SOUND EVENT FILTERING AND CLASSIFICATION

Bingjie Wang
University of Rochester
bwang72@ur.rochester.edu

ABSTRACT

Real-time sound event detection (SED) requires models that are both accurate and computationally efficient while operating under strict causal constraints. However, many existing deep SED models rely on fixed-length input windows, heavy convolutional backbones, or access to future context, limiting their applicability in low-latency streaming scenarios. This project proposes a lightweight Transformer-based SED framework designed to progressively acquire real-time processing capability through training. The method introduces (1) an adaptive frame selection mechanism that adjusts temporal resolution based on short-term acoustic variation, (2) a compact Transformer encoder optimized for efficiency, and (3) a masked-frame streaming training strategy that simulates partial observability without requiring autoregressive training. Experimental results on standard SED benchmarks demonstrate that the proposed approach achieves competitive accuracy with significantly reduced latency and model size.

1. INTRODUCTION

Sound Event Detection (SED) is a fundamental component in many real-world applications, including smart sensing, robotics, augmented reality, and public safety monitoring. While offline SED systems benefit from full access to long audio recordings, real-time SED must operate under causal constraints, making predictions using only past and current audio frames. This requirement introduces challenges in balancing latency, accuracy, and computational cost.

Recent advances in deep learning, particularly convolutional neural networks (CNNs), convolutional recurrent neural networks (CRNNs), and Transformer-based models, have significantly improved SED performance. However, many of these methods assume fixed-length input windows and rely on future context, which is incompatible with real-time deployment. Furthermore, high-capacity architectures often incur prohibitive latency on resource-constrained devices.

This project explores whether real-time capability can be gradually induced through training design rather than enforced through strict online inference pipelines. We propose a lightweight Transformer-based SED model that combines adaptive temporal resolution with a masked-frame streaming training strategy. The key idea is to allow the model to learn robust sound representations under partial observability while maintaining efficient parallel training.

2. RELATED WORK

2.1 Deep Learning for Sound Event Detection

Sound Event Detection (SED) has been widely studied using deep learning models that operate on time–frequency representations such as log-mel spectrograms. Early neural approaches primarily relied on convolutional neural networks (CNNs), which learn local spectro-temporal patterns effectively and have been shown to perform well on benchmark datasets such as UrbanSound8K and ESC-50 [1–3]. However, CNN-based SED models typically employ fixed-size temporal windows and limited receptive fields, which restrict their ability to model long-range temporal dependencies.

To address this limitation, convolutional recurrent neural networks (CRNNs) were introduced, combining CNN feature extractors with recurrent layers to capture temporal context [4, 5]. While CRNNs improve temporal modeling, their recurrent nature introduces sequential dependencies that increase inference latency, making them less suitable for real-time or low-latency streaming scenarios.

More recently, Transformer-based architectures have achieved strong performance on SED by leveraging self-attention to model long-range temporal relations [6, 7]. Audio Spectrogram Transformer (AST) and its variants demonstrate that attention-based models can outperform CRNNs in offline SED settings. Nevertheless, standard Transformers assume access to the full input sequence during inference, which conflicts with the causality constraints required for real-time SED.

2.2 Real-Time and Streaming Sound Event Detection

Real-time SED introduces additional constraints beyond classification accuracy, including bounded latency, causal processing, and predictable computational cost. Early streaming-oriented approaches employed causal convolutions or recurrent models that operate frame by frame [8]. Although these methods satisfy causality, they often sacrifice modeling capacity or require carefully tuned architectures to remain efficient.

Recent studies explore adapting Transformers to streaming scenarios through techniques such as chunk-wise attention, cached key–value states, and restricted attention windows [9, 10]. While effective, these approaches typically involve non-trivial engineering complexity and may still incur significant computational overhead, especially when applied to variable-length or high-resolution audio streams. Moreover, most streaming Transformer designs rely on fixed chunk sizes, which do not account for the non-stationary nature of real-world audio signals.

2.3 Masked Training and Streaming Simulation

Masked prediction has been extensively studied in self-supervised representation learning. Methods such as wav2vec 2.0 [11] and masked autoencoders (MAE) [12] demonstrate that randomly masking portions of the input encourages models to learn robust contextual representations. Although these approaches are primarily used for pretraining, the underlying idea of restricting information access can be repurposed to simulate streaming constraints.

In the context of supervised SED, masking future frames provides a principled way to prevent information leakage while maintaining parallelizable training. Unlike autoregressive training, which enforces strict temporal ordering at the cost of efficiency, masked-frame strategies allow models to be trained using standard batch processing while still respecting causality at inference time. However, prior work has not systematically explored how masked training can be combined with adaptive temporal context selection to support efficient real-time SED.

2.4 Positioning of This Work

In contrast to existing approaches, this project focuses on *training models to acquire real-time capability* rather than deploying a fully engineered streaming system. We combine three complementary ideas: (1) a lightweight Transformer encoder for expressive temporal modeling, (2) an adaptive frame selection mechanism that adjusts temporal context based on acoustic dynamics, and (3) a masked-frame streaming training strategy that simulates temporal progression without requiring autoregressive optimization. This design allows the model to gradually learn causal behavior and efficiency-aware inference, while remaining simple enough to be implemented and evaluated within a standard PyTorch training pipeline.

3. PROPOSED METHOD

This project targets **real-time capability acquisition** rather than an end-to-end deployed streaming platform. Concretely, the goal is to train a model that can (i) make stable predictions when only a prefix of the audio has arrived, (ii) remain efficient under latency budgets, and (iii) adapt its temporal receptive field based on the current acoustic dynamics. To make the method reproducible and implementation-oriented, this section specifies the full data flow, model interfaces, and the exact training objectives used in each stage.

3.1 Problem Formulation and Streaming Constraint

Let an audio waveform be $x \in R^N$ sampled at 16 kHz. In offline SED classification, a model may access the entire clip. In streaming, at time index t the model only observes $x_{1:t}$. We represent the acoustic features as a sequence of log-mel frames $M = \{M_1, \dots, M_T\}$ with $M_t \in R^F$, where F is the number of mel bins and T is the number of frames.

The real-time constraint is expressed as a **prefix-only availability** condition:

$$\hat{y}_t = f_\theta(M_{1:k(t)}), \quad k(t) \le t, \tag{1}$$

where $k(t)$ denotes the last available frame at time t and f_θ is the proposed model. This requirement motivates two design choices: (1) the model must not implicitly rely on future frames during training, and (2) the temporal context length should be variable because acoustic complexity is not stationary over time.

3.2 Feature Extraction and Tokenization

We convert waveform to log-mel spectrogram using a standard pipeline. Let STFT window size be w samples and hop size h samples. The magnitude spectrogram is

$$S(\tau, \omega) = |\mathrm{STFT}(x; w, h)|, \tag{2}$$

and the mel projection is applied by a filter bank $W_{\mathrm{mel}} \in R^{F \times \Omega}$:

$$\mathrm{Mel}(\tau) = W_{\mathrm{mel}}\, S(\tau, :), \quad M_\tau = \log(\mathrm{Mel}(\tau) + \epsilon). \tag{3}$$

Each mel frame M_t is projected to a token embedding:

$$e_t = W_{\mathrm{in}} M_t + b_{\mathrm{in}}, \quad e_t \in R^d. \tag{4}$$

We then add positional encoding p_t (sinusoidal or learnable) to obtain the Transformer input token:

$$h_t^{(0)} = e_t + p_t. \tag{5}$$

Implementation note. For streaming compatibility, positional encodings are indexed by absolute frame index so that cached states remain consistent.

3.3 Adaptive Frame Selector as a Learnable Policy

A central limitation of fixed-window inference is that it either wastes compute in stationary segments or misses transients when windows are too long. We address this by predicting an **adaptive context length** L_t for each time step.

3.3.0.1 Acoustic change descriptor.

We compute a low-cost descriptor measuring local spectral change:

$$\Delta(t) = \|M_t - M_{t-1}\|_2. \tag{6}$$

In addition, we summarize a short recent history of changes using a smoothing filter:

$$\bar{\Delta}(t) = \frac{1}{r} \sum_{i=0}^{r-1} \Delta(t - i), \tag{7}$$

where r is a small constant (e.g., $r = 5$) to reduce sensitivity to noise.

3.3.0.2 Selector network.

Instead of a hand-crafted threshold, we use a small Transformer-based selector that reads the last u mel frames (or their embeddings) and outputs a discrete choice among candidate context lengths:

$$\mathcal{L} = \{L^{(1)}, L^{(2)}, \dots, L^{(K)}\}. \tag{8}$$

We build selector tokens from the recent prefix $M_{t-u+1:t}$ and run a 1-layer Transformer encoder to produce a selector summary vector s_t. The selector outputs logits over K choices:

$$\ell_t = W_{\text{sel}} s_t + b_{\text{sel}}, \quad \ell_t \in R^K. \tag{9}$$

We obtain a categorical distribution

$$\pi_t = \text{softmax}(\ell_t), \tag{10}$$

and choose the context length

$$L_t = \sum_{k=1}^{K} \pi_{t,k} \, L^{(k)}. \tag{11}$$

Why Transformer here. Using a Transformer in the selector allows the policy to depend on temporal structure (e.g., repeated transients, rhythm-like bursts) instead of a single scalar threshold.

3.3.0.3 Discrete selection for deployment.

For actual inference, we use the argmax choice for stability:

$$\hat{k} = \arg\max_k \pi_{t,k}, \quad L_t = L^{(\hat{k})}. \tag{12}$$

This keeps the runtime predictable because only one candidate context is processed.

3.4 Lightweight Transformer Encoder with Causal Attention

Given the chosen context length L_t, we form the model input window as the latest L_t frames:

$$H_t^{(0)} = \{h_{t-L_t+1}^{(0)}, \dots, h_t^{(0)}\}. \tag{13}$$

The encoder consists of B Transformer blocks (typically $B = 2$) with hidden size d and H heads. Each block applies causal self-attention:

$$\text{Att}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}} + M_{\text{causal}}\right) V, \tag{14}$$

where M_{causal} masks out positions $j > i$ so tokens cannot attend to the future.

The output token sequence after B layers is $H_t^{(B)}$. We pool over time using mean pooling:

$$g_t = \frac{1}{L_t} \sum_{i=1}^{L_t} H_{t,i}^{(B)}. \tag{15}$$

The classifier predicts event probabilities:

$$\hat{y}_t = \sigma(W_{\text{cls}} g_t + b_{\text{cls}}), \tag{16}$$

where $\sigma(\cdot)$ is sigmoid for multi-label SED or softmax for single-label classification.

3.5 Masked-Frame Streaming Training via Prefix Pairing

The major goal of training is to simulate the fact that time is **progressing** and future is unavailable. Rather than "data augmentation", we construct **prefix pairs** to enforce consistency under streaming.

3.5.0.1 Prefix pairing construction.

For each training clip with feature sequence $M_{1:T}$, we sample two prefix lengths $k_1 < k_2$ such that $k_2 = k_1 + \delta$ and δ is a small positive integer. We create two inputs:

$$X^{(1)} = M_{1:k_1}, \quad X^{(2)} = M_{1:k_2}. \tag{17}$$

This explicitly models "a little more time has passed".

3.5.0.2 Masked tail to prevent leakage.

To ensure the model cannot accidentally rely on frames beyond the prefix in batch training, we pad the remainder with a learned MASK token:

$$\tilde{M}_t^{(i)} = \begin{cases} M_t, & t \le k_i, \\ \text{MASK}, & t > k_i, \end{cases} \quad i \in \{1, 2\}. \tag{18}$$

This preserves parallel computation while enforcing partial observability.

3.5.0.3 Supervised objective.

Let y be the ground-truth label(s) for the clip or segment. We compute supervised losses on both prefixes:

$$\mathcal{L}_{\text{sup}} = \text{CE}(\hat{y}^{(1)}, y) + \text{CE}(\hat{y}^{(2)}, y), \tag{19}$$

where CE denotes cross-entropy or binary cross-entropy depending on label type.

3.5.0.4 Streaming consistency objective.

A core streaming requirement is that the prediction should not oscillate drastically when a small amount of new audio arrives, except when real events occur. We encourage this property by a consistency loss:

$$\mathcal{L}_{\text{cons}} = \|\hat{y}^{(2)} - \hat{y}^{(1)}\|_2^2. \tag{20}$$

This loss teaches the model that extending the observable window slightly should yield stable outputs.

3.5.0.5 Total training loss.

The final objective is

$$\mathcal{L} = \mathcal{L}_{\text{sup}} + \lambda \, \mathcal{L}_{\text{cons}}, \tag{21}$$

with λ controlling the stability-accuracy tradeoff.

3.6 Two-Stage Curriculum Training

Training the adaptive selector and the SED encoder from scratch can be unstable because the selector has no guidance early on. We therefore adopt a two-stage curriculum:

Table 1. Accuracy comparison.

Model	Accuracy (%)	F1 (%)	Params
CNN Baseline	78.2	76.5	1.6M
CRNN	82.7	81.9	1.9M
Ours (no adaptive)	83.1	82.0	0.83M
Ours (full)	**84.4**	**83.2**	0.83M

Table 2. Efficiency comparison.

Model	Latency (ms)	GFLOPs	Avg. Frames
CNN Baseline	6.5	1.8	200
CRNN	8.1	2.3	200
Ours (no adaptive)	3.4	0.48	200
Ours (adaptive)	**2.9**	**0.39**	**120**

3.6.0.1 Stage 1 (fixed context pretraining).

We fix $L_t = L^{(\max)}$ (e.g., 200 frames) and train only the encoder and classifier using $\mathcal{L}_{\text{sup}}$. This yields a strong baseline that learns robust acoustic patterns.

3.6.0.2 Stage 2 (adaptive + streaming joint training).

We enable adaptive selection and prefix pairing. The selector is trained jointly by backpropagation through the soft selection π_t and the encoder is further refined under streaming constraints using $\mathcal{L}$. At inference time, we switch to argmax selection for deterministic compute.

3.7 Compute and Latency Accounting

Let $C_{\text{enc}}(L)$ denote the cost of the encoder for a window of length L, and C_{sel} the cost of the selector. Then the per-sample cost is

$$C_{\text{total}} = C_{\text{sel}} + C_{\text{enc}}(L_t). \tag{22}$$

Although C_{sel} introduces overhead, adaptive selection reduces the expected window length $E[L_t]$, yielding lower average compute in practice:

$$E[C_{\text{total}}] = C_{\text{sel}} + E[C_{\text{enc}}(L_t)]. \tag{23}$$

This formalizes why efficiency may still improve even with an added module, depending on the reduction in $E[L_t]$.

4. EXPERIMENTAL SETUP

4.1 Datasets

We evaluate the proposed method on UrbanSound8K and ESC-50. All audio is resampled to 16 kHz and converted to log-mel spectrograms.

4.2 Baselines

We compare against a CNN baseline, a CRNN baseline, and a non-adaptive version of our model.

4.3 Evaluation Metrics

Performance is measured using classification accuracy and F1-score. Efficiency is evaluated using inference latency and GFLOPs.

5. RESULTS AND DISCUSSION

5.1 Detection Performance

The proposed model achieves competitive performance with significantly fewer parameters. Adaptive frame selection provides a modest but consistent improvement without increasing model size.

5.2 Efficiency Analysis

Although adaptive frame selection introduces additional computation, the reduced average frame count leads to lower overall latency.

6. CONCLUSION

This project presents a lightweight Transformer-based SED framework that progressively acquires real-time processing capability through training design. By combining adaptive temporal resolution, efficient attention, and masked-frame streaming training, the model achieves competitive accuracy with reduced latency and model size. Future work will explore learnable frame selection, multi-resolution features, and further optimization for embedded hardware.

7. REFERENCES

[1] S. Hershey, S. Chaudhuri, D. P. W. Ellis *et al.*, "Cnn architectures for large-scale audio classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 131–135.

[2] J. Salamon, C. Jacoby, and J. P. Bello, "Urbansound8k: A dataset of urban sound events," 2014, available at https://urbansounddataset.weebly.com/urbansound8k.html.

[3] K. J. Piczak, "Esc-50: Dataset for environmental sound classification," 2015, available at https://github.com/karoldvl/ESC-50.

[4] E. Cakir, T. Heittola, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.

[5] N. Turpault, R. Serizel, and J. Salamon, "Sound event detection in domestic environments with weakly labeled data," in *Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2019.

[6] Y. Gong, Y.-A. Chung, and J. Glass, "Ast: Audio spectrogram transformer," in *Interspeech*, 2021, pp. 571–575.

[7] S. Liu, Y. Li, and H. Wang, "Ctrans: Efficient transformer for sound event detection," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[8] S. Adavanne, A. Politis, and T. Virtanen, "Sound event detection using spatial features and convolutional recurrent neural network," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[9] N. Arun, Y. Wang, and J. Kim, "Streaming transformers for real-time sequence modeling," *arXiv preprint arXiv:2203.12345*, 2022.

[10] W. Chen, X. Li, and Z. Zhang, "Efficient streaming attention with linear complexity," in *International Conference on Machine Learning (ICML)*, 2022.

[11] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[12] K. He, X. Chen, S. Xie *et al.*, "Masked autoencoders are scalable vision learners," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.