SoundCount: Sound Counting from Raw Audio with Dyadic Decomposition Neural Network

Yuhang He[1], **Zhuangzhuang Dai**[2], **Niki Trigoni**[1], **Long Chen**[3,4*], **Andrew Markham** [1]

[1]Department of Computer Science, University of Oxford, UK. yuhang.he@cs.ox.ac.uk
[2]Department of Applied AI and Robotics, Aston University. UK
[3]Institute of Automation, Chinese Academy of Sciences, China.
[4] WAYTOUS Ltd., China.

Abstract

In this paper, we study an underexplored, yet important and challenging problem: counting the number of distinct sounds in raw audio characterized by a high degree of polyphonicity. We do so by systematically proposing a novel end-to-end trainable neural network (which we call DyDecNet, consisting of a dyadic decomposition front-end and backbone network), and quantifying the difficulty level of counting depending on sound polyphonicity. The dyadic decomposition front-end progressively decomposes the raw waveform dyadically along the frequency axis to obtain time-frequency representation in multi-stage, coarse-to-fine manner. Each intermediate waveform convolved by a parent filter is further processed by a pair of child filters that evenly split the parent filter's carried frequency response, with the higher-half child filter encoding the *detail* and lower-half child filter encoding the *approximation*. We further introduce an energy gain normalization to normalize sound loudness variance and spectrum overlap, and apply it to each intermediate parent waveform before feeding it to the two child filters. To better quantify sound counting difficulty level, we further design three polyphony-aware metrics: *polyphony ratio*, *max polyphony* and *mean polyphony*. We test DyDecNet on various datasets to show its superiority.

Introduction

Suppose you went to the seaside and heard a cacophony of seagulls, squawking and squabbling. An interesting question that naturally arises is whether you can tell the number of seagulls flocking around you from the sound you heard? Although a trivial example, this sound "crowd counting" problem has a number of important applications. For example, passive acoustic monitoring is widely used to record sounds in natural habitats, which provides measures of ecosystem diversity and density (Aguiar et al. 2021; Dohi et al. 2021; Chronister et al. 2021). Sound counting helps to quantify and map sound pollution by counting the number of individual polluting events (Bello, Mydlarz, and Salamon 2018). It can also be used in music content analysis (J. Humphrey, Durand, and McFee 2018). Despite its importance, research on sound counting has far lagged behind than its well-established crowd counting counterparts from either images (Zhang et al. 2016; Wang et al. 2019), video (Li, Zhang, and Chen 2018) or joint audio-visual (Hu et al. 2020).

We conjecture the lack of exploration stems from three main factors. First, sound counting has long been treated as an over-solved problem by sound event detection (SED) methods (Mesaros et al. 2021; Cakir et al. 2017; Adavanne, Pertilä, and Virtanen 2017; He, Trigoni, and Markham 2021), in which SED goes further to identify each sound event's (*e.g.* a bird call) start time, end time and semantic identity. Sound counting number then becomes easily accessible by simply adding up all detected events. Secondly, current SED only tags whether a class of sound event is present within a window, regardless of the number of concurrent sound sources of the same class like a series of baby crying or multiple bird calls (Phan et al. 2022). Thirdly, labelling acoustic data is technically-harder and more time-consuming than labelling images, due to the overlap of concurrent and diverse sources. The lack of well-labelled sound data in crowded sound scenes naturally hampers research progress. Existing SED sound datasets (Adavanne, Pertilä, and Virtanen 2017; Heittola et al. 2010) capture simple acoustic scenarios with low polyphony and where the event variance is small. The simplified acoustic scenario in turn makes sound counting task by SED methods tackleable. But when the sound scene becomes more complex with highly concurrent sound events, SED methods soon lose their capability in discriminating different sound events (Pankajakshan, Bear, and Benetos 2019; Cakir et al. 2017). In the meantime, some researchers think sound counting is equivalent to sound source separation task (Neumann et al. 2020; Turpault et al. 2021; Tzinis, Wang, and Smaragdis 2022; Subakan et al. 2022; Tzinis et al. 2020), in which the sound is counted as the source number by isolating individual sound from sound mixture and assigning it to corresponding sound source. However, our proposed sound counting is different from source number counting, it directly counts the overlapping events number, regardless of if these events come from the same sound source. Therefore, a study specific for sound counting problem is desirable and overdue.

In this paper, we study the general sound counting problem under highly polyphonic, cluttered and concurrent situations. Whilst the challenges of image-based crowd counting mainly lie in spatial density, occlusion and view perspective distortion, the sound counting challenges are two-fold.

*corresponding author.

Firstly, acoustic scenes are additive mixtures of sound along both time and frequency axes, making counting overlapping sounds difficult (temporal concurrence and spectrum-overlap). Secondly, there is a large variance in event loudness due to spherical signal attenuation with distance.

To tackle these challenges, we propose a novel dyadic decomposition neural network to learn a sound density representation capable of estimating cardinality directly from raw sound waveform. Unlike existing sound waveform processing methods that all apply frequency-selective filters on the raw waveform in single stage (He, Trigoni, and Markham 2021; Cao et al. 2021; Zeghidour et al. 2021; He and Markham 2022; Davis and Mermelstein 1980), our network progressively decomposes raw sound waveform in a dyadic manner, where the intermediate waveform convolved by each parent filter is further processed by its two child filters. The two child filters evenly split the parent filter's frequency response, with one child filter encoding the waveform *approximation* (the one with the lower-half frequency response) and the other one encoding the waveform *details* (the one with the higher-half frequency response). To accommodate sound loudness variance, spectrum-overlap and time-concurrence, we further propose an energy gain normalization module to regularize each intermediate parent waveform before feeding it to two child filters for further processing. This hierarchical dyadic decomposition front-end enables the neural network to learn a robust TF representation in multi-stage coarse-to-fine manner, while introducing negligible extra computation cost. By setting each filter's frequency cutoff parameters to be learnable and self-adjustable during optimization in a data-driven way, the final learned TF representation can better characterize sound existence in time and frequency domain. Following the front-end, we add a backbone network to continue to learn a time framewise representation. Such representation can be used to derive the final sound count number by either directly regressing the count number, regressing density map (the one we choose) or following SED pipeline. Apart from the network, we further propose three polyphony-aware metrics to quantify sound counting task difficulty level: polyphony ratio, maximum polyphony and mean polyphony. We will give detailed discussion to show the feasibility of three metrics.

We run experiment on large amounts of sound datasets, including commonly heard bioacoustic, indoor and outdoor, real-world and synthetic sound. Comprehensive experimental results show the superiority of our proposed framework in counting under different challenging acoustic scenarios. We further show our proposed dyadic decomposition front-end can be used to tackle other acoustic task, like SELD (Adavanne, Pertilä, and Virtanen 2017; He, Trigoni, and Markham 2021). In summary, we make three main contributions: **First**, propose dyadic decomposition front-end to decompose the raw waveform in a multi-stage, coarse-to-fine manner, which better handles loudness variance, spectrum-overlap and time-concurrence. **Second**, propose a new set of polyphony-aware evaluation metrics to comprehensively and objectively quantify sound counting difficulty level. **Third**, show DyDecNet superiority on various counting datasets, and its potential to be used as a general learnable TF extraction front-end.

Dyadic Decomposition Neural Network

Different sound classes typically exhibit different spectral properties. A canonical way to process raw sound waveform is to apply a frequency-selective filter bank $\boldsymbol{\mathcal{F}}_f = \{f_i\}_{i=1}^k$ to project the raw sound waveform onto different frequency bins. Traditional Fourier transform (Davis and Mermelstein 1980) or Wavelet transform (Mallat 2008) construct fixed filter banks in which all filter-construction relevant hyperparameters are empirically chosen and thus may not be optimal for a particular task. Recent methods (He, Trigoni, and Markham 2021; Zeghidour et al. 2021) relax some hyperparameters to be trainable so that the filter bank can be optimized in a data-driven way. A learnable filter bank often leads to better performance than fixed filters. However, all existing methods apply all filters, either learnable or fixed, on the raw waveform in a one-stage manner. Such shallow and one-stage processing may fail to learn powerful and robust representation for sound counting task where large loudness variance and heavy spectrum overlap exist. In our dyadic decomposition framework, we instead adopt a progressive pairwise decomposition strategy to obtain the time-frequency (TF) representation. It learns a TF representation from coarse to fine-grained granularity. Particularly, it consists of a dyadic frontend and a backbone.

Dyadic Frequency Decomposition Frontend

In dyadic decomposition frontend, we construct a set of D hierarchical filter banks $\boldsymbol{\mathcal{F}}_{dyadic}^{D} = \{\boldsymbol{\mathcal{F}}_{2^1}^{1}, \boldsymbol{\mathcal{F}}_{2^2}^{2}, \cdots, \boldsymbol{\mathcal{F}}_{2^D}^{D}\}$. The d-th filter bank has 2^d filters, each filter is parameterized by a learnable high frequency-cutoff parameter and a low frequency-cutoff parameter. By cascading these filter banks, we consecutively decompose the raw waveform in frequency domain dyadically, leading to coarse-grained to fine-grained TF representation. Specifically, we denote the dyadic filter banks depth by D, in the depth d filter bank $\boldsymbol{\mathcal{F}}_{2^d}^{d}$, we have 2^d filters evenly divide the waveform sampling frequency F_s. Therefore, each single filter's frequency response length is $\frac{F_s}{2^d}$, the i-th filter f_i^d high frequency cutoff F_h and low frequency cutoff F_l are initialized as,

$$F_h(f_i^d) = \frac{F_s}{2^d} \cdot (i+1), \quad F_l(f_i^d) = \frac{F_s}{2^d} \cdot i \tag{1}$$

From Eqn. (1) we can see that dyadic decomposition frontend forms a complete binary-tree-like structure, in which the filter number doubles and each filter's frequency response length halves as the tree's depth increases by one. The intermediate waveform processed by a "parent" filter is just further processed by its two "children" filters. The frequency responses of the two children filters evenly split their parent filter's frequency response. The child filter carrying the higher half frequency response encode the parent's processed intermediate waveform's *detail* while the other one carrying the lower half frequency response instead encodes the *approximation*. For example, for the filter f_i^d in the d-th filter bank, its frequency response lies in $[\frac{F_s}{2^d} \cdot i, \frac{F_s}{2^d} \cdot (i+1)]$, its two children filters f_{2i}^{d+1} and f_{2i+1}^{d+1} in

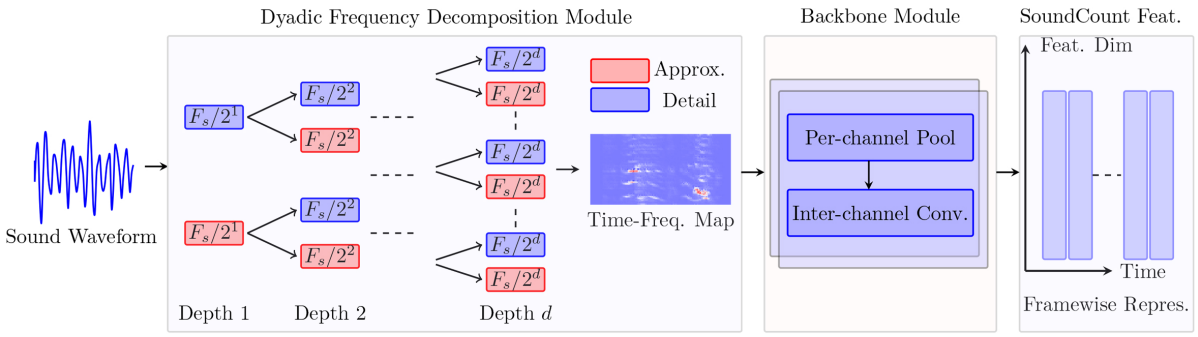

Figure 1: DyDecNet pipeline. We first feed the input raw sound waveform to the dyadic decomposition front-end to learn a time-frequency representation, which is further fed to a backbone neural network to continue to learn framewise representation. Such representation retains time information, so it is general enough to get count number by either regression or SED method. The dyadic decomposition front-end consists of a set of parameterized learnable band-pass filters. Each intermediate waveform processed by a parent filter is further processed by two child filters, with lower-half filter (red color) encoding *approximation* and higher-half filter (light-blue) encoding *details*.

the depth $d+1$ evenly divide its frequency range, so f_{2i}^{d+1} carries $[\frac{F_s}{2^d}\cdot i, \frac{F_s}{2^d}(i+\frac{1}{2})]$. f_{2i+1}^{d+1} carries $[\frac{F_s}{2^d}(i+\frac{1}{2}), \frac{F_s}{2^d}\cdot(i+1)]$.

With the pre-constructed dyadic decomposition filter banks, we cascade them together to process the raw sound waveform, progressively learning the final TF representation. In our implementation, each filter in dyadic filter banks is a learnable band-pass filter. We adopt rectangular band-pass in frequency domain filter which comprises of a learnable high frequency cutoff parameter F_h and a learnable low frequency cutoff parameter F_l. Converting it to time domain through the inverse Fourier transform, we get $sinc(\cdot)$ function like filter that is used to convolve with the waveform. For example, the filter f_i^d in Eqn. (1) is represented as,

$$f_i^d[t, F_h, F_l] = 2F_h sinc(2\pi F_h t) - 2F_l sinc(2\pi F_l t) \quad (2)$$

where $sinc(x) = sin(x)/x$, t indicates the filter's representation at time t. F_h and F_l are initialized according to Eqn. (1), but they can be further adjusted during the training process. $sinc(\cdot)$ filters have been successfully used in speech recognition (Ravanelli and Bengio 2018) and sound event detection and localization (He, Trigoni, and Markham 2021). In our dyadic decomposition frontend, each filter from different depth has separate and independent learnable parameters (high frequency cutoff and low frequency cutoff). Moreover, our constructed filter is much longer (1025 in our case) than traditional 1D/2D Conv filters (3 or 5). Its wide length characteristic enables the filter to have a wide field-of-view on the raw waveform. Cascading them together allows the filters in later layers (larger depth) to have an even wider field-of-view on the input raw waveform. With this advantage, we do not have to model sound event temporal dependency explicitly with RNN network. As a result, the whole dyadic frequency decomposition frontend is fully convolutional and parametrically learnable, it is parameter-frugal and computationally efficient. In practice, the dyadic decomposition frontend depth is 8, so the output TF representation has 256 frequency bins. At the same time, we downsample the intermediate waveform by 2 before feeding it to its two children filters in the initial 5 dyadic filter banks to reduce the memory cost.

Energy Gain Normalization

We further design an energy gain normalization module to regularize each intermediate waveform before feeding them to the next dyadic filter bank. The motivation of introducing energy gain normalization is two-fold: first, to reduce sound event loudness variance led by sound events' different spatial locations; Second, to reinforce the frontend to learn to better tackle spectrum overlap challenge led by intra-class sound events in the sound scene. Specifically, for the intermediate waveform $W_{f_i^d}$ processed by a dyadic filter f_i^d, we first smooth it with a learnable 1D Gaussian kernel g_i^d parameterized by learnable width σ to get the corresponding smoothed waveform $W_{g_i^d}$ which just contains loudness. We then introduce a learnable automatic gain control parameter α to mitigate sound loudness impact. Furthermore, another two learnable compression parameters δ and γ are introduced to further compress $W_{f_i^d}$. The overall energy gain normalization can be represented as,

$$W_{f_i^d} = (\frac{W_{f_i^d}}{(W_{g_i^d})^\alpha} + \delta)^\gamma - \delta^\gamma \quad (3)$$

where α, δ and γ are learnable parameters. As a result, the energy gain normalization *eg*-Norm is fully learnable and parameterized by four learnable parameters *eg*-Norm$(\sigma, \alpha, \delta, \gamma)$. Practically, each filter in dyadic filter banks is associated with an independent *eg*-Norm module. Similar energy normalization has been successfully used in tasks like keyword spotting (Wang et al. 2017; Lostanlen et al. 2019). The difference lies in the fact that they apply exponential moving average operation to get smoothed waveform representation, so the computation is very slow because it iterates along the time axis to compute the averaged value step by step. Our proposed energy gain nor-

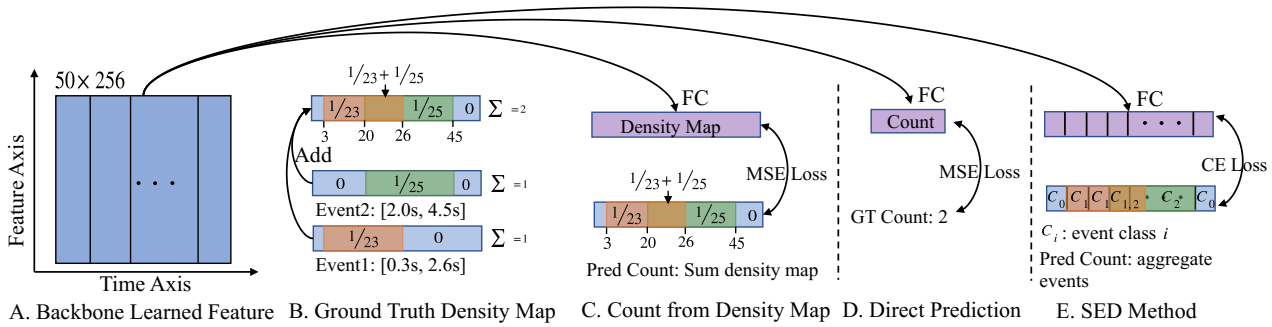

Figure 2: Three counting methods illustration. For density map (sub-fig. C), the sum (or integral) of the density map equals to the count number. We can also direct regress the final count number (sub-fig. D), or use SED method (sub-fig. E).

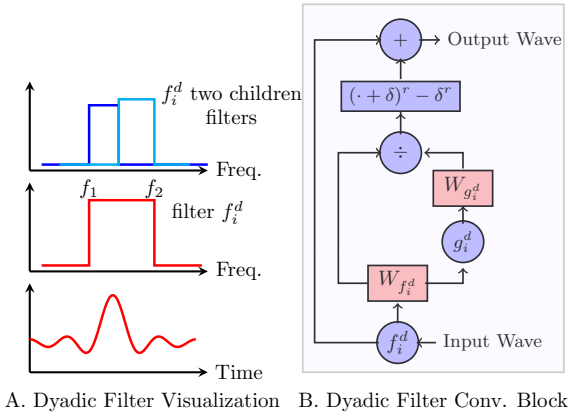

Figure 3: Dyadic filter illustration. Left: In time domain, dyadic filter is a $sinc$ function curve. In frequency domain, dyadic filter is a rectangular band-pass filter with learnable high frequency f_2 and low frequency cutoff f_1. The filter's two child filters (left-top) evenly splits the parent filter's frequency response. Right: dyadic filter convolution block. The input waveform is fed to an energy normalization module. Then a skip-connection is added.

malized strategy instead adopts a Gaussian kernel to get the smoothed waveform, in which it can be easily implemented as 1D convolution. The dyadic filter visualization and energy normalization module is shown in Fig. 3.

Backbone Neural Network

We add a lightweight backbone neural network to the frontend neural network to further learn a representation useful for call counting . The backbone network consists of two parts: per-channel pooling and inter-channel 1D convolution. Unlike existing methods (Cakir et al. 2017; Adavanne, Pertilä, and Virtanen 2017) that first convert 1D sound waveform into 2D map with fixed FFT-like transform, then learn from the 2D map with 2D Conv. operations, our method directly learns from sound raw waveform with learnable 1D Conv.. Specifically, we downsample each channel separately by assigning each channel with an independent frequency-sensitive learnable filter. We call such learnable downsampling per-channel pooling. It helps to learn sound event's frequency variance along the time axis individually. Moreover, we add normal 1D Conv. to achieve inter-channel communication, which enhances the neural network to learn concurrent sound events interaction. The backbone serves as the backend to learn framewise representation for counting.

Density Map and Loss Function

The backbone network discussed above learns a framewise representation $[T_b, F_b]$, where T_b indicates the time steps and F_b indicates feature size. There are three potential ways to derive the final sound count number from the learned representation: 1. directly regress the count number; 2.SED method: detect sound events first and then aggregate results to get final count; 3. predict the density map. For a sound event with time location $[t_1, t_2]$, its density map is a 1D vector with value $\frac{1}{t_2-t_1}$ during its occurrence time, otherwise it is 0. So the count number equals the vector integral. We thus adopt the mean squared error (**MSE**) loss during training to directly regress the density map. The comparison of three methods is shown in Fig. 2.

Counting Difficulty Quantification

Mean absolute error (**MAE**) and mean squared error (**MSE**) are two widely used metrics in crowd counting (Loy et al. 2013; Zhang et al. 2016). Specifically, denote the ground truth count and predicted count by y_i and $\hat{y}_i$ respectively, for the i-th sound clip. MAE is defined as $\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$, MSE is defined as $\text{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$. We also involve accuracy rate (**AccuRate**) to show the ratio of accurately predicted count. We introduce a tolerance term p, where $p = 0$ means the predicted count number has to be exactly the same with ground truth number in order to be treated as an accurate counting; $p = 1$ relaxes the constraint so there can be one count mismatch for an accurate counting.

The aforementioned three general metrics do not reflect the impact of sound scene nature on algorithms. We introduce three polyphony-aware metrics to quantify the sound counting difficulty level reflected by the sound scene nature. The three metrics are time-window invariant so they can be

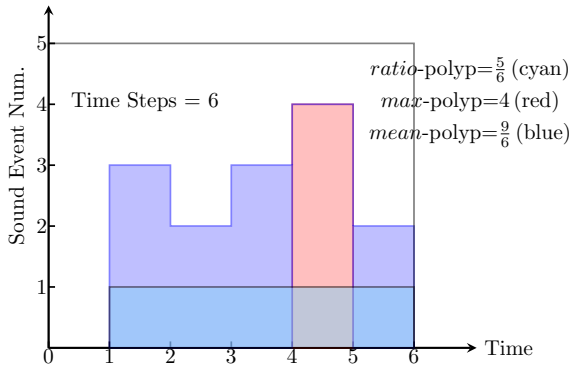

Figure 4: The max polyphony level in this sound clip is 4 (show in red, time step 5), so max-polyp=4. The $mean$-polyp indicates the purple area, averaging them over time gets $\frac{9}{6}$. $ratio$-polyp measures polyphony (no fewer than two concurrent sound events) existence ratio along the time axis (cyan), so it is $\frac{5}{6}$.

used as general metrics to quantify difficulty level of sound scene of various lengths.

Polyphony Ratio ($ratio$-polyp) describes the ratio of polyphony (at least two sound events happen at the same time) over a period of time. It binarizes each time step as either polyphonic or non-polyphonic (monophoinc or silent) so the value lies between $[0, 1]$.

Maximum Polyphony (max-polyp) focuses on the maximum polyphony level over a time period. It is motivated by the fact that human's capability in discriminating different sound events reduces seriously when the number of temporal-overlapping sound event number increases. It is a positive integer and helps us to understand an algorithm's capability in tackling polyphony peak.

Mean Polyphony ($mean$-polyp) instead focuses on the averaging level of polyphony involved within a time period. It is designed to reflect an algorithm's capability in tackling the average polyphony level over an arbitrary time window.

Given T_n time steps sound vector $[p_1, p_2, \cdots, p_{T_n}]$, where $p_i \geq 0$ is the sound event number happening at time step T_i. The three metrics are defined as,

$$ratio\text{-polyp} = \frac{\sum_{i=1}^{n} \mathbb{1}_2(p_i)}{n}; max\text{-polyp} = \max_{i=1,\cdots,n} p_i;$$
$$mean\text{-polyp} = \frac{\sum_{i=1}^{n} \max(p_i - 1, 0)}{T_n} \tag{4}$$

where $\mathbb{1}_2(p_i)$ is an indicator function, it is 1 if $p_i \geq 2$, otherwise 0. With the three metrics, we can report the general metrics (MAE, MSE) against various difficulty levels.

Experiment

We run experiments on five main datasets.

1. **Bioacoustic Sound**. We focus on bird sound as bird sound is ubiquitous in most terrestrial environments with distinctive vocal acoustic properties. Specifically, we test three datasets: one real-world NorthEastUS (Chronister et al. 2021) dataset and other two synthesized datasets: Polyphony4Birds (for heterophony test) and Polyphony1Bird (for homophony test). NorthEastUS data is recorded in nature reserve in northeastern United States. It encompasses 385 minutes of dawn chorus recordings collected in July 2018, with a total of 48 bird species. The average bird sound temporal length is very short (less than 1s) and the polyphony level (max-polyp and $mean$-polyp) is small. To test performance under highly polyphonic situations, we synthesize two bird sound datasets. Specifically, The first dataset contains four sounds: junco, American redhead, eagle, and rooster from copyright-free website `findsounds.com`. We call it Polyphony4Birds (heterophony test). The second dataset contains one sound: rooster. We call it Polyphony1Bird (homophony test).

2. **Indoor Sound**. We count telephone ring sound, the telephone ring seed sound comes from the same copyright-free website. We follow Polyphony1Bird synthesis procedure except that the room size is much smaller ($10m \times 10m \times 3m$) to reflect indoor reverberation effect.

3. **Outdoor Sound**. We count car engine, as it is widely heard in outdoor scenario. The car engine seed sound comes from the same copyright-free website. We follow Polyphony1Bird synthesis procedure to create the dataset.

4. **AudioSet** AudioSet (Gemmeke et al. 2017) is a large temporally-strong labelled dataset with a wide range of sound event classes, including music, speech and water. The AudioSet data tests all methods' capability in counting under large different event classes scenario. Specifically, we train model on the train dataset which has 103,463 audio clips and 934,821 labels, and test the model on the evaluation which has 16,996 audio clips and 139,538 labels. In total there are 456 sound event categories.

5. **Music Sound**. We use OpenMic2018 dataset (J. Humphrey, Durand, and McFee 2018) to count musical instruments.

Comparing Methods: We compare three main method categories: 1) traditional signal processing methods: Librosa-onset and Aubio-onset; 2) three SED-based methods. 3) one sound source separation method. **Librosa-onset** (McFee et al. 2015) provides an onset/offset detection method for music note detection. It measures the uplift or shift of spectral energy to decide the starting time of a note. We use its onset/offset detection ability to count sound events. **Aubio-onset** (Brossier 2006) achieves pitch tracking by aligning period and phase. We use pitch tracking to count.

SED-based methods build on traditional fixed TF representation, such as short time Fourier transform (STFT) and LogMel. The TF representation is treated as a 2D image to be processed by a sequence of 2D Conv. operators. GRU (Chung et al. 2014) and LSTM (Hochreiter and Jugen 1997) are often adopted to model temporal dependency. We compare three typical SED methods: 1) **CRNNNet** (Cakir et al. 2017) consists of 2D Conv. to learn multiple compressed TF representations from the input TF map. Then it concatenates them together along the frequency dimension and further feeds it to LSTM (Hochreiter and Jugen 1997)

Method	AudioSet		NorthEastUS		Polyp4Birds		Polyp1Bird		TelepRing		CarEngine	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Librosa-onset	26.9	4.80	2.31	1.65	28.3	4.09	37.63	5.5	30.03	4.50	33.13	4.51
Aubio-onset	8.50	1.98	4.91	1.74	8.43	1.91	35.33	5.27	33.20	4.22	35.13	4.76
SELDNet (Grondin et al. 2019)	0.93	1.28	1.35	1.79	0.92	1.41	0.89	1.19	0.97	1.30	0.92	1.23
CRNNNet (Cakir et al. 2017)	0.92	1.07	1.33	1.77	0.74	1.10	0.87	1.16	0.92	1.31	0.86	1.15
DND-SED (Drossos et al. 2020)	1.10	1.22	1.19	1.64	0.95	1.34	1.04	1.27	1.23	1.34	1.00	1.21
DPTasNet (Neumann et al. 2020)	0.81	1.02	1.11	1.60	0.97	1.47	1.22	1.43	1.47	1.21	0.89	1.11
Ours DyDecNet	**0.32**	**0.73**	**1.01**	**1.19**	**0.46**	**0.92**	**0.54**	**0.85**	**0.58**	**0.89**	**0.54**	**0.87**

Table 1: MSE (↓) and MAE (↓) on the five main category sound (six in total) datasets.

to learn framewise representation. 2) **DND-SED** (Drossos et al. 2020) instead adopts depthwise 2D convolution and dilated convolution to avoid using RNN. 3) **SELDNet** (Adavanne, Pertilä, and Virtanen 2017) is originally used for joint sound event detection and localization. It adopts 2D Conv. to convolve the 2D TF map, and bidirectional GRU to model temporal dependency. The three comparing methods' network architectures are slightly adjusted to fit our dataset. For sound source separation method, we adopt **DPTasNet** (Neumann et al. 2020), in which it trains a Dual-Path RNN (DPRNN) and TasNet to jointly separate each sound event and further count the event number. In this case, we treat each sound event as independent sound sources.

Implementation Detail For all datasets, all input audios are segmented into 5 second long clips, with sampling rate 24 kHz. So the input waveform has 120,000 data points and is normalized into $[-1, 1]$. We train the models with Pytorch (Paszke et al. 2019) on TITAN RTX GPU. To train the neural network, we adopt Adam optimizer (Kingma and Ba 2015) with an initial learning rate 0.001 which decays every 20 epochs with a decaying rate 0.5. Overall, we train 60 epochs. We train each method 10 times independently and report the mean value and standard deviation. We do not report the standard deviation explicitly in the table because we find them very small (about 0.03). We first train the comparing SED methods with both their suggested training strategy and our training strategy, then choose the one with the better performance as the final result. For the energy gain normalization we initialize them as $\alpha = 0.96$, $\delta = 2.$, $\gamma = 0.5$, $\sigma = 0.5$. The batchsize is 128.

Experimental Result

The quantitative result on MSE/MAE is given in Table 1, we can learn that DyDecNet outperforms both classic signal processing deterministic methods, comparing SED methods and sound source separation based method by a large margin, under all acoustic scenarios. DyDecNet outperforms all comparing methods in both real-world and synthesized sound datasets. It is capable of learning powerful representation from both weak sound signals (NorthEastUS), highly polyphonic (synthesized datasets) and heavy spectrum-overlapping, loudness-varying sound events. Moreover, we find DPTasNet (Neumann et al. 2020) performs worse than the three SED-based methods on the two synthesized bioacoustic datasets where high-polyphony exists, which shows source separation method is not a good counting alternative in highly polyphonic situations.

Method	MSE↓	MAE↓
SELDNet 2019	1.35	1.79
SELDNet_Dydec	**1.05**	**1.43**
CRNNNet 2017	1.33	1.77
CRNNNet_Dydec	**1.20**	**1.51**
DND-SED 2020	1.19	1.64
DND-SED_Dydec	**0.89**	**1.40**

Table 2: Ablation study on dyadic decomposition efficiency discussion: we compare existing methods with and without dyadic decomposition frontend.

Method	MSE↓	MAE↓
DyDecNet_STFT	1.35	1.51
DyDecNet_LogMel	1.33	1.50
DyDecNet_MFCC	1.32	1.49
DyDecNet_Gabor	1.33	1.48
DyDecNet	**0.85**	**1.19**

Table 3: Ablation study on traditional T-F feature for counting task: DyDecNet's dyadic decompostion frontend is replaced by various classic T-F features extractors, such STFT, LogMel, MFCC and Gabor.

At the same time, we also observe that the two signal processing deterministic methods (Librosa-onset and Aubio-onset) generate the worst result over both SED based, source separation based methods and DyDecNet. The higher the polyphony level of the dataset, the worse performance the two deterministic methods lead to. For example, in NorthEastUS dataset with a relatively smaller polyphony level, Librosa-onset and Aubio-onset generate relatively good performance with accuracy rate ($p = 1$) reaching 0.58. In our synthesized two datasets with much higher polyphony levels, however, their accuracy drops significantly to near zeros. It thus shows traditional signal processing methods do not fit for sound counting from crowded acoustic scenes.

Moreover, SED-based methods and DyDecNet produce decreasing performance from Polyphony4Birds to Polyphony1Bird and then NorthEastUS. The largest performance drop is observed on real NorthEastUS dataset, which

Method	MSE↓	MAE↓
DyDecNet_SingScale	1.22	1.43
DyDecNet_BN	1.07	1.25
DyDecNet_noNorm	1.15	1.37
DyDecNet	**0.85**	**1.19**

Table 4: Various DyDecNet variants.

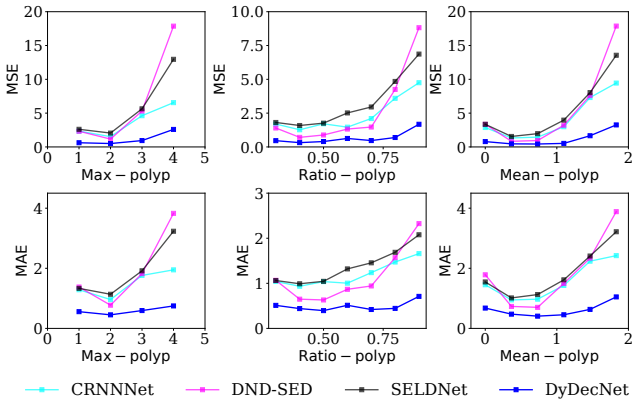

Figure 5: MSE and MAE variation against max-polyp, $ratio$-polyp and $mean$-polyp on NorthEastUS dataset.

shows counting from real-world dataset is a tough task that desires more future attention. Spectrum-overlap led by intra-class sound events is another potential challenge (better performance on Polyp4Birds than Polyp1Bird).

The MSE/MAE variation against max-polyp, $ratio$-polyp and $mean$-polyp difficulty level on NorthEastUS are shown in Fig. 5. We can observe that our proposed three metrics max-polyp, $ratio$-polyp and $mean$-polyp are effective ways to accurately quantify sound counting tasks difficulty level. The three metrics have observed dramatic performance drop as their difficulty level increases. Nevertheless, DyDecNet remains as the best one across all the three difficulty levels, showing DyDecNet outperforms the comparing methods under difficult levels discussed in this paper.

Ablation Study

We do ablation study on NorthEastUS data.

First, disentangling our proposed framework's dyadic decomposition frontend and backbone network so as to figure out their individual contribution. To this end, on the one hand, we concatenate dyadic decomposition frontend to the three SED methods backbone networks so that they can learn TF representation from raw waveform. We call them SELDNet_dydec, CRNNNet_dydec and DND-SED_dydec respectively. On the other hand, we feed our backbone neural network with fixed pre-extracted TF features, including short time Fourier transform (STFT), LogMel, MFCC and Gabor Wavelet filter. We call them DyDecNet_STFT, DyDecNet_LogMel and DyDecNet_MFCC, DyDecNet_Gabor, respectively. The results are in Table 2 and 3. We can observe that: 1) replacing traditional fixed TF feature with dyadic decomposition frontend significantly improves the performance (Table 2). The gain stems from two-fold: our dyadic decomposition frontend enables the network to directly learn from the raw waveform so that all frequency-selective filters are adjustable during training process. Second, the dyadic progressive decomposition enables the neural network to learn robust representation for sound counting. Similarly, a huge performance drop is observed if we let our proposed backbone neural network to learn from traditional fixed TF features (Table 3). Therefore, it shows that both the dyadic decomposition frontend and backbone neural networks are important for sound counting.

Method	ER (↓)	F (↑)	LE (↓)	LR (↑)
SELDNet 2017	0.63	0.46	23.1	0.69
SELDNet_DyDec	**0.60**	**0.49**	**22.7**	**0.73**
EIN 2021	0.25	0.82	8.0	0.86
EIN_DyDec	**0.21**	**0.86**	**7.4**	**0.88**
SoundDet 2021	0.25	0.81	8.3	0.82
SoundDet_DyDec	**0.21**	**0.88**	**7.2**	**0.86**
SoundDoA 2022	0.23	0.85	7.9	0.87
SoundDoA_DyDec	**0.20**	**0.89**	**7.4**	**0.89**

Table 5: Dyadic Frontend on SELD Task.

Second, we want to figure out if the dyadic decomposition is essential for sound counting, and the importance of energy normalization block. We test three variants: our network with simply single scale decomposition which means applying all filters on the raw waveform (DyDecNet_SingScale) which helps validate necessity of hierarchical dyadically decomposition framework; replacing Energy-normalization module with traditional batch normalization (Ioffe and Szegedy 2015) (DyDecNet_BN); without any normalization (DyDecNet_noNorm). The result is in Table 4, from which we can clearly observe that either removing energy normalization or replacing it with batch normalization significantly reduces the performance. It thus shows the importance of energy normalization.

Dyadic Decomposition Frontend on SELD Task

To show the dyadic decomposition front-end is a general TF feature extractor, we test it on sound event detection and localisation task (SELD). The dataset we use is TAU-NIGENS (Adavanne et al. 2018), and we compare with four main methods: SELDNet (Adavanne, Pertilä, and Virtanen 2017), EIN (Cao et al. 2021) that use classic TF feature, SoundDet (He, Trigoni, and Markham 2021) and SoundDoA (He and Markham 2022) use learnable TF-feature. We replace their time frequency (TF) extraction front-end with dyadic decomposition network front-end to see the performance change. The result is given in Table 5, we can see that dyadic decomposition front-end exhibits generalization strength to help tackle other acoustic tasks.

In summary, our proposed DyDecNet is capable of learning useful TF representation from highly polyphonic spatial. It is also a learnable general TF frontend that can be potentially used for other acoustic tasks.

References

Adavanne, S.; Pertilä, P.; and Virtanen, T. 2017. Sound Event Detection Using Spatial Features and Convolutional Recurrent Neural Network. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Adavanne, S.; Politis, A.; Nikunen, J.; and Virtanen, T. 2018. Sound Event Localization and Detection of Overlapping Sources Using Convolutional Recurrent Neural Networks. In *IEEE Journal of Selected Topics in Signal Processing*.

Aguiar, R.; Maguolo, G.; Nanni, L.; Costa, Y.; and Silla, C. 2021. On the Importance of Passive Acoustic Monitoring Filters. *Journal of Marine Science and Engineering (JMSE)*.

Bello, J. P.; Mydlarz, C.; and Salamon, J. 2018. *Sound Analysis in Smart Cities*. Springer International Publishing.

Brossier, P. 2006. *Automatic Annotation of Musical Audio for Interactive System*. Ph.D. thesis, Queen Mary University of London.

Cakir, E.; Parascandolo, G.; Heittola, T.; Huttunen, H.; and Virtanen, T. 2017. Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection. In *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*.

Cao, Y.; Iqbal, T.; Kong, Q.; Fengyan, A.; Wang, W.; and Plumbley, M. D. 2021. An Improved Event-Independent Network for Polyphonic Sound Event Localization and Detection. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Chronister, L. M.; Rhinehart, T. A.; Place, A.; and Kitzes, J. 2021. An Annotated Set of Audio Recordings of Eastern North American Birds Containing Frequency, Time, and Species Information.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical Evaluation of Gated Recurrent Neural Ntworks on Sequence Modelling. In *Advances Neural Information Processing System (NeurIPS)*.

Davis, S.; and Mermelstein, P. 1980. Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. In *IEEE Transactions on Acoustics, Speech, and Signal Processing (TASSP)*.

Dohi, K.; Endo, T.; Purohit, H.; Tanabe, R.; and Kawaguchi, Y. 2021. Flow-Based Self-Supervised Density Estimation for Anomalous Sound Detection. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Drossos, K.; Mimilakis, S. I.; Gharib, S.; Li, Y.; and Virtanen, T. 2020. Sound Event Detection with Depthwise Separable and Dilated Convolutions. In *International Joint Conference on Neural Networks (IJCNN)*.

Gemmeke, J. F.; Ellis, D. P. W.; Freedman, D.; Jansen, A.; Lawrence, W.; Moore, R. C.; Plakal, M.; and Ritter, M. 2017. Audio Set: An Ontology and Human-Labeled Dataset for Audio Events. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Grondin, F.; Glass, J.; Sobieraj, I.; and Mark D., P. 2019. A Study of the Complexity and Accuracy of Direction of Arrival Estimation Methods Based on GCC-PHAT for a Pair of Close Microphones. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop*.

He, Y.; and Markham, A. 2022. SoundDoA: Learn Sound Source Direction of Arrival and Semantics from Sound Raw Waveforms. In *Interspeech*.

He, Y.; Trigoni, N.; and Markham, A. 2021. SoundDet: Polyphonic Moving Sound Event Detection and Localization from Raw Waveform. In *International Conference on Machine Learning (ICML)*.

Heittola, T.; Mesaros, A.; Eronen, A.; and Virtanen, T. 2010. Audio Context Recognition using Audio Event Histograms. In *European Signal Processing Conference (EUSIPCO)*.

Hochreiter, S.; and Juğen, S. 1997. Long Short-Term Memory. In *Neural Computation*.

Hu, D.; Mou, L.; Wang, Q.; Gao, J.; Hua, Y.; Dou, D.; and Zhu, X. X. 2020. Ambient Sound Helps: Audiovisual Crowd Counting in Extreme Conditions. arXiv:2005.07097.

Ioffe, S.; and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *International Conference on Machine Learning (ICML)*.

J. Humphrey, E.; Durand, S.; and McFee, B. 2018. OpenMIC-2018: An Open Dataset for Multiple Instrument Recognition. In *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*.

Kingma, D.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representation (ICLR)*.

Li, Y.; Zhang, X.; and Chen, D. 2018. CSRNet: Dilated Convolutional Neural Networks for Understanding the Highly Congested Scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Lostanlen, V.; Salamon, J.; Cartwright, M.; McFee, B.; Farnsworth, A.; Kelling, S.; and Bello, J. P. 2019. Per-Channel Energy Normalization: Why and How. *IEEE Signal Processing Letters (SPL)*.

Loy, C. C.; Chen, K.; Gong, S.; and Xiang, T. 2013. Crowd Counting and Profiling: Methodology and Evaluation. In *Modeling, Simulation and Visual Analysis of Crowds*.

Mallat, S. 2008. *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*. USA: Academic Press, Inc., 3rd edition.

McFee, B.; Raffel, C.; Liang, D.; Ellis, D. P.; McVicar, M.; Battenberg, E.; and Nieto, O. 2015. librosa: Audio and Music Signal Analysis in Python. In *Proceedings of the 14th python in science conference*, volume 8.

Mesaros, A.; Heittola, T.; Virtanen, T.; and Plumbley, M. D. 2021. Sound Event Detection: A Tutorial. *IEEE Signal Processing Magazine,*.

Neumann, T. v.; Boeddeker, C.; Drude, L.; Kinoshita, K.; Delcroix, M.; Nakatani, T.; and Haeb-Umbach, R. 2020. Multi-talker ASR for an Unknown Number of Sources: Joint Training of Source Counting, Separation and ASR. In *Interspeech*.

Pankajakshan, A.; Bear, H. L.; and Benetos, E. 2019. Polyphonic Sound Event and Sound Activity Detection: A Multi-Task Approach. *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 323–327.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Phan, H.; Tho Nguyen, T. N.; Koch, P.; and Mertins, A. 2022. Polyphonic Audio Event Detection: Multi-Label or Multi-Class Multi-Task Classification Problem? In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Ravanelli, M.; and Bengio, Y. 2018. Speaker Recognition from Raw Waveform with SincNet. In *In IEEE Workshop on Spoken Language Technology (SLT)*.

Subakan, C.; Ravanelli, M.; Cornell, S.; Lepoutre, F.; and Grondin, F. 2022. Resource-Efficient Separation Transformer. arXiv:2206.09507.

Turpault, N.; Serizel, R.; Wisdom, S.; Erdogan, H.; Hershey, J. R.; Fonseca, E.; Seetharaman, P.; and Salamon, J. 2021. Sound Event Detection and Separation: A Benchmark on Desed Synthetic Soundscapes. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Tzinis, E.; Venkataramani, S.; Wang, Z.; Subakan, C.; and Smaragdis, P. 2020. Two-Step Sound Source Separation: Training On Learned Latent Targets. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Tzinis, E.; Wang, X., Zhepei amd Jiang; and Smaragdis, P. 2022. Compute and Memory Efficient Universal Sound Source Separation. In *Journal of Signal Processing Systems*.

Wang, Q.; Gao, J.; Lin, W.; and Yuan, Y. 2019. Learning from Synthetic Data for Crowd Counting in the Wild. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, Y.; Getreuer, P.; Hughes, T.; Lyon, R.; and Saurous, R. 2017. Trainable Frontend for Robust and Far-Field Keyword Spotting. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Zeghidour, N.; Teboul, O.; de Chaumont Quitry, F.; and Tagliasacchi, M. 2021. LEAF: A Learnable Frontend for Audio Classification. *International Conference on Learning Representations (ICLR)*.

Zhang, Y.; Zhou, D.; Chen, S.; Gao, S.; and Ma, Y. 2016. Single-Image Crowd Counting via Multi-Column Convolutional Neural Network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.