Active Few-Shot Learning for Sound Event Detection

Yu Wang[1], *Mark Cartwright*[2], *Juan Pablo Bello*[1]

[1]Music and Audio Research Laboratory, New York University, NY, USA
[2]New Jersey Institute of Technology, Newark, NJ, USA

wangyu@nyu.edu, mark.cartwright@njit.edu, jpbello@nyu.edu

Abstract

Few-shot learning has shown promising results in sound event detection where the model can learn to recognize novel classes assuming a few labeled examples (typically five) are available at inference time. Most research studies simulate this process by sampling support examples randomly and uniformly from all test data with the target class label. However, in many real-world scenarios, users might not even have five examples at hand or these examples may be from a limited context and not representative, resulting in model performance lower than expected. In this work, we relax these assumptions, and to recover model performance, we propose to use active learning techniques to efficiently sample additional informative support examples at inference time. We developed a novel dataset simulating the long-term temporal characteristics of sound events in real-world environmental soundscapes. Then we ran a series of experiments with this dataset to explore the modeling and sampling choices that arise when combining few-shot learning and active learning, including different training schemes, sampling strategies, models, and temporal windows in sampling.

Index Terms: sound event detection, few-shot learning, active learning

1. Introduction

Few-shot learning [1–5] has recently been proposed for sound event detection [6] and shown promising results, where a model is trained to learn to recognize novel sound classes, unseen during training, given only very few examples from each new class at inference time. It has been applied to tasks in various audio domains, including speech, music, and environmental sound, tackling the labeled data scarcity issue by incorporating minimal human input [7–13].

One of the main assumptions of few-shot learning is that human users can provide a few examples (e.g. five) of the target novel class, which we call the support set, at inference time. In research studies, this process is typically simulated by sampling the support set randomly and uniformly from all available test data with the target class label. However, this assumption and simulation might not reflect real-world sound event detection scenarios. Often, obtaining a few representative audio examples is not as straightforward as one might expect. For example, in the case of environmental sound monitoring, when a user encounters a new sound class that they want the model to learn to recognize, they might not be able to obtain five examples right away and/or the examples they can obtain may be from a limited context and not representative. Similarly, in the case of automatic drum transcription, obtaining five representative examples may be difficult if the target drum class is sparse or highly varying within the song. Therefore, model performance in these real-world scenarios may be lower than the model performance reported in research papers.

This work was partially supported by National Science Foundation award 1544753 & 1955357.

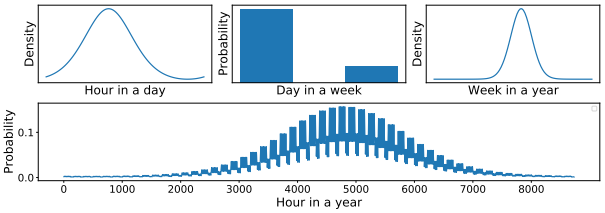

Figure 1: *Simulated occurrence probabilities of a foreground sound class in the SONYC-FSD-SED dataset.*

In this work, we show that while model performance drops when we relax these assumptions to reflect real-world scenarios, we can not only recover but supersede prior model performance by combining few-shot learning with active learning, where the model actively queries human users for labels of the most informative unlabeled data that improves model performance [14]. In prior work, active learning has been applied to various audio classification tasks and shown to be effective in improving training efficiency and reducing annotation effort [15–21]. By introducing of active learning into few-shot learning, we aim to efficiently build a better and more representative support set.

To understand how to effectively combine few-shot learning and active learning, we developed a novel dataset SONYC-FSD-SED, which will be freely available online, and designed a series of experiments using this dataset to explore the modeling and sampling choices that arise when combining these techniques. We pursue this work in the context of environmental sound monitoring. SONYC-FSD-SED simulates the long-term temporal characteristics of sound events in a real-world environmental sound monitoring system with ground-truth labels. It reflects the seasonal periodic patterns of the occurrences and co-occurrences of sound classes. In the experiments, we first explore different training schemes for the few-shot model. While a standard few-shot model is typically trained and tested with a fixed number of random support examples (i.e., fixed number of shots), during the active sampling process, the number of support examples is constantly changing. We propose new training schemes to better match the diversity and the varied number of support examples during these inference time scenarios. Second, we experiment with different few-shot models to see how they interact with active learning and affect sampling. Lastly, we study how different temporal windows in sampling affect model performance and generalization, mimicking the scenario where users only have access to data with limited variability and diversity. While in this work we perform experiments in the context of environmental sound monitoring, we expect our experimental design and findings should be generally applicable to sound event detection tasks in other audio domains such as music and speech.

2. The SONYC-FSD-SED dataset

To study our research questions in a realistic setup, we create the SONYC-FSD-SED dataset that simulates audio data in an environmental sound monitoring system, where sound class occurrences and co-occurrences exhibit seasonal periodic patterns. We use recordings collected from the Sound of New York City (SONYC) acoustic sensor network [22] as backgrounds, and single-labeled clips in the FSD50K dataset [23] as foreground events. Instead of sampling foreground sound events uniformly, we simulate the occurrence probability of each class at different times in a year, creating more realistic temporal characteristics.

We first pick a sensor from the SONYC sensor network and subsample from recordings it collected within a year (2017). We categorize these ∼550k 10-second clips into 96 bins based on timestamps, where each bin represents a unique combination of the *month of a year*, *day of a week* (weekday or weekend), and *time of a day* (divided into four 6-hour blocks). Next, we run a pre-trained urban sound event classifier [24] over all recordings and filter out clips with active sound classes. We do not filter out *footstep* and *bird* since they appear too frequently, instead, we remove these two classes from the foreground sound material. Then from each bin, we choose the clip with the lowest sound pressure level, yielding 96 background clips.

For foreground sound events, we follow the same filtering process as in [25] to get the subset of FSD50K with short single-labeled clips.We remove the two classes that exist in our backgrounds from the total 89 classes as mentioned earlier, and partition the remaining classes into disjoint train, validation, and test splits with a 43:14:30 ratio. For each class, we model its occurrence probability within a year. We use von Mises probability density functions to simulate the probability distribution over different weeks in a year and hours in a day considering their cyclic characteristics: $f(x|\mu,\kappa) = e^{\kappa cos(x-\mu)}/2\pi I_0(\kappa)$, where $I_0(\kappa)$ is the modified Bessel function of order 0, μ and $1/\kappa$ are analogous to the mean and variance in the normal distribution. We randomly sample (μ_{year}, μ_{day}) from $[-\pi, \pi]$ and $(\kappa_{year}, \kappa_{day})$ from $[0, 10]$. We also randomly assign $p_{weekday} \in [0, 1]$, $p_{weekend} = 1 - p_{weekday}$ to simulate the probability distribution over different days in a week. Finally, we get the probability distribution over the entire year with a 1-hour resolution as shown in Figure 1. At a given timestamp, we integrate f_{year} and f_{day} over the 1-hour window and multiply them together with $p_{weekday}$ or $p_{weekend}$ depends on the day. To speed up the following sampling process, we scale the final probability distribution using a temperature parameter randomly sampled from $[2, 3]$.

Lastly, we sample and mix the background and foreground sound events into 10-sec soundscapes using Scaper [26]. At a timestamp t in a year, we pick the corresponding background clip, sample foreground classes based on the simulated $p(t)$ per class, and pick one clip per sampled class. Each sampled sound event is randomly pitch-shifted within ± 2 semitones and time-stretched by a ratio in [0.8, 1.2], The clips are then randomly placed in the background and mixed with an SNR randomly sampled from [-5, 20] dB. We limit the number of classes in a soundscape to be in $[1, 5]$. We take the 43 training classes as foreground sound classes to build the training set, and run the sampling and mixing process over a year multiple times to get at least 20 soundscapes at each timestamp. For validation and test sets, we change foreground sound classes accordingly and make sure we get at least one soundscape at each timestamp. The resulting dataset spans a simulated year and contains 335k training, 69k validation, and 62k test 10-sec soundscapes.

3. Experimental design

3.1. Prototypical networks

We use prototypical networks [3] as the few-shot model in this work. Prototypical networks have been found to perform well on several few-shot audio-related tasks [8–11, 27]. They aim to learn a discriminative feature space in which a prototype representation can be computed for a novel class by averaging the feature vectors of a few novel examples. Classification is then performed for an embedded query point by simply finding the nearest class prototype based on squared Euclidean distance.

We train a basic prototypical network using the SONYC-FSD-SED training set via *episodic training* [3] with *10-way 5-shot* classification tasks [10, 11]. In each training iteration, a training episode is formed by randomly selecting 10 classes from the training set. For each selected class, we sample 5 samples into the support set and another 16 samples for computing the classification loss [5]. By training with a large collection of episodes, the model learns class-agnostic ability to learn from limited labeled data. We use the pre-trained CNN-based OpenL3 audio subnetwork [28] (with fixed wights) with an additional fully-connected layer as the backbone, which exhibited better performance compared to training the original 4-layer CNN from scratch in our preliminary experiments.

3.2. Exp. 1: Comparing sampling strategies at inference

With a trained prototypical network, we can detect a novel sound event at test time by formulating it as a binary classification problem, providing a few target examples to compute a positive prototype, and model the negative prototype using random examples from available data as proposed in the previous work [10]. However, this assumes we have access to a few target examples, which might not be the case in many real-world scenarios. In this work, we relax this assumption by starting from just one random target example, mimicking the scenario when a user first recognizes a new sound event of interest, and 100 random examples from the training set as initial negative support examples. Then, the goal is to update the support set by asking few more labels from the users in an efficient way. To do so, we propose to incorporate active learning techniques, specifically, the uncertainty-based sampling [14], where we use the trained few-shot model to find the most uncertain example from the entire test set and query for its label. We simulate the human labeling process by directly using the ground-truth label of the queried example. We run this sampling process 100 times. At each iteration, we choose the example with the predicted probability closest to 0.5, add it to the support set to update the corresponding prototype, and report the resulting model performance on the test data. For comparison, we also run the same pipeline but sample additional examples using random sampling. Note that annotating 100 times would require much less human effort than finding 100 positive examples.

3.3. Exp. 2: Comparing prototypical net. training schemes

The main motivation of the episodic training technique is to match training and testing scenarios. It has been shown that matching the number of shots n between training and testing a few-shot model results in better performance [3, 10]. That is, if we plan to test the model with 5 shots (5 examples per novel class), it is ideal to train the model with 5-shot episodes. However, in our setup, at test time, we start with only one target example ($n = 1$) and iteratively sample additional examples with increasing n, which does not match our standard training

setup. Therefore, we propose two other training schemes that further match training and testing scenarios.

First, instead of training with fixed 5-shot, we train with random shots. In each training episode, we randomly choose n from $[1, 20]$ and build the support set accordingly. By doing so, the model can learn to work with different numbers of support examples as in different sampling stages at inference time. We refer to this training scheme as *random-shot training*.

Next, we go a step further to match the entire active sampling process. In each training episode, we start from $n = 1$ and select an additional 100 samples per class to form a pool with 10×100 total samples. Then, we run 100 active sampling iterations to sample the most uncertain examples from the pool to update the support set. Here we measure uncertainty using the best-vs-second-best strategy [29] designed for multi-class setup, which considers the difference between the probability values of the top two predicted classes. We compute loss and gradient at each sampling iteration and perform one back-propagation at the end of iteration 100 based on the accumulated gradient. We call this training scheme *active-shot training*.

3.4. Exp. 3: Comparing different sampling windows

Thus far our experiments have assumed having access to the entire test set at inference time to sample addition support examples. However, we also want to explore scenarios where we only have limited access to the data, for example when we do not save all historical data collected by an acoustic sensor, and observe how this affects model performance and generalizability. To do so, in this set of experiments, we generate an additional 2-years of test data following the same process in Section 2, and consider the updated test set as data in the *past year*, *current year*, and *future year*. For each test class, we find the peak position of its simulated occurrence probability distribution in the current year. Then, we define 5 temporal windows, *2-week*, *1-month*, *3-month*, *6-month*, and *1-year* to sample additional support examples. We start from a 2-week window centered around the peak timestamp, and expand it on one side along the direction to the past, simulating scenarios in which we have varying access to the historical data. Note that the initial support example is always sampled from the 2-week window. Finally, we compute performance on the data a year before and after the current peak timestamp, to see how the model generalizes to data in the past (which we have varying access to) and future.

3.5. Exp. 4: Comparing few-shot to logistic regression

Recent works have shown that a simple logistic regression (LR) model on top of a pre-trained embedding model outperforms few-shot algorithms when $n > 10$ on audio classification task [25]. In addition to the prototypical networks, we experiment with this transfer learning approach to compare their behavior with active sampling. We train the same backbone architecture on the same training set via standard supervised learning. At test time, for each novel class, we embed the support examples via the trained embedding model and use them to directly train a binary LR model. During active sampling, we add the sampled examples to the support set to retrain the LR model.

4. Results

4.1. Training schemes and inference sampling strategies

We first compare prototypical networks with three different training schemes: *5-shot*, *random-shot*, and *active-shot*. At test time, we run each model with two different sampling strategies: *random* and *active* sampling, to sample additional examples to update corresponding prototypes. Figure 2 shows how model performance and sampling efficiency change with sampling iteration at inference time. We report the mean F-measure for performance and mean total positives for sampling efficiency, averaged over 30 test classes. We also show the model performance when we have five random positives as assumed in the typical few-shot setup as a baseline (red dotted line).

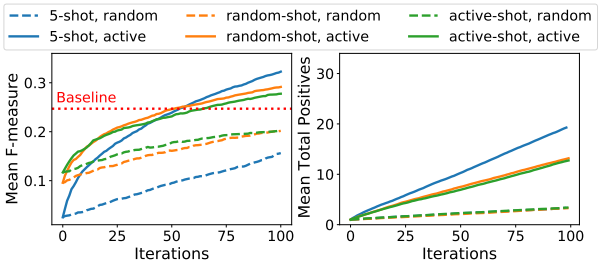

Figure 2: *Mean F-measure (left) and total positive examples (right) at each sampling iteration for prototypical networks with different training schemes and inference sampling strategies.*

First, the right plot of Figure 2 shows that random sampling only returns less than five positives after 100 iterations while active sampling returns more positives at a faster rate. Meanwhile, the left plot of Figure 2 shows that for all three models, using active sampling consistently achieves better performance at all iterations. In addition, we see performance drops from the baseline on models with random sampling, as a result of relaxing the standard few-shot assumption. With active sampling, we are able to bring the performance towards the baseline and even go beyond it with increasing sampling iterations. This indicates that the uncertainty-based sampling strategy is effective at sampling informative examples more efficiently and forming representative prototypes for few-shot models.

Next, we compare model performance when fixing sampling strategy to random sampling. We see that the random-shot model performs significantly better than the 5-shot model, while the active-shot model improves even further. This shows that in the regime with very few positive examples, matching the setup with varying numbers of shots at training time is critical for model performance. And on top of that, training with more informative support examples sampled with the uncertainty sampling strategy results in an additional performance boost.

Lastly, we look at model performance with active sampling. The random-shot model outperforms the 5-shot model until around iteration 60 when more positive examples are labeled. Compared to the random-shot model, active-shot model achieves slightly higher performance in the first 20 iterations and falls behind afterward. Note that we consistently observe this trade-off (including the preliminary experiments) between performance in few-shot and mid-shot regimes. No one model dominates both regimes and there is always a cross-point in performance. As part of future work, we plan to investigate further in the learned embedding space to understand this behavior.

Considering that performance in the few-shot regime is particularly important, since in real use cases, users might not want to annotate up to 100 iterations, together with the longer train time for the active-shot model, we fix the training scheme for prototypical networks to random-shot in the rest of this work.

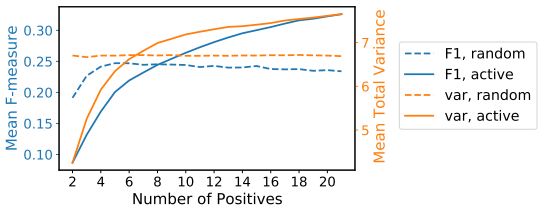

Figure 3: *Mean F-measure for prototypical networks and mean total variance of positive examples sampled with different sampling strategies.*

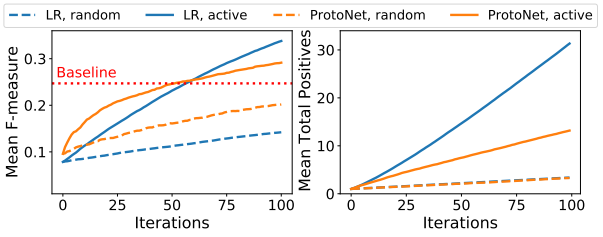

Figure 4: *Mean F-measure (left) and total positive examples (right) at each sampling iteration for prototypical networks and LR model with different inference sampling strategies.*

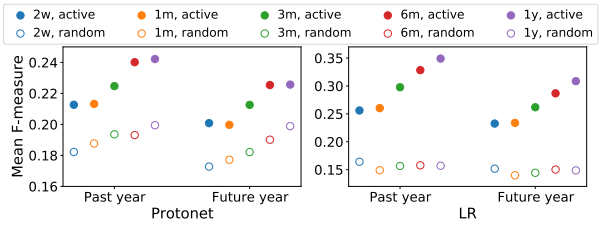

Figure 5: *Mean F-measures computed on past and future test data with different sampling strategies and sampling windows for prototypical networks (left) and LR model (right).*

4.2. Support set diversity

We have shown that it is more efficient to sample the support set using active sampling. While active sampling gets more "informative" samples according to the uncertainty heuristic, these samples are not necessarily the most diverse, especially in early iterations when starting with just one positive example. To better understand the relationship between support set diversity and model performance, we look at model performance with different numbers of positive support examples sampled using both random and active sampling, and we measure the diversity of the sampled examples by their mean total variance.

The result in Figure 3 shows a strong correlation between model performance and support examples diversity, implying that few-shot models can benefit from a diverse support set. Random sampling shows consistent example diversity across different numbers of examples. Active sampling starts from lower example diversity which then increases with the increasing number of examples. This indicates that, as future work, it is worth exploring other active learning sampling strategies which explicitly take the example diversity into account to improve the support set diversity and thus the following few-shot model performance in the few-shot regime. Note that the result in Figure 3 does not imply that we should favor random sampling in the few-shot regime since these two sampling strategies require different numbers of sampling iterations to get the same number of positives. It takes a long time to get even just a few positives using random sampling as shown in Figure 2.

4.3. Prototypical networks v.s. logistic regression

Next, we compare prototypical networks with the LR model. The results in Figure 4 show that with random sampling, prototypical networks consistently achieve better performance. While with active sampling, prototypical networks dominate until around iteration 60, at which point the LR model takes over given more positive examples.

Previous work [25] found a similar result that the algorithm specifically designed for few-shot learning outperforms other approaches in the standard few-shot regime while simple transfer learning approach becomes more effective when the number of support examples increases. This trend also matches our previous result that no one model dominates between few-shot and mid-shot regimes. For real-world applications, we can choose one model over the other based on desired annotation effort or design a model switching mechanism to take advantage of both models in different regimes.

4.4. Different sampling windows

Figure 5 shows how different sampling windows affect model performance and generalizability. To get more evident trends, we look at model performance after 100 sampling iterations. The results show two general trends. First, given a model, a sampling strategy, and a sampling window, the model performs better on data in the past than on data in the future. This matches our intuition that the model generalizes better to data it has access to (at least part of it) during the sampling process.

Second, given a model, a sampling strategy, and a test data set, the larger temporal window of sampling results in higher performance (except for the LR model with random sampling). However, there are two competing factors here. A smaller sampling window likely has less diverse examples but a higher ratio of positive examples since we start the sampling window around the mode of the prior distribution of each test class. The results in Figure 5 indicate that these two competing factors both affect the LR model while example diversity might play a more important role for prototypical networks.

5. Conclusions

In this work, we relax a common assumption in few-shot sound event detection that a handful of examples, sampled from the entire test set, are available at test time. We consider a realistic setup starting from only one support example, and propose to incorporate active learning techniques to efficiently sample more examples to update the support set. We show that uncertainty-based active sampling expands the support set more efficiently and achieves better few-shot model performance compared to random sampling. It also returns more diverse examples with increasing sampling iteration. Next, we propose new training schemes for the few-shot model to address varying number of support examples, and show how models trained with different schemes perform differently in few-shot and mid-shot regimes. Lastly, we show that a larger temporal window of sampling results in better few-shot model performance, and the model generalizes better on historical data, which it has access to during sampling, than the future data.

6. References

[1] G. R. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML Workshop*, 2015.

[2] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems 29*, 2016, pp. 3630–3638.

[3] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems 30*, 2017, pp. 4077–4087.

[4] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 1199–1208.

[5] W. Chen, Y. Liu, Z. Kira, Y. F. Wang, and J. Huang, "A closer look at few-shot classification," in *International Conference on Learning Representations*, 2019.

[6] A. Mesaros, T. Heittola, T. Virtanen, and M. D. Plumbley, "Sound event detection: A tutorial," *IEEE Signal Processing Magazine*, vol. 38, no. 5, pp. 67–83, 2021.

[7] K. Cheng, S. Chou, and Y. Yang, "Multi-label few-shot learning for sound event recognition," in *IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, 2019, pp. 1–5.

[8] S. Chou, K. Cheng, J. R. Jang, and Y. Yang, "Learning to match transient sound events using attentional similarity for few-shot sound recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 26–30.

[9] B. Shi, M. Sun, K. C. Puvvada, C. Kao, S. Matsoukas, and C. Wang, "Few-shot acoustic event detection via meta learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 76–80.

[10] Y. Wang, J. Salamon, N. J. Bryan, and J. P. Bello, "Few-shot sound event detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 81–85.

[11] Y. Wang, J. Salamon, M. Cartwright, N. J. Bryan, and J. P. Bello, "Few-shot drum transcription in polyphonic music," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[12] Y. Wang, N. J. Bryan, M. Cartwright, J. P. Bello, and J. Salamon, "Few-shot continual learning for audio classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[13] H. F. Garcia, A. Aguilar, E. Manilow, and B. Pardo, "Leveraging hierarchical structures for few-shot musical instrument recognition," in *International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[14] B. Settles, "Active learning literature survey," University of Wisconsin–Madison, Computer Sciences Technical Report 1648, 2009.

[15] D. Hakkani-Tür, G. Riccardi, and A. Gorin, "Active learning for automatic speech recognition," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2002, pp. IV–3904–IV–3907.

[16] D. Yu, B. Varadarajan, L. Deng, and A. Acero, "Active learning and semi-supervised learning for speech recognition: A unified framework using the global entropy reduction maximization criterion," *Computer Speech & Language*, vol. 24, no. 3, pp. 433 – 444, 2010.

[17] W. Han, E. Coutinho, H. Ruan, H. Li, B. Schuller, X. Yu, and X. Zhu, "Semi-Supervised Active Learning for Sound Classification in Hybrid Learning Environments." *PLoS ONE*, vol. 11, no. 9, pp. 1–23, 2016.

[18] Z. Shuyang, T. Heittola, and T. Virtanen, "Active learning for sound event classification by clustering unlabeled data," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 751–755.

[19] K. Qian, Z. Zhang, A. Baird, and B. Schuller, "Active learning for bird sound classification via a kernel-based extreme learning machine," *The Journal of the Acoustical Society of America*, vol. 142, no. 4, pp. 1796–1804, 2017.

[20] B. Kim and B. Pardo, "A human-in-the-loop system for sound event detection and annotation," *ACM Trans. Interact. Intell. Syst.*, vol. 8, no. 2, pp. 13:1–13:23, 2018.

[21] Y. Wang, M. A. E. M., M. Cartwright, and J. P. Bello, "Active learning for efficient audio annotation and classification with a large amount of unlabeled data," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 880–884.

[22] J. P. Bello, C. T. Silva, O. Nov, R. L. DuBois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "SONYC: A system for monitoring, analyzing, and mitigating urban noise pollution," *Commun. ACM*, vol. 62, no. 2, p. 68–77, 2019.

[23] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50k: an open dataset of human-labeled sound events," *arXiv:2010.00475*, 2020.

[24] M. Cartwright, J. Cramer, A. Mendez, Y. Wang, H. Wu, V. Lostanlen, M. Fuentes, G. Dove, C. Mydlarz, J. Salamon, O. Nov, and J. Bello, "SONYC-UST-V2: An urban sound tagging dataset with spatiotemporal context." in *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2020.

[25] Y. Wang, N. J. Bryan, J. Salamon, M. Cartwright, and J. P. Bello, "Who calls the shots? rethinking few-shot learning for audio," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2021.

[26] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 344–348.

[27] J. Pons, J. Serrà, and X. Serra, "Training Neural Audio Classifiers with Few Data," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 16–20.

[28] J. Cramer, H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3852–3856.

[29] A. J. Joshi, F. Porikli, and N. Papanikolopoulos, "Multi-class active learning for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2372–2379.