Supplementary Material: A Sequential Self Teaching Approach for Improving Generalization in Sound Event Recognition

Anurag Kumar [1] **Vamsi Krishna Ithapu** [1]

1. Technical Results

$$\mathcal{L}(\mathbf{p}^s, \mathbf{y}^s) = \frac{1}{C}\sum_{c=1}^{C} \ell(\mathbf{p}^s, \mathbf{y}^s) \quad \text{where} \tag{1}$$

$$\ell(p_c^s, y_c^s) = -y_c^s \log(p_c^s) - (1 - y_c^s)\log(1 - p_c^s) \tag{2}$$

$$\bar{\mathbf{y}}_t^s = \alpha_0 \mathbf{y}^s + \sum_{\tilde{t}=1}^{t} \alpha_{\tilde{t}} \hat{\mathbf{p}}_{\tilde{t}-1}^s \quad s.t. \quad \sum_{\tilde{t}=0}^{t} \alpha_{\tilde{t}} = 1 \tag{3}$$

$$\bar{\mathbf{y}}_t^s = \alpha_0 \mathbf{y}^s + (1 - \alpha_0)\hat{\mathbf{p}}_{t-1}^s \tag{4}$$

$$y_c^s = \begin{cases} y_c^{*s} & w.p. \;\; \delta_c \\ 1 - y_c^{*s} & else \end{cases} \tag{5}$$

$$\bar{\mathbf{y}}_1^s = \alpha_0 \mathbf{y}^s + (1 - \alpha_0)\hat{\mathbf{p}}_0^s \tag{6}$$

$$\hat{p}_{0,c}^s = \begin{cases} y_c^{*s} & w.p. \;\; \bar{\delta}_c \\ 1 - y_c^{*s} & else \end{cases} \tag{7}$$

Proposition 1. *Let $\mathcal{N}^1$ be trained using $\{\mathbf{x}^s, \bar{\mathbf{y}}^s\}$ $\forall$ s using binary cross-entropy loss, and let ϵ_c denote the average accuracy of $\mathcal{N}^0$ for class c. Then, we have*

$$\bar{\delta}_c = \epsilon_c \delta + (1 - \epsilon_c)(1 - \delta) \; \forall \; c \tag{8}$$

and whenever $\delta < \frac{1}{2}$, $\mathcal{N}^1$ improves performance over $\mathcal{N}^0$. The per class performance gain is $(1 - \epsilon_c)(1 - 2\delta)$

Proof. Recall the entropy loss from Eq. 1, for a given s and c. Using the definition of the new label from Eq. 6, we get the following

$$\ell(p_c^s, \bar{y}_c^s) = \alpha_0 \ell(p_c^s, y_c^s) + (1 - \alpha_0)\ell(p_c^s, \hat{p}_c^s) \tag{9}$$

Now, Eq. 5 says that *w.p.* δ (recall $\delta_c = \delta \; \forall \; c$ here), $\ell(p_c^s, y_c^s) = \ell(p_c^s, y_c^{*s})$, else $\ell(p_c^s, y_c^s) = \ell(p_c^s, 1 - y_c^{*s})$. Hence, using Eq. 5 and Eq. 7, and using the resulting equations in Eq. 9 we have the following

$$\mathop{\mathbb{E}}_s \ell(p_c^s, y_c^s) = \delta \sum_{s=1}^{S} \ell(p_c^s, y_c^{*s}) + (1 - \delta)\sum_{s=1}^{S} \ell(p_c^s, 1 - y_c^{*s})$$

$$\mathop{\mathbb{E}}_s \ell(p_c^s, \hat{p}_c^s) = \bar{\delta}_c \sum_{s=1}^{S} \ell(p_c^s, y_c^{*s}) + (1 - \bar{\delta}_c)\sum_{s=1}^{S} \ell(p_c^s, 1 - y_c^{*s})$$

$$\mathop{\mathbb{E}}_s \ell(p_c^s, \bar{y}_c^s) = (\alpha_0 \delta + (1 - \alpha_0)\bar{\delta}_c)\sum_{s=1}^{S} \ell(p_c^s, y_c^{*s})$$
$$+ (\alpha_0(1 - \delta) + (1 - \alpha_0)(1 - \bar{\delta}_c))\sum_{s=1}^{S} \ell(p_c^s, 1 - y_c^{*s})$$

If $(\alpha_0 \delta + (1 - \alpha_0)\bar{\delta}_c) > \delta$ then we can ensure that using $\bar{y}_c^s$ as targets is better than using y_c^s. Now given the accuracy of $\mathcal{N}^0$ denoted by ϵ_c $\forall$ c, combining Eq. 5 and Eq. 7, we can see that $\bar{\delta}_c = \epsilon_c \delta + (1 - \epsilon_c)(1 - \delta)$. Using this, for $\mathcal{N}^1$ to be better than $\mathcal{N}^0$, we need

$$\alpha_0 \delta + (1 - \alpha_0)(\epsilon_c \delta + (1 - \epsilon_c)(1 - \delta)) > \delta \tag{10}$$

which requires $\delta < \frac{1}{2}$. And the gain is simply $\alpha_0 \delta + (1 - \alpha_0)\bar{\delta}_c - \delta$ which reduces to $(1 - \epsilon_c)(1 - 2\delta)$. □

Corollary 1. *Let ϵ_c^t denote the accuracy of $\mathcal{N}^t$ for class c. Given some δ, there exists an optimal $\bar{T}^c$ such that $\epsilon_c^{\bar{T}^c} \geq \epsilon_c^t$.*

Proof. When $\delta > \frac{1}{2}$, Eq. 10 will not hold, and Proposition 1 says that $\mathcal{N}^1$ is worse than $\mathcal{N}^0$. Hence $\bar{T}^c = 1$. On the other hand, if $\delta < \frac{1}{2}$, then $\bar{\delta}_c > \delta$, and the performance improves. For the given c, one can repeat the analysis for next stages with different values of $\bar{\delta}_c$. $\bar{T}_c$ is the stage t where the corresponding $\bar{\delta}_c$ increases over $\frac{1}{2}$. □

2. WEANET^L for FSDKaggle-2019

Table 1 shows the $WEANET$ architecture used for experiments on FSDKaggle-2019 dataset. $WEANET^L$ is just a lighter version of the one shown in Table 1 in the main paper. To keep things simple, we also use a simpler parameter-free

[1]Facebook Reality Labs, Redmond, USA. Correspondence to: Anurag Kumar <anuragkr@fb.com>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

Stage	Layers	Output Size
Input	Unless specified – (S)tride = 1, (P)adding = 1	$1 \times 1024 \times 64$
Block B1	Conv: 64, 3×3 Conv: 64, 3×3 Pool: 4×4 (S:4)	$64 \times 1024 \times 64$ $64 \times 1024 \times 64$ $64 \times 256 \times 16$
Block B2	Conv: 128, 3×3 Conv: 128, 3×3 Pool: 2×2 (S:2)	$128 \times 256 \times 16$ $128 \times 256 \times 16$ $128 \times 128 \times 8$
Block B3	Conv: 256, 3×3 Conv: 256, 3×3 Pool: 2×2 (S:2)	$256 \times 128 \times 8$ $256 \times 128 \times 8$ $256 \times 64 \times 4$
Block B4	Conv: 256, 3×3 Conv: 256, 3×3 Pool: 2×2 (S:2)	$256 \times 64 \times 4$ $256 \times 64 \times 4$ $256 \times 32 \times 2$
Block B5	Conv: 512, 3×2 (P:0)	$512 \times 30 \times 1$
Block B6	Conv: C, 1×1	$C \times 30 \times 1$
$g()$	Global Average Pooling	$C \times 1$

Table 1. Model architecture for $WEANET^L$ for FSDKaggle-2019 dataset: All convolutional layers (except B6) are followed by batch norm and ReLU; B6 is followed by sigmoid activation.

mapping function $g()$. We use global average pooling as $g()$, which takes an average of segment level outputs to produce recording level output.

3. Class-wise performance for Audioset

Figure 1 shows class-wise performance for different sound classes and the improvement obtained from the sequential self-teaching approach. The blue bar shows performance obtained from base-model (a.k.a default teacher $\mathcal{N}^0$). The green or red bar shows the change in performance from SUSTAIN model (corresponding to $\mathcal{N}^4$) in Table 4 from main text. The classes have been sorted by change in performance, with maximum improvement for first bar in top plot and maximum reduction in *Vibraphone* class in right most bar of bottommost plot.

We see that classes such as *Zing, Moo, Cattle, Owl, Yodeling* (first 5 bars in topmost plot), get an absolute improvement of up to 0.16 to 0.19 in MAP, leading to $40-60\%$ improvement in relative sense. As mentioned in the main text, there are few classes such as *Mouse, Squeal, Rattle* for which performance improves by more than 100%. Overall, *Bagpipes* sounds are easiest to recognize and we achieve an AP of 0.931 for it. *Squish* on the other hand is hardest to recognize with an AP of 0.02.

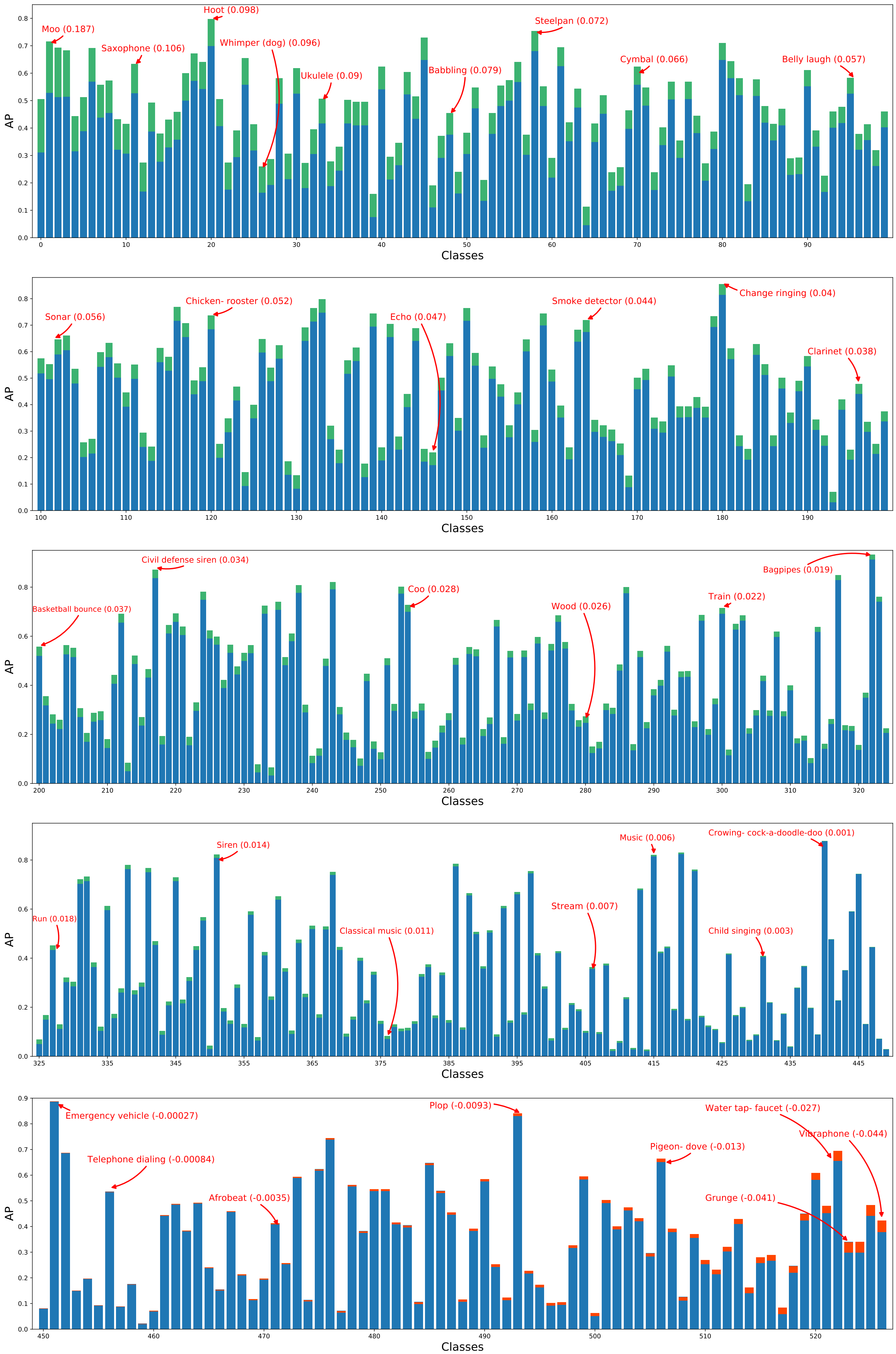

Figure 1. **Audioset** Class-wise AP and improvement in AP from SUSTAIN. The blue bar shows performance of $\mathcal{N}^0$, i.e. model trained only on available labels. The bar on top of each blue bar shows improvement (green) or deterioration (red) in performance from sequential teaching. Several classes (along with **absolute change** in performance) have been annotated to bring out noteworthy observations.