https://doi.org/10.7840/kics.2023.48.2.150

Real-Time Tunnel Abnormal Sound Detection Algorithm Using Convolutional Neural Networks

Juyoung Lee◆°, Chunkyun Park*, Hyunjoong Kim*

ABSTRACT

In the traffic industry, the automatic accident detection system is a major concern. Although image-based and radar-based traffic accident detection systems are commonly employed, they have several drawbacks, including the need to secure the camera's field of view, a high rate of false alarms, and a lengthy detection time. Using a real-time acoustic surveillance system and the classification algorithm via Convolutional Neural Network (CNN), this article proposes several methods for identifying abnormal situations, such as a car crash or tire skid sound, to overcome the limitations of existing methods. We create an audio database by collecting sounds from two tunnels in South Korea using self-made microphones for eight months and classifying them into three categories: car crash, tire skid, and normal environmental sounds. We establish a three-step classification procedure using an algorithm. We compare the detection rate and false alarm rate of our proposed method to those of deep learning techniques including MLP (Multi-Layer Perceptron), Long-Short Term Memory, ShuffleNetv2, and MobileNetv2. In addition, we present a method for filtering out irrelevant sound data to improve the computational efficiency of our approach.

Key Words : Convolutional Neural Networks, Acoustic-based Accident Detection System (AADS), Real-time Accident Detection System, Audio Classification, Car Crash Sound, Tire Skid Sound, Signal Processing

Ⅰ. Introduction

Due to the rising need for road security and safety, there has been a rise in interest in advanced traffic surveillance systems[1]. Tunnels are a unique type of road in that they have a very restricted escape route and distinctive acoustic properties compared to other types of roads. Due to the tunnel's spatial and temporal limitations, it is difficult to detect accidents with speed and accuracy. As a result, most tunnel accidents result in secondary incidents, such as multiple collisions, which can cause severe economic losses, fatalities, or both. Because of this, accurate accident detection in tunnels is more crucial than on other types of roads.

Over the past several decades, global expansions of security-related fields have been observed. As the need for security has increased, image-based surveillance systems have emerged as a crucial area of research. These systems primarily utilize visual data, such as video footage[2, 3], radar data[4] and ultraviolet/infrared data[5]. They perform well, but have several limitations in environments where vision is dysfunctional, such as when there is smoke, darkness, fog, or other environmental conditions. Not only do these systems often have a delayed detection time and a high false alarm rate[6], but they also have high processing costs and are susceptible to certain physical factors, such as

◆° First and Corresponding Author : Yonsei University Department of Applied Statistics, juyoung.lee0422@gmail.com, 학생회원
* Yonsei University Department of Applied Statistics, chunkyun1028@gmail.com; hkim@gmail.com

논문번호 : 202211-282-RU, Received November 22, 2022; Revised December 5, 2022; Accepted December 9, 2022

camera blind spots and the obstruction of vision by other large-sized vehicles.

As a complementary and alternative approach to image-based accident detection systems, Acoustic-based Accident Detection Systems (AADS)[7] have garnered significant interest as a suitable means of addressing these issues. The AADS process consists of three steps: the collection of real-time sounds, the extraction of features for accident detection using a proposed model, and the classification of sounds into three categories: car crash, tire skid, and normal environmental sounds.

In the final step of the algorithm, a performant classifier is essential. Historically, early classifiers were based on statistical models, such as Support Vector Machine (SVM)[8, 9] and Hidden Markov Models[10, 11]. Recent advances in computing power have given rise to a DNN-based method[12] such as Recurrent Neural Network (RNN)[13] and Long-Short Term Memory (LSTM)[14, 15]. They are frequently used as classifiers in Sound Event Detection (SED). Classifiers based on Convolutional Neural Networks (CNN), which are widely used in computer vision, have demonstrated remarkable performance in SED tasks when compared to conventional approaches.

There are studies on the classification of environmental sounds[16-20], bird sounds[21], detection of abnormal heart and lung sounds[22] and classification of traffic sounds[23, 24]. However, the focus of the paper was primarily on the performance of classifiers that predict the classes of each sound based on previously collected datasets. They have not considered how to collect and process acoustic data in real-time. We have developed the entire procedure for collecting acoustic data from tunnels, extracting features from collected acoustic data, and classifying them in real-time based on our years of experience operating AADS in tunnels. In this paper, we propose a comprehensive method for detecting tunnel-caused accidents or hazardous situations using a real-time acoustic surveillance system.

Process efficiency and performance are important in real-time surveillance systems because they are related directly to the time required to detect an event and the operational costs of the system. Voice Activity Detection (VAD)[25] is a voice recognition technique that detects the occurrence of a voice in a noisy environment. We assumed that the fundamental concept of VAD could make our algorithm more effective because tunnel accidents are uncommon. In this paper, we present an Event Activity Detection (EAD) method that employs the results of sound feature extraction to determine when an important event occurs. In contrast to other EAD approaches, our proposed EAD method can ignore irrelevant noises and extract only the specified event sounds.

The structure of the remaining sections of the article is as follows: Section 2 describes the details of the dataset relating to tunnel-related occurrences. In 3.1, we describe how to preprocess sounds and introduce the hyper-parameter associated with the input scale of the deep learning model, including methods for sound feature extraction, body size, shift size, and window size. In 3.2, we describe the EAD algorithm for detecting the precise occurrence of an event in an efficient manner. In 3.3, we present a method for identifying real-time audio data using CNN models and introduce the architecture of our CNN model. Summarize section 3 in 3.4. Section 4 explains performance evaluation metric, learning methodologies, and testing procedures. In Section 5, we compare the performance of the proposed CNN model to that of the MLP, LSTM, and ShuffleNetv2[26], MobileNetv2[27], utilizing the evaluation metric described in section 4. Section 6 concludes this paper.

Ⅱ. Dataset

The absence of public tunnel databases for use as references has posed the greatest challenge to the research. The Mivia road dataset[28, 29] has been extensively applied to the classification of abnormal traffic sounds and includes three classes: car crash sounds, tire skid sounds, and environmental sounds. However, it does not include every conceivable event that could occur in the tunnels under consideration. Moreover, because we are solely interested in the classification of tunnel-related events, this dataset is unsuitable for our research. The acoustic environment of a tunnel is distinct from that of a public road. Since the road conditions in long tunnels are incredibly unique due to weather or traffic conditions[30], we had to apply a different methodology to our tunnels data as opposed to the methods currently in use.

Eight months in two domestic tunnels, we collected real-time sound data from tunnels using microphones we made ourselves.

The self-made microphone is capable of recording sound data up to 75m away. We installed these microphones every 100m from the entrance to the exit of the tunnel and equipped the tunnel with 2m-tall walls.

When input signal is recorded, this microphone converts input signal to a digital signal and transfers it to an analytic server using the TCP/IP protocol. We use a digital signal with a sample rate of 48kHz, 1 second, 2 channels (L/R), and 16 bits (=2bytes) for sound recording. Therefore, 1 second of sound data generates 192,000 bytes, and we send 16,384 bytes of data to the analytic server due to a predefined protocol. Based on the ratio of 16,384 bytes to 192,000 bytes, the value is close to 0.085 seconds. Thus, every 0.085 seconds, 16,384 bytes of data can be transmitted.

Table 1. Description of tunnel event classes and data figures

Class	Description	n
a1	Car crash	111
a2	Tire skid	255
b0	Environmental sounds	1141
Total		1507

The tunnel occurrences were divided into three distinct categories. Table 1 provides material and statistical data by category. The classes that must be detected, including car crash and tire skid sounds, are labeled with the letter “a”, while the remaining environmental sound classes, including siren and horn sounds, are labeled with the letter “b”. Since the data used in this paper belongs to a specific company, we ask for your understanding as we disclose only a portion of it. Each class's sample data is available at

https://github.com/joo-young-lee/TunnelCNN

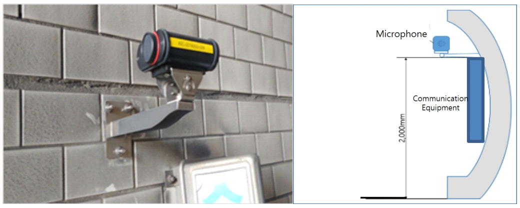

Fig. 1. (left) Image of the self-made microphone, (right) Installation of the self-made microphone

Each data sample lasts 30 seconds, and only a small portion of each sample contains the sound of an actual event. In general, event sounds begin approximately 10 seconds after the beginning of the sample and end within three seconds, except for sirens.

In some circumstances, a single sample data can include multiple events. For example, car crash sounds (a1) typically follow tire skid sounds (a2) because drivers instinctively stop their vehicles to limit the damage as soon as they realize a collision is likely to occur. We categorized the data samples in this instance as belonging to the class with louder sounds for convenience.

Ⅲ. Proposed Methods

3.1 Real-Time Sound Preprocessing

This section explains how to implement real-time sound preprocessing, including how to convert raw sound data into the correct format for a classifier.

For processing in real-time, the data must be transformed into the right format with a defined size. In two-dimensional data - time-frequency grid - the horizontal axis represents time, and the vertical axis represents frequency. As we mentioned above, the sound that was recorded in real-time must be separated into the right size because the time axis of the input cannot be infinite. Additionally, the time shift till the next time-frequency data must be calculated. We have defined the following related terms:

The smallest among terms is a window size. We use a rectangular window function for the window size and set the duration to 0.05s. And these windows are group to form a body, and the body size is set to 3s. The distance between one sample and the next is referred to as a shift size. The shift size is set to 1s. A min/max bandwidth is parameter that specifies the section of the vertical axis (frequency) that will be used to extract features from actual inputs. We set it to 300-7 kHz. Since a frequency distribution of the collected sounds was within 7 kHz and that noises in the tunnel below 300 Hz had a common characteristic, they were eliminated to reduce the size of the input.

One of the straightforward ways to predict the sound collected in real-time with our CNN models is to extract the features corresponding to one body from the current time point and use it as the initial input value, then move the time point and extract the features corresponding to the next body. In this regard, we must define in advance certain hyper-parameters associated with a sound feature extraction.

There are so many sound feature extraction methods. In this paper, the Fast Fourier Transform (FFT)[31], with low operating cost and low complexity is used to transform the data.

3.2 EAD (Event Activity Detection)

Traditionally, multiple methods for sound classification were employed to collect sound data all at once in the end. Then put all of them to train the model, and apply it. In other words, there is no sound detection in real-time - this is the reason for the lengthy detection time. In addition, the collection of sound data all at the same time may decrease operational efficiency.

EAD is designed to detect an event only when a particular event has happened. If the EAD method is not used, all data should be used as input data for deep learning models. Since only a minute portion of the real-time sound collection contains remarkable events, the remaining data is insignificant and can be disregarded. To filter out irrelevant sound data, our initial EAD hypothesis is that, if any events have occurred, the current body's information quantity is likely to be greater than the previous body's.

Here, we define some necessary notations. Let denote the body index $l(l = 0, 1, 2, ...)$, b represents the body size, s represents the shift size, $x:y$ represents a body starting at time point x and ending at time point y. We begin by extracting features, the magnitude of $(l * s - b):(l * s)$ at the first body and using it as the classification model's initial input value. Then, we shift the body by multiplying s to the first body. In this manner, we can detect the event using the $(l * s - b):(l * s)$, $\{(l + 1) * s - b\}:\{(l + 1) * s\}$, $\{(l + 2) * s - b\}:\{(l + 2) * s\}$, $\cdots$, as the input, by iterating. We consider the sum of the body's magnitude values to be its information quantity and determine when an event occurs by calculating the ratio of the current body's information quantity to that of the previous body.

Again, EAD is a method for event detection. Also, we cannot enter all inputs all at the same. Input must be segmented into specific units, converted to accommodate the classifier, and only those that can be regarded events must be implemented into the model.

Consider the case where the event occurred not at the first body but at the second body. In this situation, the sum of the second body relative to the sum of the first body would be astronomically large. If the ratio between and the sum of the second body and the sum of the first body is greater than the constant k, we can assume an event has taken place, the third body, $\{(l + 2) * s - b\}:\{(l + 2) * s\}$, will be the input value. In other words, if the following equation for $k \geq 1$ holds true:

$$\frac{\Sigma(l:l+s)}{\Sigma(l-b):(l*s-b)} > k \tag{1}$$

In this case, if the second body is used as the input value, the event in the second body exists at the extreme right, and there may be instances where the entire event cannot be included. For example, the tire skid sound frequently occurs after a car accident, but if we only use the second body, the tire skid sound characteristics may not be contained within the second body. It is possible to predict incorrectly a sound like car sounds in this circumstance. We must move a critical window to the center of the body. Consequently, the accuracy of the deep learning model can be improved by inserting the skid sound in the next body, the third body $\{(l+2)*s-b\}:\{(l+2)*s\}$.

If b in equation (1) is greater than s, the numerator and the denominator overlap $(l*s-b):(l*s)$. In other words, s cannot exceed b. If s is greater than b, data can be discontinuous. This may result in data loss. To enhance the efficiency of the EAD algorithm, it is possible to eliminate the redundant sum of frequently overlapping $(l*s-b):(l*s)$. Consequently, if the input is $\{(l+2)*s-b\}:\{(l+2)*s\}$.

In the case of $b>s$ in equation (1), $\{(l+1)*s-b\}:\{(l+1)*s\}$ of the numerator and denominator frames overlaps. To enhance the performance of the EAD algorithm, it is possible to reduce the redundant sum of frequently overlapping $\Sigma\{(l+1)*s-b\}:\{(l+1)*s\}$. That is, if $\{(l+2)*s-b\}:\{(l+2)*s\}$ is used as the input. Eventually, all our inputs are data that satisfied equation (1) and should have to be the third body - $\{(l+2)*s-b\}:\{(l+2)*s\}$.

$$\frac{\Sigma(l:l*s)}{\Sigma(l-b):(l*s-b)} > k' \tag{2}$$

In general, the constant k' should be the highest value capable of generating bodys containing all accident sounds, and it is dependent on tunnel-specific parameters such as time and environment (weather, seasonal, etc.). Because sound is amplified and echoed more in a narrow tunnel than in other tunnels, must be increased. We chose daytime (06:00-22:00) k values to be lower than midnight (22:00-06:00) k values. The reason for setting this parameter is that, according to an 8-month analysis of the data, minor accidents are more likely to occur during the day because there are more vehicles on the road, whereas there is a greater chance of a major accident occurring at midnight because there are fewer vehicles on the road. This resulted in an average daily reduction of input bodies to the classifier of approximately 27 percent.

3.3 Classifier

We use our own CNN model. MLP, LSTM, ShuffleNetv2, and MobileNetv2 were also utilized for comparison.

The architecture of the CNN model utilized in the experiment is depicted in Figure 2. Conv2d is a two-dimensional (2-D) convolutional computation with (i, o, ($k1$, $k2$), ($s1$, $s2$), ($p1$, $p2$)), where i is the number of input channels, o is the number of output channels, $k1 \times k2$ is the size of the kernels, $s1 \times s2$ is the size of the strides, and $p1 \times p2$ is the size of the padding. Each convolution layer has batch normalization, dropout, and ReLU activation functions applied. Due to the asymmetry of the input, we also utilized a rectangular convolutional operation with a kernel size of (3, 5) and a padding size of (1, 2). The reason for the asymmetric input is described in full in section 4.1.

Early CNN models employed fully connected (FC) layers after flattening the feature map generated from the convolution layer, and global average pooling (GAP)[32] layers have been commonly used for CNN classification in recent years. GAP computation is the process of acquiring $(1 \times 1 \times d)$ tensors by averaging the values of all height & weight values of $(h \times w \times d)$ tensors by each channel, flattening them to a one-dimensional value, and connecting them to the FC layer based on the number of outputs. Because pooling computation does not require learning parameters, the GAP layer produces a model with far less parameters than the FC layer due to its light-weighted nature. Consequently, the GAP layer was applied to the final convolution layer of the model based on the size of each channel $(h \times w)$. Finally, Softmax functions were employed in the output layer to compute the cross-entropy loss.

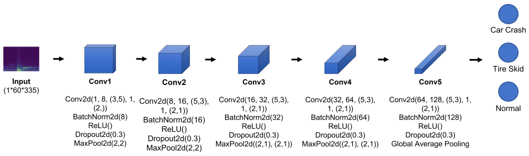

Fig. 2. A detailed structure of the proposed convolutional neural networks

3.4 Overview

This section provides a summary of the techniques introduced in Section 3. The ratio between the sum of features in the current body and the next body is calculated. If the value is greater than the predefined threshold , the third body is classified as a car crash, tire skid, or environmental sounds using a classifier; otherwise, the process continues. Then, these steps are repeated for $(l * s - b) : (l * s)$ to shift the current body by the shift size s. The schematic representation of this procedure is depicted in Figure 3. We use our own CNN model. MLP, LSTM, ShuffleNetv2, and MobileNetv2 were also utilized for comparison.

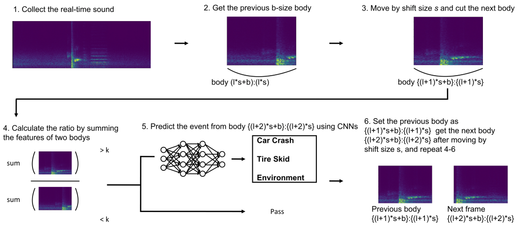
1. Collect the real-time sound
2. Get the previous b-size body
body (l*s+b):(l*s)
3. Move by shift size s and cut the next body
body {(l+1)*s+b}:{(l+1)*s}
4. Calculate the ratio by summing the features of two bodys
sum
sum
> k
< k
5. Predict the event from body {(l+2)*s+b}:{(l+2)*s} using CNNs
Car Crash
Tire Skid
Environment
Pass
6. Set the previous body as {(l+1)*s+b}:{(l+1)*s} get the next body {(l+2)*s+b}:{(l+2)*s} after moving by shift size s, and repeat 4-6
Previous body
{(l+1)*s+b}:{(l+1)*s}
Next frame
{(l+2)*s+b}:{(l+2)*s}

Fig. 3. Proposed algorithm to process real-time sound and detect events using convolutional neural network

Ⅳ. Experiments

In this section, we compare the proposed CNN model's detection rate and false alarm rate to those of existing deep learning models. The primary goal is to maximize detection while minimizing false alarms. For training purposes, we designate the last three classes as follows: the car crash as a1, the tire skid as a2, and everything else as b0. Following are the calculations for Table 2's metric, which is a confusion matrix with three outputs.

Table 2. Confusion matrix of the car crash, tire skid, and environmental sounds classification

Actual \ Pred	a1	a2	b0
a1	x_{11}	x_{12}	x_{13}
a2	x_{21}	x_{22}	x_{32}
b0	x_{31}	x_{32}	x_{33}

- Detection Rate (DR): The following formula demonstrates the proportion of correct detection of a1 and a2. The detection rate is computed based on the following data from Table 2:

$$\mathrm{DR} = \frac{x_{11}+x_{22}}{x_{11}+x_{12}+x_{21}+x_{22}+x_{31}+x_{33}} \tag{3}$$

- False Alarm Rate (FAR): The proportion of environmental sounds that are incorrectly identified as a1 or a2 is depicted by the following formula. The following formula is used to calculate the false alarm rate based on Table 2 values:

$$\mathrm{FAR} = \frac{x_{13}+x_{23}}{x_{13}+x_{23}+x_{33}} \tag{4}$$

4.1 Learning method and criterion

Among the many proposed and well-known classifiers, we chose MLP, LSTM, ShuffleNetv2 and MobileNetv2 as competitors for our own model. ShuffleNetv2 and MobileNetv2 are CNN models with few parameters that are learned from TorchVision's pre-trained models via transfer learning.

The LSTM model utilizes two LSTM layers to predict FC output layers, whereas the MLP model consists of two hidden FC layers and one output layer after flattening the image data. Table 3 displays the number of parameters for each of the five classification models, whereas our CNN model possesses 0.1M parameters. This is a minimum of a third less than the ShuffleNetv2 and the least amount among comparable models.

In addition, we investigated how the EAD algorithm's threshold affects the algorithm's equation (1). Real-time processing efficiency can be enhanced by determining the optimal value of

Table 3. The number of parameters by classification models

Classifier	# of parameters
Ours	0.1M
MLP	5.2M
LSTM	1.1M
ShuffleNetv2	0.34M
MobileNetv2	2.2M

to maintain the detection rate as the model's optimal performance and reduce the false alarm rate as the filter of the negligible body. Thus, the threshold k determined by the train set was applied to the validation set and test set. Figure 4 is a scatter plot displaying a spectrum of EAD criteria for each class for the train set. At the optimal value k (= 1.028), the EAD algorithm did not filter out the sounds of the car crash or tire skid, but it did filter out all other environmental sounds. We divided EAD scenarios into these four categories for the experiment: non-application ($k = 0$), minimal application ($k = 1$), optimal application ($k = 1.028$), and excessive application (k = 1.1).

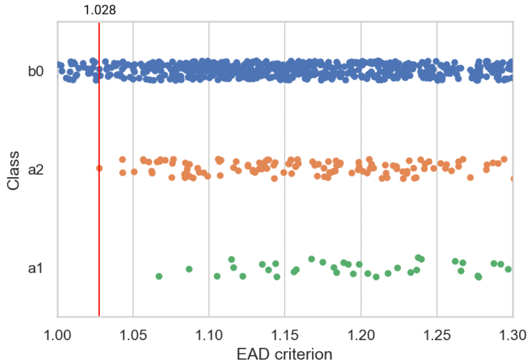

Fig. 4. EAD criterion for each sound source by class (car crash, tire skid, environmental sounds)

4.2 Hyper-parameters setting

Prior to conducting experiments, it was necessary to select hyper-parameters described in section 3.1. By extracting acoustic features, Mel spectrogram, MFCC, and FFT have all been used to overcome acoustic classification issues. Before classifying abnormal sounds from raw data, the number of component signals in the raw data had to be determined. To determine the number of distinct signals, we utilized FFT, which with low operating cost and low complexity was used to detect, convert, analyze, and store acoustic data in real-time during operation of the system. This enabled the frequency domain conversion of a time-sequenced mixed signal. The FFT algorithm is derived from the Numpy Python library[33], and only the real components of the magnitude are used. As input values, we only use FFT output information for the frequency bandwidth between 300 Hz and 7 kHz. We determined empirically that environmental sounds in tunnels corrupt the low-frequency range (0-299 Hz), whereas the higher-frequency range (above 7 kHz) are not informative, we found out through many trainings. In addition, because events in three classes do not last longer than three seconds, the body size has been set to 3s, and the window size has been set to 0.05s which, according to multiple studies, best distinguishes between class characteristics. In this configuration of hyper-parameters, the horizontal axis measured 60 and the vertical axis measured 335. Figure 5 illustrates three visualization classes.

Min-max normalization and an Adam optimizer[34] with weight decay = 0.005 were applied to the input values. Learning's hyper-parameters were as follows: learning rate = 0.002, dropout rate = 0.3, and mini-batch size = 64. In addition, model

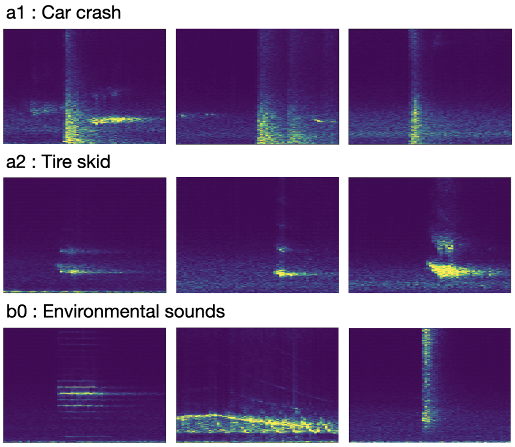

Fig. 5. FFT visualization for each class (body = 3s, window = 0.05s, 300Hz-7kHz)

overfitting was avoided via the use of early stopping. In order to be more precise, the train set and test set were split 7:3, and 5-fold cross-validation was performed in the train set to prevent overfitting. As a bodywork for deep learning, PyTorch[35] was utilized.

4.3 Test Procedure

30-second test sets were combined into a single test file to simulate real-time conditions. Due to security concerns, information from all time periods could not be utilized. Therefore, a sound source was created for 30 seconds, including 20 seconds forward and 10 seconds backwards from the time the event concluded, and all event sources were combined to create an experimental environment that resembled the real-time environment. This 30-second file is processed using the Section 3 method, and classifiers detect the event utilizing EAD body's containing meaningful data. If the current body's information is reduced or identical to the previous body's information, it is skipped and not examined because it does not meet the implementation requirements of the EAD algorithm.

We can obtain multiple output values from deep learning models using a single sound using our method. In this instance, the various output values were compiled into a single result in order to populate Table 2. If an accident sound includes at least one of a1 and a2, maximum voting determines whether the sound is a car crash sound (a1) or a skid sound (a2). If neither a1 nor a2 is present, it is determined that the condition has not been detected or processed as an accident has not occurred. For example, for true a1, deep learning models that anticipated the car crash sound is applied to x_{11}, whereas a2 is applied to x_{12}. If the results do not apply to the instances listed above, they are implemented as x_{13}. Similarly, in the case of tire skid sounds, a single body prediction as a2 is applied to x_{22}, and no accident is applied to x_{23}. If the event is truly b0 and not a false alarm, it will be counted according to Table 2 in x_{33}. Unless the actual occurrence is b0, the results are recorded as x_{13} or x_{23}.

V. Results

Table 4 displays the results of each classifier's DR and FAR according to the EAD method. Despite having the fewest parameters, our suggested model has the highest detection rate (84.65%) and the lowest FAR (7.72%) among the other models with optimal. On the one hand, MLP with excessive has the lowest FAR (6.46%), but it is not a desirable model because the detection rate (the primary metric) is incredibly low. Similarly, our model with an excessive k has a lower FAR (7.19%) than when k is optimal (7.72%), but the DR is nearly 5% lower. We can conclude that our model with the optimal k had the best performance. In addition, Table 4 reveals that CNN-based models (including ours, ShuffleNetv2, and MobileNetv2) have a higher DR and lower FAR than MLP- and LSTM-based models. High DR and FAR contribute to LSTM's tendency to detect every loud sound regardless of class. Because it disregards acoustic characteristics, MLP has the lowest detection rate.

This indicates that it is challenging to distinguish false alarm classes from event classes when translating sound data to visual data using FFT. In order to improve the performance of CNN classifiers, it will be necessary to investigate different feature extraction approaches from one that can distinguish between b0 and accident sounds in future research.

Table 4. DR/FAR (unit: %) performance comparison of classification models by threshold k in the EAD algorithm

EAD	Metrices	Classifier				
		Ours	MLP	LSTM	Shuffle Netv2	Mobile Netv2
No EAD (=0)	DR	**84.65**	47.94	75.61	71.50	76.44
	FAR	**7.98**	12.36	60.43	12.72	9.38
Min k (=1.0)	DR	**84.65**	47.94	75.61	71.50	76.44
	FAR	**7.89**	9.73	55.69	12.54	8.86
Optimal k (=1.028)	DR	**84.65**	47.94	75.61	71.23	76.44
	FAR	**7.72**	9.12	48.41	11.58	8.42
Excessive k(=1.1)	DR	**79.45**	44.93	73.97	68.76	73.69
	FAR	7.19	**6.46**	24.82	11.05	8.16

Ⅵ. Conclusion

In this study, we present a series of procedures that detect abnormal sounds in real-time, enabling tunnel operators to respond immediately to the first accident and to prevent the second accident in advance. This sequence comprises three stages. To recognize occurrences, the first step is to identify abnormal sound and filter out irrelevant sound. The second step consists of extracting and converting sound data to 2D image data. The last step is to classify the visual data into three distinct categories: car crash sounds, tire skid sounds, and environmental sounds.

The contributions of the paper are listed below. Using an algorithm, the model can distinguish between useful and irrelevant sounds and eliminate irrelevant data to increase efficiency. In addition to classifying abnormal or environmental sounds in real-time, converting sound to a 2D image, and defining this data. This procedure sequence has been algorithmized and implemented in a real tunnel. In upcoming projects, data loss as a result of data passing through multiple processes should also be considered. Additionally, we can utilize and modify novel algorithms. In addition, compare our research to the common sound datasets used in signal processing in the future.

References

[1] J. Kim, "Vehicle detection using deep learning technique in tunnel road environments," *Symmetry*, vol. 12, no. 12, 2020. (https://doi.org/10.3390/sym12122012)

[2] J. Baek, J. Min, S. Namkoong, and S. Yoon, "An in-tunnel traffic accident detection algorithm using cctv image processing," *KIPS Trans. Softw. and Data Eng.*, vol. 4, no. 2, pp. 83-90, 2015. (https://doi.org/10.3745/KTSDE.2015.4.2.83)

[3] K. B. Lee and H. S. Shin, "An application of a deep learning algorithm for automatic detection of unexpected accidents under bad cctv monitoring conditions in tunnels," in *IEEE Int. Conf. Deep Learn. and Mach. Learn. in Emerging Appl. (Deep-ML)*, pp. 7-11, Istanbul, Turkey, Aug. 2019. (https://doi.org/10.1109/Deep-ML.2019.00010)

[4] Y.-D. Kim, G.-J. Son, C.-H. Song, and H.-K. Kim, "On the deployment and noise filtering of vehicular radar application for detection enhancement in roads and tunnels," *Sensors*, vol. 18, no. 3, 837, 2018. (https://doi.org/10.3390/s18030837)

[5] Y.-D. Kim, G.-J. Son, H. Kim, C. Song, and J.-H. Lee, "Smart disaster response in vehicular tunnels: Technologies for search and rescue applications," *Sustainability*, vol. 10, no. 7, 2509, 2018. (https://doi.org/10.3390/su10072509)

[6] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio Surveillance of Roads: A System for detecting anomalous sounds," *IEEE Trans. Intell. Transport. Syst.*, vol. 17, no. 1, pp. 279-288, 2016. (https://doi.org/10.1109/TITS.2015.2470216)

[7] J. Jang, "Instantaneous incident detection system based on analysis of acoustic signal from crash and skid in tunnel," *The Open Transport. J.*, vol. 12, no. 1, 2018. (http://dx.doi.org/10.2174/1874447801812010344)

[8] Y. Zhu, Z. Ming, and Q. Huang, "Svm-based audio classification for content-based multimedia retrieval," *Multimedia Content Anal. and Mining, LNCS*, vol. 4577, Heidelberg, Berlin, Jun. 2007. (https://doi.org/10.1007/978-3-540-73417-8_56)

[9] M. Xu, C. Xu, L. Duan, J. S. Jin, and S. Luo, "Audio keywords generation for sportsvideo analysis," *ACM Trans. Multimedia Comput., Commun., and Appl.*, vol. 4, no. 2, pp. 1-23, 2008. (https://doi.org/10.1145/1352012.1352015)

[10] C. Couvreur, V. Fontaine, P. Gaunard, and C. G. Mubikangiey, "Automatic classification of environmental noise events by hidden markov models," *Applied Acoustics*, vol. 54, no. 3, pp.

187-206, 1998.
(https://doi.org/10.1016/S0003-682X(97)00105-9)

[11] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, “Acoustic event detection in real life recordings,” in *IEEE Eur. Sign. Process. Conf.*, pp. 1267-1271, Aalborg, Denmark, Aug. 2010.
(https://ieeexplore.ieee.org/abstract/document/7096611)

[12] Z. Kons, O. Toledo-Ronen, and M. Carmel, “Audio event classification using deep neural networks,” *Interspeech*, pp. 1482-1486, 2013. (chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.isca-speech.org/archive_v0/archive_papers/interspeech_2013/i13_1482.pdf)

[13] H. Phan, et al., “Audio scene classification with deep recurrent neural networks,” *arXiv preprint arXiv:1703.04770*, 2017.
(https://doi.org/10.48550/arXiv.1703.04770)

[14] I. Lezhenin, N. Bogach, and E. Pyshkin, “Urban sound classification using long short term memory neural network,” in *IEEE FedCSIS*, pp. 57-60, Leipzig, Germany, Sep. 2019.
(https://doi.org/10.15439/2019F185)

[15] C. Villanueva, J. Vincent, A. Slowinski, and M.-P. Hosseini, “Respiratory sound classification using long-short term memory,” *arXiv preprint arXiv:2008.02900*, 2020.
(https://doi.org/10.48550/arXiv.2008.02900)

[16] K. J. Piczak, “Environmental sound classification with convolutional neural networks,” in *IEEE 25th Int. Wkshp. Mach. Learn. for Sign. Process.(MLSP)*, pp. 1-6, Boston, MA, USA, Sep. 2015.
(https://doi.org/10.1109/MLSP.2015.7324337)

[17] J. Salamon and J. P. Bello, “Deep convolutional neural networks and data augmentation for environmental sound classification,” *IEEE Sign. Process. Lett.*, vol. 24, no. 3, pp. 279-283, 2017.
(https://doi.org/10.1109/LSP.2017.2657381)

[18] M. Huzaifah, “Comparison of time-frequency representations for environmental sound classification using convolutional neural networks,” *arXiv preprint arXiv:1706.07156*, 2017.
(https://doi.org/10.48550/arXiv.1706.07156)

[19] Y. Tokozume and T. Harada, “Learning environmental sounds with end-to-end convolutional neural network,” in *IEEE ICASSP*, pp. 2721-2725, New Orleans, LA, USA, Mar. 2017.
(https://doi.org/10.1109/ICASSP.2017.7952651)

[20] Y. Su, K. Zhang, J. Wang, and K. Madani, “Environment sound classification using a two-stream cnn based on decision-level fusion,” *Sensors*, vol. 19, no. 7, 1733, 2019.
(https://doi.org/10.3390/s19071733)

[21] S. Kahl, T. Wilhelm-Stein, H. Hussein, H. Klinck, D. Kowerko, M. Ritter, and M. Eibl, “Large-scale bird sound classification using convolutional neural networks,” *CLEF (working notes)*, vol. 1866, 2017.
(chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://www.researchgate.net/profile/Danny-Kowerko/publication/322144806_Large-Scale_Bird_Sound_Classification_using_Convolutional_Neural_Networks/links/5ad4bfe2a6fdcc2935808d8e/Large-Scale-Bird-Sound-Classification-using-Convolutional-Neural-Networks.pdf)

[22] Q. Chen, W. Zhang, X. Tian, X. Zhang, S. Chen, and W. Lei, “Automatic heart and lung sounds classification using convolutional neural networks,” in *IEEE APSIPA*, pp. 1-4, Jeju, Korea, Dec. 2016.
(https://doi.org/10.1109/APSIPA.2016.7820741)

[23] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, “Audio surveillance of roads: A system for detecting anomalous sounds,” *IEEE Trans. Intell. Transport. Syst.,* vol. 17, no. 1, pp. 279-288, 2015.
(https://doi.org/10.1109/TITS.2015.2470216)

[24] J. Lee, W. Kim, and K. Lee, “Convolutional neural network based traffic sound classification robust to environmental noise,” *The J. Acoustical Soc. Korea*, vol. 37, no. 6, pp. 469-474, 2018.

(https://doi.org/10.7776/ASK.2018.37.6.469)

[25] R. P. Ramachandran and R. Mammone, *Modern methods of speech processing*, Springer Science & Business Media, vol. 327, 2012.

[26] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," *ECCV*, pp. 122-138, 2018. (https://doi.org/10.1007/978-3-030-01264-9_8)

[27] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. CVPR*, pp. 4510-4520, Salt Lake City, UT, USA, Jun. 2018. (https://doi.org/10.1109/CVPR.2018.00474)

[28] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance of roads: A system for detecting anomalous sounds," *IEEE Trans. Intell. Transport. Syst.*, vol. 17, no. 1, pp. 279-288, 2015. (https://doi.org/10.1109/TITS.2015.2470216)

[29] P. Foggia, A. Saggese, N. Strisciuglio, and M. Vento, "Cascade classifiers trained on gammatonegrams for reliably detecting audio events," in *IEEE Int. Conf. Advanced Video and Signal-Based Surveillance*, Seoul, Korea, Aug. 2014. (https://doi.org/10.1109/AVSS.2014.6918643)

[30] N. Almaadeed, M. Asim, S. Al-Maadeed, A. Bouridane, and A. Beghdadi, "Automatic detection and classification of audio events for road surveillance applications," *Sensors*, vol. 18, 2018. (https://doi.org/10.3390/s18061858)

[31] P. S. Pariyal, D. M. Koyani, D. M. Gandhi, S. F. Yadav, D. J. Shah, and A. Adesara, "Comparison based analysis of different FFT architectures," *Int. J. Image, Graphics and Sign. Process.*, vol. 8, no. 6, 41, 2016. (https://www.proquest.com/docview/1886765558?pq-origsite=gscholar&fromopenview=true)

[32] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013. (https://doi.org/10.48550/arXiv.1312.4400)

[33] T. E. Oliphant, *A guide to NumPy*, vol. 1, Trelgol Publishing USA, 2006. (chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/https://ecs.wgtn.ac.nz/foswiki/pub/Support/ManualPagesAndDocumentation/numpybook.pdf)

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *3rd Int. Conf. for Learn. Representations*, San Diego, CA, USA, May 2015. (https://doi.org/10.48550/arXiv.1412.6980)

[35] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," *Advances in NIPS*, vol. 32, 2019. (https://doi.org/10.48550/arXiv.1912.01703)

이주영 (Juyoung Lee)

2010년 2월 : 연세대학교 중어중문학과 졸업

2012년 2월 : 연세대학교 경영학과 석사

2016년 9월~현재 : 연세대학교 응용통계학과 박사과정

<관심분야> Aritifical Intelligence, Signal Processing

[ORCID:0000-0001-6643-9284]

박천균 (Chunkyun Park)

2017년 2월 : 인하대학교 통계학과 졸업

2022년 8월 : 연세대학교 응용통계학과 박사

<관심분야> Aritifical Intelligence, Signal Processing, Sound Classification

[ORCID:0000-0002-3280-9416]

김현중 (Hyunjoong Kim)

1989년 2월 : 연세대학교 응용통계학과 졸업

1991년 2월 : 연세대학교 응용통계학과 석사

1998년 7월 : University of Wisconsin - Madison 박사

2003년 9월~현재 : 연세대학교 응용통계학과 전임 교수

<관심분야> 빅데이터, 머신러닝, 데이터마이닝, 딥러닝

[ORCID:0000-0001-6761-6318]