MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

No Class Left Behind: A Closer Look at Class Balancing for Audio Tagging

Ebbers, Janek; Germain, François G; Wilkinghoff, Kevin; Wichern, Gordon; Le Roux, Jonathan

TR2025-037 March 08, 2025

Abstract

Large-scale audio tagging datasets like AudioSet usually suffer from severe class imbalance comprising many audio examples for common sound classes but only few examples of rare sound classes. The latter, however, may yet be equally or even more important to recognize. Therefore, it is common practice to sample examples from rare classes more frequently during training. At the same time, the effects of such balancing on a model's training and tagging performance are still little understood. In this work, we investigate how it affects training convergence and tagging performance. We consider varying degrees of balancing and investigate whether classes converge simultaneously or if there is a benefit from selecting different balancing rates for each class. Furthermore, we investigate data efficient oversampling, which keeps audio files from rare classes in memory, and repeats them in close succession over multiple batches, minimizing data loading from disk. Finally, we show that for AudioSet, the optimal amount of class balancing is different when fine-tuning a model pre-trained via self-supervised learning, versus training a supervised model from scratch.

IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2025

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

No Class Left Behind: A Closer Look at Class Balancing for Audio Tagging

Janek Ebbers, François G. Germain, Kevin Wilkinghoff, Gordon Wichern, Jonathan Le Roux

Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA

***Abstract*—Large-scale audio tagging datasets like AudioSet usually suffer from severe class imbalance comprising many audio examples for common sound classes but only few examples of rare sound classes. The latter, however, may yet be equally or even more important to recognize. Therefore, it is common practice to sample examples from rare classes more frequently during training. At the same time, the effects of such balancing on a model's training and tagging performance are still little understood. In this work, we investigate how it affects training convergence and tagging performance. We consider varying degrees of balancing and investigate whether classes converge simultaneously or if there is a benefit from selecting different balancing rates for each class. Furthermore, we investigate data efficient oversampling, which keeps audio files from rare classes in memory, and repeats them in close succession over multiple batches, minimizing data loading from disk. Finally, we show that for AudioSet, the optimal amount of class balancing is different when fine-tuning a model pre-trained via self-supervised learning, versus training a supervised model from scratch.**

***Index Terms*—audio-tagging, class balancing, AudioSet**

I. Introduction

Audio tagging systems apply time-invariant class labels to audio recordings based on the present sound events, and multiple class labels can be applied to each audio clip. Progress in audio tagging research accelerated significantly with the release of AudioSet [1], a large dataset of approximately 10-second-long audio recordings collected from YouTube videos and human-annotated from a list of 527 audio class labels. Many widely-used pre-trained audio representation models were trained on the audio tagging task of AudioSet [2]–[5], and it is the standard task when extending successful network architectures originally developed for the ImageNet classification task in computer vision to the audio domain [2], [4]. Because of its importance in the field, many researchers have focused attention on subtleties in the recipes used for training AudioSet tagging models.

These training improvements include data augmentation [6], [7], knowledge distillation [8], [9], and label enhancement [10], among others. However, because there is a large skew in the available number of audio samples for each of the 527 classes in AudioSet, balancing the class distribution [11] has emerged as a promising way to improve performance by ensuring that trained models are not overly biased towards the most commonly occurring classes. Class balancing on AudioSet is further motivated because the commonly used evaluation set was designed to be balanced, i.e., contain an equal number of examples per class, although some class imbalance remains due to the multi-label nature of the dataset (i.e., many audio files have labels corresponding to both rare and prevalent classes). Furthermore, performance on the evaluation set is usually reported as the mean of per class average precisions (mAP), so both prevalent and rare classes contribute equally to overall performance. Thus, sampling training samples in a manner that encourages a uniform distribution over AudioSet labels is expected to better match both the test distribution and the evaluation metric, leading to higher performing models. Class balancing, by either oversampling training examples or weighting the loss function for different classes, has also been widely used in other fields such as vision [12], [13], audio onset detection [14], and source separation [15].

While multiple works have demonstrated benefits of class balancing strategies for audio tagging performance [3], [4], [10], [16], it was pointed out in [10] that class-wise performance on AudioSet is often not well correlated with the number of samples in the training set containing that class label, or the quality of the class labels, which were also collected by the AudioSet authors. Furthermore, a recent paper by many of the AudioSet creators [17] shows that the benefits of class balancing are fragile and do not extend to another dataset with similar class distribution characteristics. They also showed that, while some classes benefited from balanced sampling during training, others exhibited a drop in performance, and the performance improvements from balancing were not significantly correlated to the prior probability on the presence of that class in the training set. However, [17] did note that models with class balancing converged more quickly, likely because fewer training batches were required to see all classes a sufficient number of times.

Given this conflicting evidence on the benefits of naive class balancing for AudioSet, we further study its impact in this work. We first train a set of models with different amounts of class balancing to investigate their impact on performance and convergence. We further propose and evaluate an approach for determining class-specific oversampling amounts. Moreover, we investigate a local oversampling approach, which keeps oversampled training samples in memory and repeats them in close succession over multiple batches. This minimizes data loading operations (which can be a bottleneck in modern training pipelines) compared to global oversampling, which naively reads oversampled files from disk every time they are used.

An additional aspect of class balancing for AudioSet that, to the best of our knowledge, has yet to be studied, is how conclusions on its effectiveness differ for training a model from scratch using supervised learning, versus the currently in vogue training paradigm that first pre-trains a model using self-supervised learning, and then uses supervised fine-tuning [5], [18]–[22]. On experiments fine-tuning the BEATs [5] model, we show that stronger class balancing (i.e, making the class distribution more uniform) is more effective for fine-tuning compared to training a supervised model from scratch.

II. Class Balancing for Audio Tagging

A. Preliminaries

Following the notation from [17], we denote by N the total number of training audio files and by N_k the number of files where sound class k is present. An imbalance ratio can be computed as

$$\rho = \frac{N_{\text{max}}}{N_{\text{min}}} = \frac{\max_k N_k}{\min_k N_k} \tag{1}$$

Typically $\rho \gg 1$, e.g., in the case of AudioSet the training set's imbalance ratio is $\approx 15\,\text{k}$ meaning the most frequent sound class "Music" ($\approx 1\,\text{M}$ examples) appears $15\,\text{k}$ times more often than the least frequent class "Toothbrush" (67 examples).

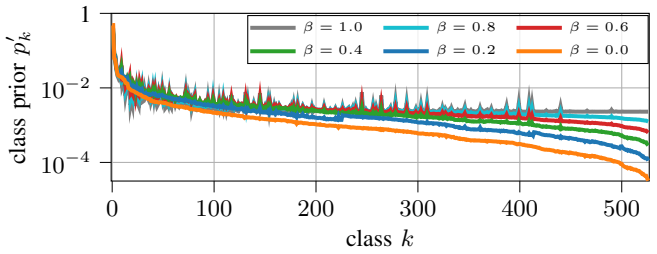

Fig. 1. Ratio of clips containing each class k after oversampling using the balancing exponent approach from Eq. (3).

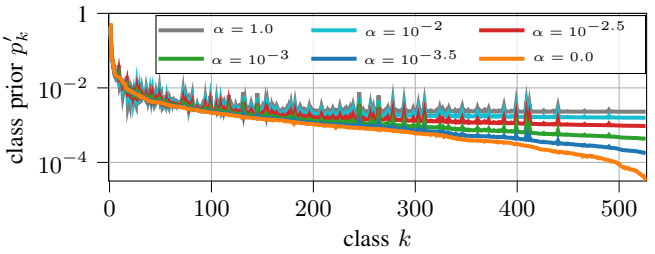

Fig. 2. Ratio of clips containing each class k after oversampling using the linear interpolation approach from Eq. (4).

To overcome such imbalance, different balancing strategies have been employed. A common approach is full oversampling [3], [4], [10], where training examples are sampled uniformly across classes. Consequently, an example containing a rare class is sampled more often than examples of more common classes. Denoting by $c_{j,k}$ a binary tag label indicating absence ($c_{j,k} = 0$) or presence ($c_{j,k} = 1$) of sound class k in the j-th audio example, this is analogous to oversampling the j-th audio example by a factor of

$$m_j = \sum_{k:c_{j,k}=1} \underbrace{\frac{N_{\text{max}}}{N_k}}_{r_k}, \tag{2}$$

where the class oversampling r_k is set such that each class has the same pool size of N_{max} examples.

To prevent excessive oversampling of audio examples which have multiple tags, the authors of [17] oversample an example only for the rarest present sound class rather than oversampling it for each of the present classes. Further, they introduce a balancing exponent β allowing to control the degree of oversampling and eventually round a class's oversampling r_k to an integer number:

$$m_j = \max_{k:c_{j,k}=1} \underbrace{\left\lfloor \left(\frac{N_{\text{max}}}{N_k}\right)^{\beta} \right\rceil}_{r_k}. \tag{3}$$

As an alternative, we propose the following linearly interpolated oversampling scheme:

$$m_j = \max_{k:c_{j,k}=1} \underbrace{1 + \alpha\left(\frac{N_{\text{max}}}{N_k} - 1\right)}_{r_k} \tag{4}$$

which does interpolate between no oversampling ($\alpha = 0$) and full oversampling ($\alpha = 1$), with $\alpha \in [0, 1]$ denoting the *balancing rate*. Note that we do not round the oversampling to an integer number here. Instead, we implement a non-integer oversampling by randomly selecting ceiling or flooring of m_j in each epoch such that the expected oversampling of audio j over epochs matches m_j.

Note that all the above oversampling strategies follow the scheme

$$m_j = \underset{k:c_{j,k}=1}{\text{aggregate}}\ r_k, \tag{5}$$

with different methods for aggregation and for computing r_k.

Given clip oversampling $m_j, \forall j$, the total number of oversampled examples N', the number of examples N'_k in which class k is present, and eventually the class prior p'_k, i.e., the probability of a class being present in a training example, can be derived as

$$N' = \sum_j m_j, \quad N'_k = \sum_j m_j \cdot c_{j,k}, \quad p'_k = N'_k / N'. \tag{6}$$

Figures 1 and 2 plot the prior (or rate) p'_k over classes k (with classes sorted according to N_k) for different values of the balancing exponent β and the balancing rate α, respectively, for AudioSet. It can be seen that without balancing ($\beta = 0.0$ and $\alpha = 0.0$), class priors differ by multiple orders of magnitude. When using full oversampling ($\beta = 1.0$ and $\alpha = 1.0$), most classes have an approximately equal rate p'_k, with the "noise" in the rates resulting from the multi-label nature of AudioSet, as repeating any audio clip with multiple labels will impact the rate of all its classes. Changing β and α allows for controlling the degree of balancing.

Note that, for $\alpha = 10^{-2}$, class priors already almost match the distribution of full oversampling. This is because for most classes $N_k \ll N_{\text{max}} \cdot 10^{-2}$ so that $r_k \approx \alpha \frac{N_{\text{max}}}{N_k}$ when $\alpha \geq 10^{-2}$. Further increasing α hence mainly scales up the dataset size without changing class distribution much. As found in [10], the size of the oversampled data with full oversampling is over 400 M. In practice, this results in many audio clips, especially those from less-oversampled classes, to never be seen in training. In contrast, using $\alpha = 10^{-2}$, which gives almost the same class priors, results in a dataset size of only approx. 6 M ($\times 3$ compared to original dataset size). Using a balancing exponent of $\beta = 0.8$ instead to obtain a nearly balanced class distribution (c.f. Fig. 1), also yields a huge dataset size of over 100 M clips, making our proposed interpolation approach favorable for strong balancing (i.e., for larger values of α resp. β).

B. Tuned Oversampling

When it comes to finding the ideal degree of oversampling (α or β), we hypothesize that this may vary across classes, with some classes benefiting from stronger balancing than others. Hence, we find for each class different optimal class priors $\hat{p}'_k$ originating from different values for α or β.

Then, the question arises how to derive corresponding class oversampling $\hat{r}_k, \forall\, k$ so that the desired class priors can be simultaneously achieved in a single training run. To do so, we first express computation of p'_k from r_k (Eqs. (5)-(6)) as a matrix operation as follows:

$$\mathbf{p}' = \mathbf{C}^{\mathrm{T}}\mathbf{m}/N', \quad \text{with} \quad \mathbf{m} = \mathbf{A}\mathbf{r}, \tag{7}$$

where $\mathbf{m} = (m_j)_{j=1}^{N}$, $\mathbf{r} = (r_k)_{k=1}^{K}$, and $\mathbf{p}' = (p'_k)_{k=1}^{K}$ are vectors, $\mathbf{C}$ is the $(N \times K)$ matrix of tag labels $c_{j,k} \in \{0, 1\}$, and $\mathbf{A}$ is the $(N \times K)$ matrix representing aggregation from Eq. (5).

For a given $\hat{\mathbf{p}}'$ and $\hat{N}'$, with the latter being the desired number of examples in an epoch, we can obtain $\hat{\mathbf{r}}_k$ as the solution of the constrained optimization problem

$$\begin{aligned} &\text{minimize } ||\hat{\mathbf{p}}' - \mathbf{W}\hat{\mathbf{r}}/\hat{N}'||^2 \\ &\text{subject to } \mathbf{1}^{\mathrm{T}}\mathbf{A}\hat{\mathbf{r}} = \hat{N}' \text{ and } \hat{r}_k \geq 1\, \forall\, k. \end{aligned} \tag{8}$$

with $\mathbf{W}{=}\mathbf{C}^{\mathrm{T}}\mathbf{A}$ and $\mathbf{1}$ being a vector of ones so that $\mathbf{1}^{\mathrm{T}}\mathbf{A}\hat{\mathbf{r}} = \sum_j \hat{m}_j$. However, when aggregation in Eq. (5) is the max operation, as in Eqs. (3) and (4), $\mathbf{A}$ itself depends on $\mathbf{r}$ and hence cannot be computed beforehand to solve the optimization problem in Eq. (8). Therefore, we use the mean operation here so that $\mathbf{A}$ is independent from $\mathbf{r}$.

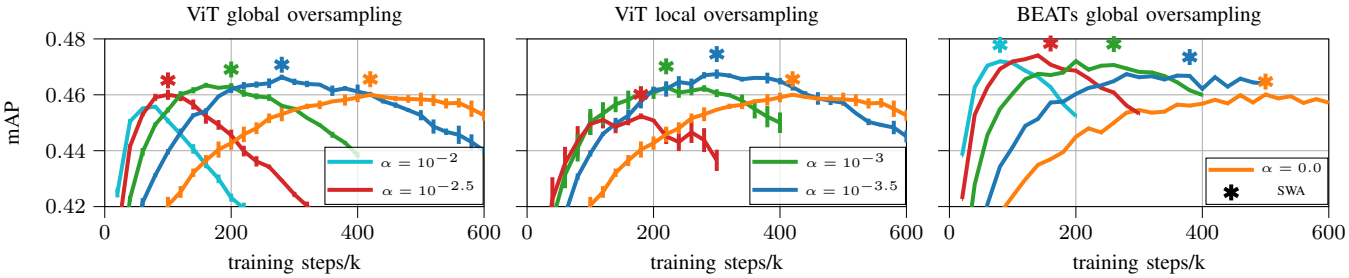

Fig. 3. Evaluation results for trainings with different degree of balancing by using different values for α from Eq. (4).

C. Data Efficient Oversampling

The common way of implementing oversampling is by simply duplicating the examples in the dataset, which results in the j-th example being loaded m_j times (on average) across different training steps of an epoch. The duplicates of an example are randomly scattered across the epoch, which we refer to as global oversampling.

However, with the constantly increasing training data throughput due to hardware and software improvements, there is also an increasing need for efficient data loading pipelines to not overload data servers and network connections.

Therefore, we propose a more data efficient way of oversampling, which we refer to as local oversampling in the following, and which temporally keeps a loaded example j in memory and reuses it m_j times in close succession over different batches. This further allows for bundling multiple examples with different m_j in larger files, which makes data loading more efficient compared to loading many small files. In contrast, this is not easily realizable with global oversampling, as examples with different m_j have to be reloaded multiple times. Hence, local oversampling loads data less frequently and more efficiently. On the counter side, it does not uniformly scatter duplicate examples over an epoch as global balancing does, the effects of which are to be evaluated in this work.

D. Class Balancing for Fine-tuning

Recently, self-supervised pre-trained models have been shown to achieve state-of-the-art performance in audio tagging [5], [20], [22]. In these approaches, models are first pre-trained on AudioSet using self-supervised objectives such as masked spectrogram modeling [21]. Then, the model is fine-tuned for the target task with a supervised objective. Interestingly, when fine-tuning on AudioSet, this approach appears to outperform models trained with a supervised objective from scratch even without using any additional data.

With this new paradigm of training audio tagging models, the question arises as to what extent the requirements for class balancing differ between the fine-tuning of a pre-trained self-supervised model versus fully supervised training of a model.

III. Experiments

We conduct a set of experiments to evaluate the impact of balancing on the training of state-of-the-art Transformer-based audio tagging models. Particularly, we want to answer the following questions:

1) How does balancing impact convergence of the model?
2) How does the ideal degree of oversampling vary across classes?
3) Can we achieve competitive performance using the data efficient local oversampling instead of global oversampling?
4) How does the impact of balancing differ when fine-tuning a pre-trained self-supervised model vs. fully supervised training?

For the first experiments, we consider the training of an state of the art Audio Spectrogram Transformer (AST) [4], [23] model following the Vision Transformer Base (ViT-B) architecture from [24] and using pre-trained weights from the vision domain[1] as weight initialization. As classification head, we use a two-layer MLP with BatchNorm [25] and GeLU [26] activation function in the hidden layer. As input feature map, we extract a log-mel spectrogram using a short-time Fourier transform (STFT) window size and shift of 60 ms and 20 ms, respectively, and 128 mel filters up to a maximum frequency of 16 kHz yielding a feature map size of 500×128 for a 10 sec audio clip. Patch size is chosen to be 16×16 with a patch stride of 10×10.

All experiments are performed using AudioSet with approx. 2 M and 20 k clips in the official training and evaluation sets, respectively. From the training data, we further split three disjoint validation folds by randomly selecting 60 examples per class or, for classes with less than 300 examples, 20% of the available examples, which yields validation folds with approx. 20 k examples.

For training, we use a batch size of 128 clips, AdamW [27] optimizer with a weight decay of 10^{-4} and a cosine learning rate (lr) annealing schedule with $\text{lr}_{\text{max}} = 10^{-4}$, $\text{lr}_{\text{min}} = 10^{-6}$, and warm restarts [28] every 20 k training steps. We further apply mixup, time- and frequency-warping data augmentation [29] as well as structured patchout [23] to counteract memorization of oversampled clips.

To explore the impact of different degrees of balancing on the training process we conduct experiments with global balancing and linearly interpolated oversampling according to Eq. (4) with different $\alpha \in \{0.0, 10^{-3.5}, 10^{-3}, 10^{-2.5}, 10^{-2}\}$. We choose the linear interpolation approach here as it does not overly increase the dataset size when considering strong balancing as discussed in Sec. II-A. The same set of experiments could similarly be conducted using the balancing exponent from Eq. (3) for interpolation. Mean average precision (mAP) is used as performance metric as is common for audio tagging on AudioSet.

In the left plot of Fig. 3, we report, for each α, the mAP performance on the evaluation set for checkpoints every 20 k training steps, i.e., after each lr annealing cycle. Each training was repeated on each of the three cross validation folds and averages and standard deviations of mAP values are reported. We further show, for each α, the performance obtained by using stochastic weight averaging (SWA) of multiple checkpoints as an asterisk of the same color. The checkpoints used for SWA were determined for α=$\{10^{-2.5}, 10^{-3}, 10^{-3.5}, 0.0\}$ by finding the training steps giving on average the best validation mAP, which are steps 100 k, 200 k, 280 k and 420 k, respectively, and selecting the respective 3, 5, 7, and 9 checkpoints centered around it for SWA. We chose larger number of checkpoints for smaller α to take advantage of their flatter performance curves.

It can be seen that stronger balancing drastically reduces the required training time. Here, a mAP of 46.5% can be achieved with only 120 k training steps and SWA when using $\alpha = 10^{-2.5}$ (red asterisk), which poses and important finding along the lines of recent

[1] https://huggingface.co/timm/deit3_base_patch16_384.fb_in22k_ft_in1k

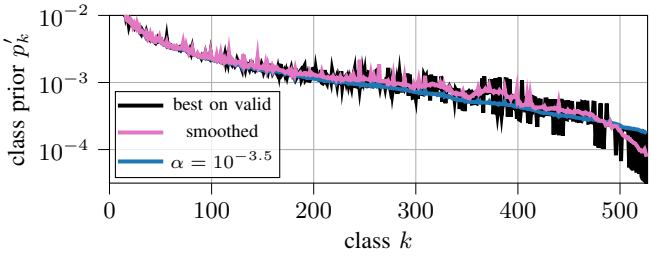

Fig. 4. Best class priors (on valid. set) vs. priors for $\alpha = 10^{-3.5}$ (Eq. (4)).

TABLE I
EVALUATION RESULTS FOR RE-TRAINING ON ALL TRAINING DATA AFTER HYPER-PARAMETERS WERE TUNED USING HELD-OUT VALIDATION DATA.

Oversampling type	$\alpha = 10^{-3.5}$ (Eq. (4))	$\hat{r}_k$ (Eq. (8))	$\beta = 0.2$ (Eq. (3))
mAP	47.76 %	47.94 %	48.20 %

efforts to balance performance versus energy consumption [30], [31]. For best performance, however, only light balancing and longer training appears better. Using $\alpha = 10^{-3.5}$ with 360 k training steps and SWA achieves 47.1% mAP (dark blue asterisk).

We now evaluate if and how the ideal degree of balancing varies across classes. For that, we derive for each α an SWA model from the same range of training checkpoints, namely the 7 checkpoints centered around step 280 k. From these models, each corresponding to different α, we then select for each sound class individually the model that gives best validation AP performance for that class, eventually giving us "ideal" $\hat{\alpha}_k$ and $\hat{p}'_k$ for each class k, for the considered SWA range. The corresponding average performance gain by this selection is $+1.0\%$pt. and $+0.21\%$pt. mAP on the validation and evaluation sets, respectively, suggesting that the selection may not generalize very well. The resulting $\hat{p}'_k$ are shown in Fig. 4.

Considering the pink curve ("smoothed"), which is a moving average of the raw black curve ("best on valid"), and comparing it to the distribution obtained for $\alpha = 10^{-3.5}$ (blue curve), it can be observed that rare classes tend to favor lighter balancing, with the tuned class priors falling below the blue curve, whereas the more prevalent classes tend to favor stronger balancing, i.e., tuned class priors higher than the blue line. This suggests that, when using stronger oversampling for the rare classes, the model may tend to overfit to these classes earlier than for the more prevalent classes.

To investigate whether we can train a better single model with these supposedly better class priors, we derive oversampling rates $\hat{r}_k$ that yield class priors $\hat{p}'_k$ by solving the optimization problem in Eq. (8). We further notice that the shape of the pink curve is well matched by the class distribution from Fig. 1 when using a balancing exponent of $\beta = 0.2$ (dark blue curve). This may indicate that the interpolation method from Eq. (3) may be better suited in this case when applying only light balancing.

Therefore, we compare model trainings using the tuned $\hat{r}_k$, linearly interpolated oversampling with $\alpha = 10^{-3.5}$, and interpolated oversampling with a balancing exponent of $\beta = 0.2$, respectively, where we train on the whole training data now without held-out validation set. For each, we show performance of SWA model (averaging 7 checkpoints centered around training step 280 k) in Table I. While the tuned $\hat{r}_k$ improve performance only insignificantly ($+0.18\%$pt) over using $\alpha = 10^{-3.5}$, the oversampling interpolation with a balancing exponent $\beta = 0.2$ improves mAP by $+0.44\%$pt. Note, however, that for stronger balancing it is still advisable to use linear interpolation to avoid scaling up the dataset size by orders of magnitude.

We turn now to the third question above, namely whether we can use local balancing for improved data loading efficiency instead of global balancing. To answer it, we run the same set of trainings we conducted previously to investigate training convergence (left subplot of Fig. 3) but use local balancing instead as described in Section II-C. We again derived checkpoints for SWA from validation results. Evaluation results are shown in the middle plot of Fig. 3. It can be seen that, for light balancing rates (green and blue curves), the local balancing approach does achieve results similar to or slightly better than global balancing. For stronger balancing (red curve), however, performance deteriorates. With clips (especially from rare classes) then being repeated many times, it makes intuitive sense that clustered repetitions will perform worse than scattering a large number of duplicates across the whole epoch. As we previously found that light balancing does give better results though, local balancing presents a good option for improving data loading efficiency.

Finally, as many recent state-of-the-art models are obtained by fine-tuning a pre-trained self-supervised model [5], [20], [22], we investigate how balancing affects the convergence of a model during fine-tuning. To do so, we consider the fine-tuning of the BEATs model [5], which we initialize with the pre-trained "BEATS_iter3" checkpoint[2]. The BEATs architecture is also Transformer-based, matching ViT-B in terms of number of Transformer layers and layer widths, but it does differ in some details. For example, it does not use pre-norm and uses a convolutional positional embedding. Its log-mel spectrogram input feature map uses a short-time Fourier transform (STFT) window size and shift of 25 ms and 10 ms, respectively, and 128 mel filters up to a maximum frequency of 8 kHz. Further, the patch size and stride are both 16×16 (non-overlapping patches). We use the same classification head as with ViT-B here.

We again run trainings for different oversampling interpolations α and present results in the right plot of Fig. 3. Note that, due to an increased training time, we only run a single training for each α with one of the validation folds. As before, checkpoints for SWA are determined on validation results. While performance obtained with $\alpha = 0.0$ and $\alpha = 10^{-3.5}$ are similar to our previous results, it can be seen that we see significantly higher performance when using stronger balancing with $\alpha \geq 10^{-3}$. We argue that this is due to the fact that, with longer training, the model diverges more from the pre-trained model, thereby losing the advantage of the pre-training. When performing fine-tuning with $\alpha = 10^{-2}$ (red curve) on the whole training set, i.e., without held-out validation set, and performing SWA on the same range of checkpoints as determined beforehand with held-out validation set, a mAP of 48.15% is achieved.

IV. CONCLUSIONS

In this paper, we investigated different aspects of class balancing for audio tagging. We showed that, when training state-of-the-art AST models on AudioSet, best performance can be achieved by only employing light oversampling of rare classes. However, stronger balancing can significantly speed up training convergence thus saving development time and energy consumption. We further found it advantageous to not overly oversample rare classes, as it possibly leads to early overfitting for these classes. Moreover, we proposed a data-loading-efficient local balancing scheme that yields similar or even better performance than conventional balancing for light to medium balancing rates. Finally, we found that for fine-tuning of pre-trained self-supervised models, stronger balancing and shorter training time gives superior performance.

[2]https://github.com/microsoft/unilm/tree/master/beats

REFERENCES

[1] J. F. Gemmeke, D. P. 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. ICASSP*, 2017, pp. 776–780.

[2] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "CNN architectures for large-scale audio classification," in *Proc. ICASSP*, 2017, pp. 131–135.

[3] 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 Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 2880–2894, 2020.

[4] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio spectrogram transformer," in *Proc. Interspeech*, 2021.

[5] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, W. Che, X. Yu, and F. Wei, "BEATs: audio pre-training with acoustic tokenizers," in *Proc. ICML*, 2023, pp. 5178–5193.

[6] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proc. ICLR*, 2018.

[7] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019.

[8] F. Schmid, K. Koutini, and G. Widmer, "Efficient large-scale audio tagging via transformer-to-cnn knowledge distillation," in *Proc. ICASSP*, 2023.

[9] H. Dinkel, Y. Wang, Z. Yan, J. Zhang, and Y. Wang, "CED: Consistent ensemble distillation for audio tagging," in *Proc. ICASSP*, 2024, pp. 291–295.

[10] Y. Gong, Y.-A. Chung, and J. Glass, "PSLA: Improving audio tagging with pretraining, sampling, labeling, and aggregation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 3292–3306, 2021.

[11] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, 2009.

[12] L. Shen, Z. Lin, and Q. Huang, "Relay backpropagation for effective learning of deep convolutional neural networks," in *Proc. ECCV*, 2016, pp. 467–482.

[13] Y. Gao, X. Bu, Y. Hu, H. Shen, T. Bai, X. Li, and S. Wen, "Solution for large-scale hierarchical object detection datasets with incomplete annotation and data imbalance," *arXiv preprint arXiv:1810.06208*, 2018.

[14] C. J. Steinmetz and J. D. Reiss, "WaveBeat: End-to-end beat and downbeat tracking in the time domain," in *Audio Engineering Society Convention 151*, 2021.

[15] F. Pishdadian, G. Wichern, and J. Le Roux, "Finding strength in weakness: Learning to separate sounds with weak supervision," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 2386–2399, 2020.

[16] I.-Y. Jeong and H. Lim, "Audio tagging system using densely connected convolutional networks." in *Proc. DCASE*, 2018, pp. 197–201.

[17] R. C. Moore, D. P. Ellis, E. Fonseca, S. Hershey, A. Jansen, and M. Plakal, "Dataset balancing can hurt model performance," in *Proc. ICASSP*, 2023, pp. 1–5.

[18] Y. Gong, C.-I. Lai, Y.-A. Chung, and J. Glass, "SSAST: Self-supervised audio spectrogram transformer," in *Proc. AAAI*, vol. 36, no. 10, 2022, pp. 10 699–10 709.

[19] A. Baade, P. Peng, and D. Harwath, "MAE-AST: Masked autoencoding audio spectrogram transformer," in *Proc. Interspeech*, 2022.

[20] P.-Y. Huang, H. Xu, J. Li, A. Baevski, M. Auli, W. Galuba, F. Metze, and C. Feichtenhofer, "Masked autoencoders that listen," *Advances in Neural Information Processing Systems*, vol. 35, pp. 28 708–28 720, 2022.

[21] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "Masked spectrogram modeling using masked autoencoders for learning general-purpose audio representation," in *HEAR: Holistic Evaluation of Audio Representations*, 2022, pp. 1–24.

[22] W. Chen, Y. Liang, Z. Ma, Z. Zheng, and X. Chen, "EAT: Self-supervised pre-training with efficient audio transformer," *arXiv preprint arXiv:2401.03497*, 2024.

[23] K. Koutini, J. Schlüter, H. Eghbal-Zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," *arXiv preprint arXiv:2110.05069*, 2021.

[24] H. Touvron, M. Cord, and H. Jégou, "DeiT III: Revenge of the ViT," in *Proc. ECCV*, 2022, pp. 516–533.

[25] S. Ioffe, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[26] D. Hendrycks and K. Gimpel, "Gaussian error linear units (GELUs)," *arXiv preprint arXiv:1606.08415*, 2016.

[27] I. Loshchilov, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[28] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[29] J. Ebbers and R. Haeb-Umbach, "Pre-training and self-training for sound event detection in domestic environments," in *Proc. DCASE*, 2022.

[30] R. Serizel, S. Cornell, and N. Turpault, "Performance above all? energy consumption vs. performance, a study on sound event detection with heterogeneous data," in *Proc. ICASSP*, 2023, pp. 1–5.

[31] F. Ronchini and R. Serizel, "Performance and energy balance: A comprehensive study of state-of-the-art sound event detection systems," in *Proc. ICASSP*, 2024, pp. 1096–1100.