.e., the number of epochs a given subset of classes is trained depicted in Figure 2 (bottom), results suggest that

[1]Long-tailed versions of CIFAR-10 and CIFAR-100 will be made publicly available

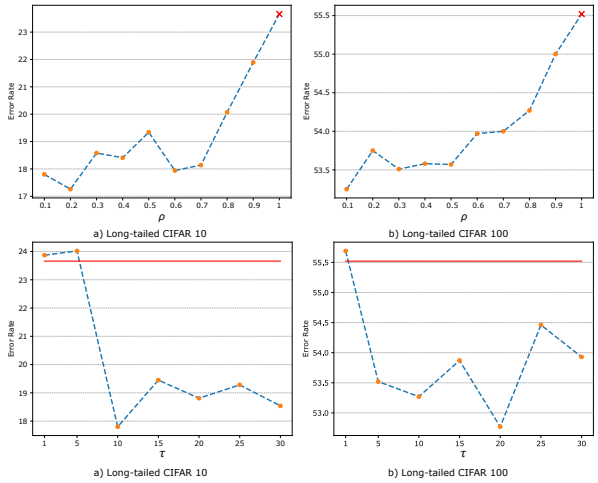

Fig. 2. Results (in error rates) for different relative class increment ρ (top) and τ values (bottom) in long-tailed CIFAR-10 and CIFAR-100. The red cross (top) and line (bottom) represents the performance of the *Baseline*.

there is a lower limit for τ: small values seem to be insufficient for learning representative features. However, using large τ values might derive in over-fitted representations, hindering feature generalization to new classes. Furthermore, as the decay policy is only applied after epoch $C\tau$, a larger step size τ increases the learning time. We select $\rho = 0.1$ and $\tau = 10$ as trade-off parameters that yield good performance for both datasets.

State-of-the-art Comparison. Table 1 compares reported performances of state-of-the-art methods with those of the proposed training protocol in its Self-Paced (CCL-SP) and Self-Taught (CCL-ST) variants. For comparison, we include two versions of mix-up augmentation strategies ([2], [3]), loss-weighting methods such as Focal Loss [11] and Class-Balanced Focal Loss [10], a margin-loss regularization [21], and two feature-driven methods [12] [13]. The proposed curricula naturally accounts for semantic inter-class similarities together with the number of samples, e.g., for CIFAR-10, the *Cat* class is the fourth one in terms of samples, but is considered the most difficult one in the curriculum of the Self-Taught variant. To further validate the proposed CCL strategy, we have also included an Anti-Curriculum version (CCL-A). CCL-A equals Self-Taught method but reversing the obtained class order, i.e., the most complex classes are learnt first. Performances in Table 1 are either extracted from [13]—indicated by the * symbol, or with models built with the same backbone and trained from scratch using the hyper-parameters suggested by their authors.

Performances in Table 1 indicate that class complexity is also a key factor for explaining performance of class imbalance trained models. The simple yet effective CCL-ST protocol outperforms all the compared methods for slightly imbalanced CIFAR-10 and CIFAR-100 datasets and performs close to leading approaches for highly imbalanced factors. Notwithstanding, for the balanced scenario—where Imbalance Factor is 1, the proposed learning strategy reduces the error rates, suggesting that even with uniform class distribution, following a proper order in the class feeding is beneficial, agreeing with the Curriculum Learning paradigm [15]. The CCL-ST method benefits from stronger evidences on the classes' complexities provided by the *Baseline* model, yielding slightly better results than CCL-SP for the analyzed imbalance factors. CCL-A performs worse than all the other reported methods disregarding the imbalance factor; stressing the relevance of using adequate scoring functions that sort the classes in an useful order. If difficult classes are fed first results might be even worse than feeding all the classes simultaneously.

Table 1. ResNet-32 error rates results for Long-Tailed CIFAR 10 (top) and CIFAR 100 (bottom). *: as reported in [13].

Dataset	Long-Tailed CIFAR 10					
Imbalance	200	100	50	20	10	1
Baseline	36.25	28.95	23.66	17.18	13.35	6.71
[11] ($\gamma = 1$)	34.71	29.62	23.29	17.24	13.34	6.60
[10]	31.11	25.43	20.73	15.64	12.51	6.36
[2] (*)	-	26.94	-	-	12.90	-
[3] (*)	-	24.64	-	-	11.85	-
[21] (*)	-	22.97	-	-	11.84	-
[12] (*)	-	20.18	-	-	11.68	-
[13] (*)	-	**19.91**	-	-	11.61	-
CCL-A	40.18	27.16	23.74	17.21	13.86	6.64
CCL-SP	29.93	24.14	17.82	13.47	9.96	5.66
CCL-ST	**29.75**	23.87	**17.80**	**12.14**	**9.76**	**4.95**
Dataset	Long-Tailed CIFAR 100					
Imbalance	200	100	50	20	10	1
Baseline	68.33	61.29	55.52	49.07	43.09	28.66
[11] $\gamma = 1$	64.38	61.59	55.68	48.05	44.22	28.85
[10]	**63.40**	60.40	54.68	47.41	42.01	28.39
[2] (*)	-	60.46	-	-	41.98	-
[3] (*)	-	58.06	-	-	40.64	-
[21] (*)	-	57.96	-	-	41.29	-
[12] (*)	-	57.44	-	-	40.88	-
[13] (*)	-	**57.03**	-	-	40.64	-
CCL-A	69.19	63.82	56.32	51.47	45.77	28.89
CCL-SP	64.61	58.30	54.43	47.62	**39.46**	27.78
CCL-ST	64.28	58.20	**53.25**	**45.53**	40.43	**26.30**

4. CONCLUSIONS

In this paper, we describe Class-wise Curriculum Learning (CCL) strategies to train target classes in a non-uniform fashion to tackle the effect of class imbalance problems in long-tailed datasets. Experimental results in highly imbalanced datasets support the advantages of using class curricula for image classification. Our future work will explore the use of CLL for multi-label problems and loss-driven strategies to dynamically adapt the step values in the pacing function and its combination with feature-driven methods.

5. REFERENCES

[1] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas, "Imbalance problems in object detection: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[2] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

[3] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan, "Remix: Rebalanced mixup," in *Computer Vision – ECCV 2020 Workshops*, Adrien Bartoli and Andrea Fusiello, Eds., Cham, 2020, pp. 95–110, Springer International Publishing.

[4] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[5] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang, "Unsupervised domain adaptation for semantic segmentation via class-balanced self-training," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 289–305.

[6] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang, "Learning deep representation for imbalanced classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5375–5384.

[7] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert, "Learning to model the tail," in *Advances in Neural Information Processing Systems*, 2017, pp. 7029–7039.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[9] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten, "Exploring the limits of weakly supervised pretraining," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 181–196.

[10] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie, "Class-balanced loss based on effective number of samples," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9268–9277.

[11] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[12] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen, "Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[13] Zhe Wang, Qida Dong, Wei Guo, Dongdong Li, Jing Zhang, and Wenli Du, "Geometric imbalanced deep learning with feature scaling and boundary sample mining," *Pattern Recognition*, vol. 126, pp. 108564, 2022.

[14] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[15] Guy Hacohen and Daphna Weinshall, "On the power of curriculum learning in training deep networks," in *Proceedings of the 36 the International Conference on Machine Learning*, 2019.

[16] Alex Krizhevsky, Geoffrey Hinton, et al., "Learning multiple layers of features from tiny images," 2009.

[17] Petru Soviany, Claudiu Ardei, Radu Tudor Ionescu, and Marius Leordeanu, "Image difficulty curriculum for generative adversarial networks (cugan)," in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 3463–3472.

[18] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu, "Automated curriculum learning for neural networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1311–1320.

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

[20] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer, "Automatic differentiation in pytorch," 2017.

[21] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. 2019, vol. 32, Curran Associates, Inc.