Comparative Evaluation of CNN, VGG16, Conformer, and YAMNet Models for Skateboard Riding Sound Detection

Haruka SAKAKI†, Kenshiro TANAKA†, Chenyu ZHAO†, Hiroshi TSUTSUI††, and Takeo OHGANE††
† Graduate School of Information Science and Technology, Hokkaido University
†† Faculty of Information Science and Technology, Hokkaido University

***Abstract*—In this paper, we present a comparative evaluation of four models, namely CNN, VGG16, Conformer, and YAMNet, as core components of a skateboard riding sound detection system. This system is designed to help reduce monitoring costs in public areas where unauthorized skateboarding may cause pedestrian obstruction or noise problems. We focus on the trade-off between model complexity, measured by the number of parameters, and detection performance, as well as the robustness of each model in realistic noisy environments. Experimental results show that the VGG16-based model achieves the highest performance, with area under the curve (AUC) values of 0.987 and 0.948 for clean and noisy test data, respectively. Notably, the CNN model, despite having fewer parameters and lacking pre-training, demonstrates strong performance, outperforming YAMNet and Conformer under noisy conditions, with AUC values of 0.970 and 0.904 for clean and noisy test data, respectively.**

I. Introduction

Skateboarding is a fascinating sport that combines accessibility, thrill, and a sense of accomplishment. However, skateboarding in areas other than designated areas can cause pedestrian obstruction and noise problems. This is also the case in Odori Park, one of the most famous parks in Japan, located in the center of Sapporo, Hokkaido. In a survey of 480 Sapporo residents conducted in 2021, skateboarding was cited as the most common complaint about Odori Park (169 responses), more than street smoking or bicycling. Despite the fact that skateboarding is prohibited by the Sapporo City ordinance, the city of Sapporo has implemented security patrols in Odori Park as skateboarding continues to be a common activity in the park. These patrols have proven somewhat effective, but continuous monitoring remains costly [1].

Therefore, we propose a skateboard riding sound detection system to reduce monitoring costs. In such a system, the discrimination performance of the model is important, and it is necessary to maintain high discrimination accuracy, especially when different sounds are mixed in a real environment. In this paper, we compare and evaluate the acoustic identification technique, which is the core of the skateboarding sound detection system, using four models: CNN, VGG16 [2], Conformer, and YAMNet. We discuss the trade-off between the number of parameters and performance of each model, and their effectiveness in real environments. We also focus on noise robustness as a critical factor for practical deployment, given the complex acoustic environment of the urban park.

The remainder of this paper is organized as follows. Section II describes the overview of the proposed skateboard riding sound detection system, the dataset used for our comparative evaluation of the models, and the features used as input to the models. Section III presents the architectures of the models evaluated in this paper, including CNN, VGG16, Conformer, and YAMNet. In Section IV, we present the experimental results and discuss the performance under clean and noisy test scenarios. Finally, Section V concludes the paper and outlines directions for future work.

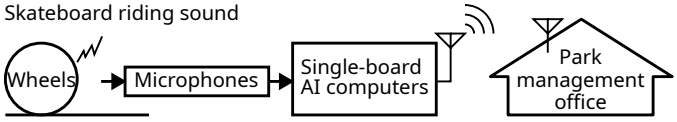

Fig. 1. Overview of the skateboard riding sound detection system

TABLE I
Datasets breakdown, counts of four-second clips

	Positive (skateboard)	Negative (others)
Train	1,353	3,656
Validation	338	913
Test	422	1,142
Total	2,113	5,711

Continuous skateboard sounds throughout the four-second duration are labeled as positive samples, while other sounds are labeled as negative samples.

II. System Overview and Dataset

A. System Overview

Figure 1 shows the overview of the skateboard riding sound detection system. Environmental sounds are acquired through microphones and converted into acoustic features for identification by models installed on the single-board AI computers. The single-board AI computers are small and easy to reposition, making them suitable for installation in various locations within the park. When skateboard riding sounds are detected, notifications are sent to the park management office via wireless communication. This system eliminates the need for constant monitoring and enables efficient patrolling by security personnel.

B. Dataset Collection

To collect skateboard riding sounds in real environments, recordings were made in Odori Park, Sapporo, Hokkaido. A TASCAM DR-07X [3] was used for recording, with a sampling frequency of 48,000 Hz and a quantization bit depth of 16 bits. Recordings were primarily conducted during summer evenings between 7 and 9 PM, collecting approximately 120 minutes of data in total. The recorded sounds mainly included fountain sounds, voices, singing, footsteps, and vehicle sounds, along with skateboard riding sounds.

The collected data was segmented into four-second clips, classifying those with continuous skateboard sounds throughout the four-second duration as positive samples and others as negative samples. Table I shows the breakdown of the

979-8-3315-6946-4/25/$31.00 ©2025 IEEE

TABLE II
FEATURE EXTRACTION PARAMETERS

Parameter name	Parameter value/type
Sampling rate	16 kHz
Frame length	25 ms for YAMNet, 64 ms for others
Shift length	10 ms for YAMNet, 32 ms for others
Windowing	Hanning window
Feature	Log-mel spectrum, 64- and 128-dimensional for YAMNet and others, respectively

TABLE III
COMPARISON OF MODEL ARCHITECTURES

Model	# params	Pre-trained	Input size
CNN	2.8 million	No	128×124×1
VGG16	14.8 million	ImageNet	128×124×3
Conformer	1.7 million	No	128×124×1
YAMNet	3.9 million	AudioSet	64 × 96 ×8

params: number of parameters.

TABLE IV
MODEL TRAINING PARAMETERS AND STRATEGY

Parameter name	Parameter value/type
Optimizer	AdamW with weight decay of 0.01
Loss function	Binary cross entropy
Batch size	128
Maximum epochs	20 to 160
Learning rates	Conformer: 5×10^{-6}, CNN: 10^{-5}, VGG16: 10^{-5}, YAMNet: 10^{-3}
Learning rate scheduler	Warmup and cosine decay
Early stopping	40 epochs patience

We adopt a warmup strategy by linearly increasing the learning rate during the first five epochs, and then apply cosine decay. The AdamW optimizer was used with a weight decay of 0.01 to prevent overfitting and improve generalization. Training parameters including learning rate and number of epochs were optimized for each model through preliminary experiments.

dataset. It forms an imbalanced dataset with a higher number of negative samples than positive samples.

C. Feature Extraction

We employ log-mel spectrograms as input features for skateboard riding sound detection purposes. Feature extraction is performed using the Librosa library [4], with parameters listed in Table II. Each four-second audio segment is transformed into a spectrogram of dimensions 128×124, where 128 corresponds to the number of mel frequency bins, and 124 to the number of time frames. Note that for the YAMNet classifier, which uses pre-trained weights on AudioSet, different parameters are used: frame length of 25 ms, shift length of 10 ms, and a 64-dimensional log-mel spectrum.

III. MODEL ARCHITECTURES

In this section, we describe the architectures of the four models evaluated in this paper. Each model was selected based on its distinct architectural characteristics:

- CNN for its simplicity and locality,
- VGG16 for deep convolutional structure and pre-training,
- Conformer for long-range dependencies, and
- YAMNet for lightweight event classification on mobile devices.

Tables III and IV show a key network parameter comparison and the model training parameters, along with the strategy, respectively. For all four models, the output layer is modified to be suitable for skateboard riding sound classification.

A. CNN

Figure 2 shows the architecture of the simple convolutional neural network (CNN) used in this paper. The CNN captures local features using convolutional layers, reduces the dimensionality with pooling layers, and performs final classification using fully connected layers. Our proposed model consists of six 2D convolutional layers, one average pooling layer, one global average pooling layer, and two fully connected layers. The first five convolutional layers use a kernel size of 3×3 and a stride of 1, with the number of channels set to 32, 64, 128, 256, and 512, as shown in Fig. 2. After the fifth convolutional layer, a 2×2 average pooling layer with a stride of two is applied to reduce the spatial dimensions. Then, a sixth convolutional layer with 256 channels is applied to further refine the features. The global average pooling layer aggregates the feature map into a single value per channel, and the final fully connected layer outputs the classification result. The total number of parameters in this model is approximately 2.8 million.

B. VGG16

Figure 3 shows the VGG16-based architecture used in this paper. The original VGG16 consists of 13 convolutional layers, five max-pooling layers, and three fully connected layers. Among these layers, 16 layers, convolutional and fully connected layers, have tunable parameters. In this paper, we modified it by replacing the flatten layer with a global average pooling layer as shown in Fig. 3. We experimentally confirmed that our modification had no significant effect on performance. The input size is 128×124×3 since VGG16 requires a three-channel input. We convert the input feature data from a spectrogram of dimensions 128×124×1 to 128×124×3 by duplication. Note that we utilize the weights pre-trained on ImageNet [5] to fine-tune the model. The total number of parameters in the model is approximately 14.8 million.

C. Conformer

Figure 4 shows Conformer architecture [6], which combines the local feature extraction capabilities of CNNs with the long-range dependency modeling abilities of the Transformer [7]. The model consists of multiple Conformer blocks, each of which effectively integrates a multi-head self-attention module (MHA), a convolutional module, and a feed-forward module (FFN).

Following the official PyTorch documentation [8], we set the kernel size of the first and second convolutional layers to 3×3 and the convolution modules in conformer blocks to 31. Considering the constraints of running on edge devices, we set the model dimension to 128 and the feed-forward network dimension to 256. The number of Conformer blocks and attention heads are both set to four. Although these values are smaller than those used in original Conformer models, we experimentally confirmed that the performance was not significantly degraded for this task. For positional encoding, we employ absolute positional encoding as proposed in [7]. The total number of parameters in this model is approximately 1.7 million.

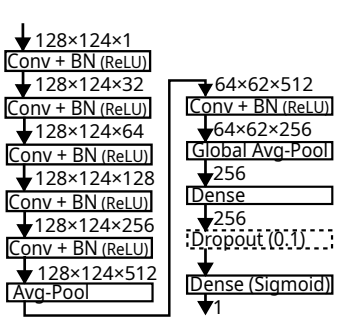

Fig. 2. CNN model. The number of parameters is 2.8 M, the smallest among the models. No pre-training is used.

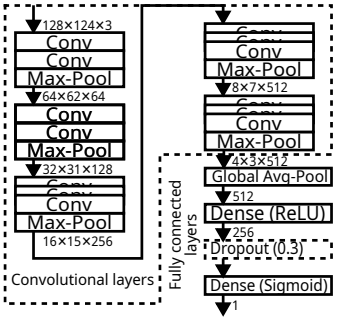

Fig. 3. VGG16-based model. The number of parameters is 14.8 M, the largest among the models. Pre-trained on ImageNet.

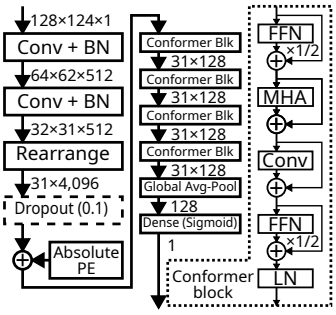

Fig. 4. Conformer-based model. The number of parameters is 1.7 M. No pre-training is used.

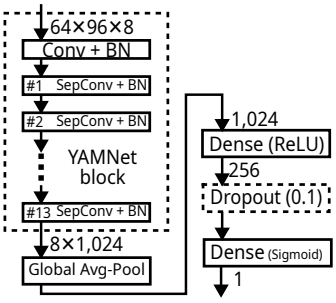

Fig. 5. YAMNet model. The number of parameters is 3.9 M. Pre-trained on AudioSet.

D. YAMNet

Figure 5 shows the architecture of YAMNet [9], an audio event classifier based on MobileNetV1 [10], where depthwise separable convolutions are utilized. In this paper, we fine-tune a model pre-trained using AudioSet [11]. To adapt it to our task, we follow the official documentation of YAMNet, where each four-second audio clip at 16 kHz is converted into data of size $8\times1{,}024$, as shown in Fig. 5. For comparison purposes, we describe the details of the YAMNet input feature, which has dimensions of $64\times96\times8$. Each four-second audio clip is segmented into eight patches, with each patch having a duration of 0.96 seconds and a stride of 0.48 seconds. Since the frame shift for YAMNet is 10 ms, as shown in Table II, each patch contains 96 frames, each of which has a 64-dimensional log-mel spectrum. The total number of parameters in this model is approximately 3.9 million.

IV. Comparative Evaluation Results

In this paper, we use the following metrics to evaluate model performance:

- Precision = TP/(TP+FP): the proportion of samples predicted as positive that were actually positive.
- Recall = TP/(TP+FN): the proportion of actual positive samples that were correctly predicted as positive.
- PR curve (precision-recall curve): a graph visualizing the relationship between precision and recall, which is suitable for evaluating imbalanced datasets.
- AUC (area under the curve): the area under the PR curve, where values closer to one indicate better performance.

A. Model Performance Comparison

Figure 6 shows the PR curves and corresponding AUC values for each model. The VGG16-based model attains the highest AUC of 0.987, followed by the CNN model at 0.972, YAMNet at 0.967, and the Conformer-based model at 0.947.

VGG16, despite having the largest number of parameters (14.8 million), leveraged transfer learning from ImageNet and achieved the best result. This suggests that pre-trained visual features, even from a different domain, can benefit audio-related tasks when properly adapted. The CNN model, though much lighter (2.8 million parameters) and trained from scratch, still performed comparably well, highlighting the effectiveness of compact convolutional designs with appropriate depth. YAMNet and the Conformer-based model, both with relatively low parameter counts and different architectural focuses, showed slightly lower AUCs. YAMNet uses a log-mel spectrogram input and depthwise separable convolutions, benefiting from AudioSet pre-training. The Conformer-based model incorporates self-attention mechanisms but uses fewer convolutional layers, which might have limited its ability to effectively extract localized features.

These results indicate that model performance is influenced not only by parameter size, but also by the combination of architecture, pre-training, and input representation.

B. Evaluation Under Noisy Conditions

We evaluated model performance under noisy environments using the following three types of noise from NOISEX-92 [12]:

- Clean data (C)
- Speech babble noise (B)
- White noise (W)
- Machine gun noise (M)

These noises were added to recorded data to achieve a signal-to-noise ratio (SNR) of 0, resulting in three distinct noisy datasets. At 0 dB, the skateboard riding sound can still be perceptible under close listening. Combined with the clean data, we constructed a multi-condition training dataset (Train: C+B+W+M), that was four times the size of the original dataset shown in Table I.

Figure 7 shows the evaluation results using clean data (Test: C) as the test data, while Fig. 8 presents the results using data with speech babble noise (Test: B). Both results are obtained using the models trained on the multi-condition dataset (Train: C+B+W+M).

Among all models, the VGG16-based model achieves the highest performance in both cases, with AUCs of 0.987 on clean and 0.948 on noisy data. Despite its large number of parameters, its use of ImageNet pre-training and deep convolutional structure likely contributed to its superior robustness.

The CNN model, although significantly smaller and trained from scratch without any pre-training, exhibited strong performance with an AUC of 0.970 on clean data and 0.904 under noisy conditions. Notably, CNN outperformed both YAMNet and the Conformer-based model in the noisy setting, demonstrating its relative robustness even without external feature priors.

YAMNet, pre-trained on AudioSet, showed a moderate drop in performance from 0.954 on clean data to 0.861 under noisy

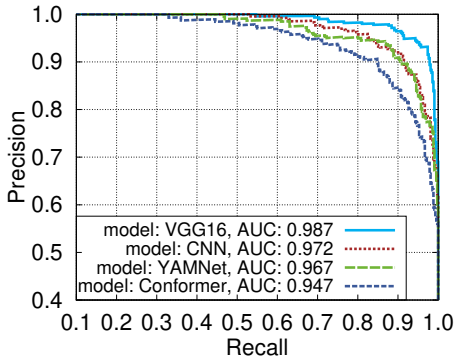

Fig. 6. Model performance comparison, trained and tested with clean data (Train: C, Test: C)

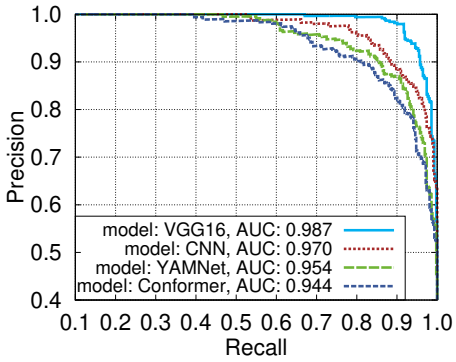

Fig. 7. Model performance comparison, trained with clean data and three types of noise-added data, tested with clean data (Train: C+B+W+M, Test: C)

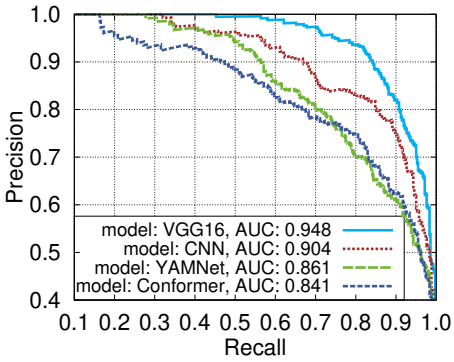

Fig. 8. Model performance comparison, trained with clean data and three types of noise-added data, tested with data augmented with speech babble noise (Train: C+B+W+M, Test: B)

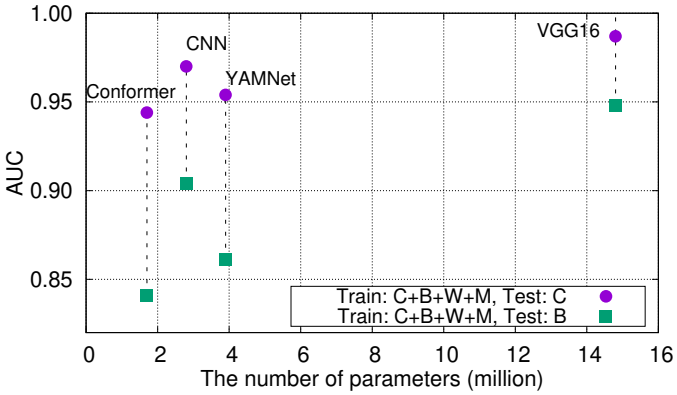

Fig. 9. Relationship between the number of parameters of each model and the AUC values. The AUC values can be found in Figs. 7 and 8 for Test: C (clean) and B (speech babble), respectively. Note that both results are obtained using the models trained on the multi-condition dataset, Train: C+B+W+M. The number of parameters is summarized in Table III.

conditions. The Conformer-based model, having the fewest parameters and no pre-training, exhibited the lowest AUCs of 0.944 on clean and 0.841 on noisy data. This suggests that while compact and efficient, these models are more susceptible to noise, possibly due to limited convolutional capacity.

To further contextualize these findings, Fig. 9 plots model AUCs against their parameter sizes. While VGG16 achieves the highest accuracy at the cost of model size, CNN offers a compelling trade-off, maintaining high robustness under noise despite its lightweight and absence of pre-training. This contrasts with YAMNet and Conformer, whose compactness and alternative architectural choices result in decreased generalization in noisy environments.

These results highlight that, although model size and pre-training can enhance robustness, architectural factors such as convolutional depth and local receptive fields, which are well embodied by VGG16 and CNN, remain critical for reliable sound detection in real-world noisy conditions.

V. Conclusion

In this paper, we compared the performance of four model architectures, CNN, VGG16, Conformer, and YAMNet, for use in the skateboard riding sound detection system. The results showed that models with deep convolutional layers, such as CNN and the VGG16-based model, achieved higher classification performance. Furthermore, applying multi-condition training with various types of environmental noise enabled the VGG16-based model to maintain high performance under both clean and noisy conditions, indicating strong noise robustness despite its larger model size. As future work, we plan to evaluate the model's performance in more complex real-world scenarios, such as environments with overlapping wheel sounds from strollers and luggage, to further enhance the system's practical applicability.

Acknowledgments

This work was supported by the Grants for Revitalization of Regional Universities and Industries, "Realization of a semiconductor complex base triggered by next-generation semiconductors and revitalization of local economies."

References

[1] The Hokkaido Shimbun Press, "Why does skateboarding persist in Odori Park despite complaints?: A surge in complaints to Sapporo City — Free but narrow facility in Nishi Ward," 2023, in Japanese.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, May 2015.

[3] TASCAM, "DR-07X | 2-channel portable handheld recorder with USB interface," Retrieved July 20, 2025. [Online]. Available: https://tascam.jp/int/product/dr-07x/top

[4] B. McFee, C. Raffel, D. Liang *et al.*, "librosa: Audio and music signal analysis in python," in *Proc. SCIPY 2015*, 2015.

[5] J. Deng, W. Dong, R. Socher *et al.*, "ImageNet: A large-scale hierarchical image database," in *Proc. CVPR*, Jun. 2009, pp. 248–255.

[6] A. Gulati, Q. Qin, C.-C. Chiu *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020. [Online]. Available: https://arxiv.org/abs/2005.08100

[7] A. Vaswani, N. Shazeer, N. Parmar *et al.*, "Attention is all you need," in *Proc. NeurIPS*. Curran Associates, Inc., 2017.

[8] Torchaudio, "Conformer | Torchaudio 2.5.0.dev20241105 documentation, torchaudio.models.Conformer," 2024, Retrieved July 20, 2025. [Online]. Available: https://docs.pytorch.org/audio/main/generated/torchaudio.models.Conformer.html

[9] Google Research, "YAMNet: A pretrained deep net for audio event classification," 2020. [Online]. Available: https://tfhub.dev/google/yamnet/1

[10] A. G. Howard, M. Zhu, B. Chen *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017. [Online]. Available: https://arxiv.org/abs/1704.04861

[11] J. F. Gemmeke, D. P. W. Ellis, D. Freedman *et al.*, "AudioSet: An ontology and human-labeled dataset for audio events," in *Proc. ICASSP*, 2017, pp. 776–780.

[12] A. Varga and H. J. M. Steeneken, "Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems," *Speech Communication*, vol. 12, no. 3, pp. 247–251, Jul. 1993.