Weakly Labelled Sound Event Detection with a Capsule-Transformer Model

Kanghao Li[a], Shuguo Yang[a,*] Li Zhao[b], and Wenwu Wang[c]

Abstract

Sound event detection (SED) is a widely studied field that has achieved considerable success. The dynamic routing mechanism of capsule networks has been used for SED, but its performance in capturing global information of audio is still limited. In this paper, we propose a method for SED that by combining the capsule network with transformer leverages the strength of transformer in capturing global features with that of capsule network in capturing local features. The proposed method was evaluated on the DCASE 2017 Task 4 weakly labeled dataset. The obtained F-score and Equal Error Rate are 60.6% and 0.75, respectively. Compared to other baseline systems, our method achieves significantly improved performance.

Keywords: Sound event detection, audio tagging, gated convolution, transformer, capsule network.

1 Introduction

Sound Event Detection (SED) is a task that involves classifying sound events in an audio clip while determining their temporal boundaries. The main objective is to assign labels to detected events and identify their start and end time within the given audio clip. SED has attracted significant attention, with many potential applications, such as biological scene analysis [1, 2], speech recognition [3, 4], multimedia retrieval and analysis [5], among others.

Traditional models for sound event detection include Gaussian mixture models (GMM) trained on Mel-frequency cepstral coefficients (MFCC) [6], Hidden markov models [7], and dictionaries constructed using non-negative matrix factorization (NMF) [8, 9]. Early methods on sound event detection primarily focused on individual sound events, and when dealing with multiple sound events, it was challenging to extract effective features to separate overlapping sound events. This could result in a lack of reliability and accuracy in the identification and detection of these events. Hence, many deep learning-based methods have emerged to address this issue [10–13].

Deep Neural Networks (DNN)-based sound event detection methods, such as [14], often require a large number of strongly labeled audio samples [15, 16], where the sound event categories and their onset and offset time are annotated. Obtaining accurate and reliable annotations can be challenging in practice. On the other hand, weakly labeled sound event detection addresses this issue by using labels that only provide category information of sound events, but not specify their onset and offset time. This approach effectively mitigates the requirement of strongly labelled data.

Several deep learning models have been developed. For example, convolutional neural networks (CNN) have been used to learn audio features through translational invariance, eliminating the need for complex data reconstruction in sound event classification [17]. Recurrent neural networks (RNN) enhance the accuracy of audio classification and recognition by capturing relationships between preceding and subsequent audio frames through recurrent neurons. Combining the local shift invariance of CNN and the contextual modeling capability of RNN, convolutional recurrent neural networks (CRNN) have shown promising performance in sound event detection tasks [19].

In recent years, several methods have emerged to enhance the performance of sound event detection models. For instance, attention mechanisms are applied to SED in [20]. In this work, a weakly labeled SED model based on multiple instance learning (MIL) is established, where a two-step attention pooling mechanism is adopted to improve model training. By incorporating features obtained from CNN networks into local predictions in the time and frequency domains of audio events, this approach yields more accurate detection results compared to traditional methods for weakly labeled sound event detection. Furthermore, NMF has been combined with CNN to provide approximately strong labels for weakly labeled datasets used in sound event detection [20, 21]. The CNN-SAN-Transformer architecture [22] is introduced to replace CNN for extracting high-level features with a self-attention networ (SAN). This architectural modification effectively reduces model complexity while achieving higher prediction accuracy when compared to the CNN-Transformer architecture. In addition, ResNet and its variants were used in

[23], which significantly improves the system performance through multichannel spatial audio data augmentation.

Another approach is based on capsule networks (CN) [24] which offer the potential ability to accurately detect targets within overlapping features. In contrast to traditional neural networks, capsule networks accurately capture the contextual relationships among words in a sentence through dynamic routing [24]. This addresses the limitations of CNNs in representing feature angles, relative positions, and avoiding information loss caused by pooling. Moreover, CNs automatically adjust capsules to extract overlapping features, thereby enhancing the overall model's capability to recognize targets. Capsule networks have vector inputs and outputs, enabling the network, through the dynamic routing algorithm, to identify and establish relationships between different features. Recent research has shown promising results of CNs used for sound event detection [25]. In this research, gate convolutional networks are employed to extract features, which are then utilized by CN models for sound event detection and recognition [26]. The dynamic routing algorithm, serving as the core of CNs, can be considered an attention mechanism that learns and trains multiple attributes such as target shape and position while retaining crucial features. CN has also been applied to weakly labeled sound event detection [28], showing promising performance. The CN model is thus our focus in this paper.

Traditional capsule networks, however, suffer from low training efficiency due to the internal loops of their dynamic routing algorithm. In addition, CN is limited in capturing global feature of sound events which could potentially result in performance degradation. To address this issue, we propose a weakly labeled SED model based on capsule-transformer model. More specifically, we replace the traditional convolutional layers with parallel gated convolutional layers, effectively improving the training speed, and reducing model computation complexity, then we use transformer's encoder structure to extract audio features. In addition, in the capsule layer, inspired by the model in [26], we introduce a temporal attention (TA) layer, which employs temporal segments in the attention mechanism, thereby enhancing the overall performance of the model. We evaluate our proposed method on the DCASE 2017 Task 4 dataset [28]. Compared to the baselines, our method demonstrated a significant performance improvement. The main contributions are summarized below:

- We introduce the integration of the transformer model with the capsule model to improve the performance of the capsule model for sound event detection.
- We optimize a multi-layer parallel gated convolutional structures to improve the computational efficiency and detection accuracy of the proposed model.

2 Background

2.1 Capsule

Capsule networks [24] aim to overcome some of the limitations of traditional network structures, such as CNN. The overall framework of capsule networks, as shown in Fig. 1, can be divided into two parts: the encoding part, which comprises convolutional layers with rectified linear unit (ReLU) (e.g. ReLU Conv1), primary capsule layer (i.e. PrimaryCaps), and the second capsule layer (i.e. SecondCaps), and the decoding part, which includes multiple fully connected layers with nonlinear activation functions ReLU and Sigmoids (e.g. FC ReLU and FC Sigmoid). The encoder aims to take audio input (e.g. log-mel spectrograms) and generate more compact embeddings. In SecondCaps, the frame highlighted refers to a masked frame that system is learned to reconstruct.

The inputs and outputs of the neurons from traditional neural networks can only express the likelihood of extracted features without considering their spatial relationships. In contrast, capsule networks utilizes capsules as fundamental components [24], which consist of multiple neurons, with each neuron represented by a vector. Notably, both the inputs and outputs of these neurons are vectors, where the output value denotes the probability of entity existence within the range of 0 to 1. The magnitude and direction of these vectors correspondingly indicate the likelihood and attributes of the capsules.

Table 1 illustrates the disparities between vector neurons (VN) and scalar neurons (SN). In this table, x_i, $i = 1, 2, ..., n$, represents the input of a scalar neuron, w_i, $i = 1, 2, ..., n$, represents the corresponding weight, and b represents the bias. The variable u_i, $i = 1, 2, ..., n$ represents the lower-level capsule, while $\hat{u}_i$, $i = 1, 2, ..., n$ represents the prediction of the lower-level capsule for the higher-level capsule, $\sum$ denotes the summation operation on the inputs, c_{ij} represents the coupling coefficient between different layer vector elements, and s_j represents the input to the capsule vector of the current layer, which is the weighted sum of the prediction vectors. During the forward propagation process of vector neurons, different capsules interact with each other using the dynamic routing mechanism, following the algorithmic process in Table 2. During the forward propagation process of scalar neurons, the product of the input x_i and the weight is summed to form scalar a_i, which is then transformed into the output h_j through a non-linear function.

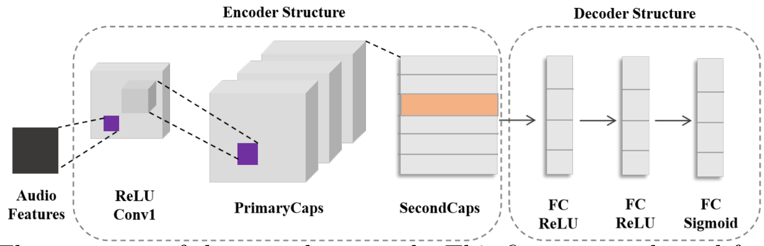

Fig. 1 The structure of the capsule network. This figure was adapted from [22].

Table 1 Differences between vector neurons (VN) and scalar neurons (SN)

		VN	SN
Input		u_i	x_i
Operations	Transformation	$\hat{u}_{j\|i} = W_{ij} u_i$	-
	Weighted summation	$s_j = \sum_i c_{ij} \hat{u}_{j\|i}$	$a_j = \sum_i w_i x_i + b$
	Nonlinear activation	$v_j = \frac{\|\|s_j\|\|}{1+\|\|s_j\|\|^2} \cdot \frac{s_j}{\|\|s_j\|\|}$	$h_j = g(a_j)$
Output		v_j	h_j

The dynamic routing algorithm aims to iteratively update the weight matrix connecting the capsule layers in order to select the detection capsules that exhibit high consistency with the primary capsule layer. This algorithm facilitates the matching of the primary capsule, which represents sound features, with the secondary capsule layer, which represents event categories. The calculation process is outlined below:

$$\hat{u}_{j|i} = W_{ij} u_i \tag{1}$$

$$s_j = \sum_i c_{ij} \hat{u}_{j|i} \tag{2}$$

$$v_j = \frac{||s_j||}{1 + ||s_j||^2} \cdot \frac{s_j}{||s_j||} \tag{3}$$

where $\hat{u}_{j|i}$ represents the prediction from u_i to v_j, W_{ij} indicates the corresponding weight matrix, and v_j represents the output vector of capsule j. The vector s_j undergoes a squash non-linear function for compression and normalization, resulting in v_j of unit-norm.

$$c_{ij} = \frac{exp(b_{ij})}{\sum_k exp(b_{ik})} \tag{4}$$

where the parameter b_{ij} is used for updating the coupling coefficient, with initial value typically set to 0, as illustrated by the following equation,

$$b_{ij} \longleftarrow b_{ij} + \hat{u}_{j|i} \cdot v_j \tag{5}$$

During each forward propagation process, the value of v_j is computed based on b_{ij}. The optimal coupling coefficient is eventually obtained through iterative updates to b_{ij} and subsequent updates to c_{ij}.

2.2 Transformer

The transformer was initially proposed by the Google team in 2017 [29] as a sequence-to-sequence model for machine translation. Different from CNN and RNN, it employs a self-attention mechanism to establish global contextual information and represents input data using positional encodings. As a result, the transformer enables more parallel computations, leading to significant performance improvements compared to traditional network structures.

The transformer architecture consists of multiple encoder and decoder layers. Both the encoder and decoder are comprised of N identical layers, each utilizing residual connections and layer normalization. The encoder takes input features and converts them into high-level embeddings, which are then transformed by the decoder to generate the output.

Each encoder primarily consists of a multi-head self-attention (MSA) module and a position-wise feed-forward network (FFN). To enable deeper models, residual connections are applied to each module, followed by layer normalization (LN). In contrast, the decoder includes an additional cross-attention (CA) module between the MSA and FFN modules.

In SED, the events often involve multiple occurrences within an audio clip. For instance, in a traffic environment, car honking sounds can appear at any time within the audio recording. By leveraging the attention mechanism of the transformer (scaled dot-product attention), information from different time points in an audio clip can be effectively captured. For SED, only the encoder is required. Each encoder is composed of multiple layers, and the input to each layer undergoes processing through the MSA mechanism. The input vectors are transformed into outputs using query, key, and value transformation matrices. In this case, we adhere to the notation format from [23], where the input is represented as a $T \times C$ matrix, with matrices W^Q and W^K having shapes of $C \times d_k$, and matrix W^V having a shape of $C \times d_v$. Here, d_k and d_v are in-

Table 2 The description of Dynamic Routing Algorithm

Dynamic Routing Algorithm	
Input:	$\hat{u}_{j\|i}$, r, l
Output:	layer $(l+1)$ capsule v_j
Step 1	for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \longleftarrow 0$
Step 2	for r interations do
Step 3	for all capsule i in layer $(l+1)$: $c_{ij} \leftarrow Softmax(b_{ij})$
Step 4	for all capsule i in layer $(l+1)$: $s_j \leftarrow \sum_i c_{ij}\hat{u}_{j\|i}$
Step 5	for all capsule i in layer $(l+1)$: $v_j \leftarrow Squash(s_j)$
Step 6	for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow b_{ij} + \hat{u}_{j\|i} \cdot v_j$
Step 7	end for

tegers, and Q, K, and V can be obtained from the following formula.

$$Q = xW^Q \tag{6}$$

$$K = xW^K \tag{7}$$

$$V = xW^V \tag{8}$$

The structural diagram of the transformation matrices is depicted in Fig. 2, where q_i, k_i, v_i represent the query, key, and value vector, respectively. The formula for the attention mechanism is defined as follows.

$$Attention(Q,K,V) = (\frac{QK^T}{\sqrt{d_k}})V \tag{9}$$

where the shape of $Attention(Q,K,V)$ is $T \times d_k$, indicating that the attention mechanism calculates softmax functions on the vectors Q, K, and V. This step involves transforming the related vector groups into probabilities along the temporal steps. In the equation above, Q, K, and V represent the feature correlations at different time steps, with a shape of $T \times T$. We utilize $\sqrt{d_k}$ to perform the scaling operation. The operational flowchart is shown in Fig. 3 [29].

The MSA mechanism divides Q, K, and V into h heads, enabling parallel computation of the input x and its similarity with other inputs. The outputs are then concatenated, leading to a significant improvement in the computational efficiency of the model. In the parallel computation, we perform matrix multiplication between the input x_i and weight matrices W_i^Q, W_i^K and W_i^V. Using the obtained Q, K, and V matrices, we calculate the attention. The resulting matrices are concatenated and multiplied by the weight matrix W^O to obtain the output of the encoding layer, as follows:

$$MulHead(Q,K,V) = Concat(head_1, \dots, head_h)W^O \tag{10}$$

$$head_i = Attention(xW_i^Q, xW_i^K, xW_i^V) \tag{11}$$

where $head_i$ represents the attention from the i-th head.

The feed-forward layer in the transformer encoder section essentially consists of a multi-layer perceptron (MLP) with a linear structure and a convolutional structure. It utilizes the Gaussian error linear units (GELU) and linear activation functions and can be obtained from the following formula, where x is the output from the previous layer, and W and b are the learning parameters.

$$GELU(x) = 0.5x(1 + tanh(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3))) \tag{12}$$

$$FFN(x) = max(0, xW_1 + b_1)W_2 + b_2 \tag{13}$$

2.3 Applications to sound event detection

The transformer model has demonstrated excellent performance in audio classification [22]. Compared to traditional neural network detection models, the self-attention mechanism in the transformer captures long-range dependencies and mitigates issues of gradient vanishing or exploding. Moreover, the model structure in transformer is highly adaptable, allowing for flexible adjustments tailored to specific tasks.

The capsule model has gained significant attention in the audio domain. To deal with overlapping sound events, capsule networks utilize their dynamic routing mechanism to gather diverse information related to the temporal and spatial aspects of audio features. These networks perform well at learning representations from limited data, compensating for information loss that occurs in traditional neural network structures during training. However, using the capsule network model alone also has some limitations, such as a low training speed and degraded performance when dealing with complex audio datasets. The self-attention mechanism of transformer is useful for feature extraction in complex datasets, such as polyphonic audio event datasets.

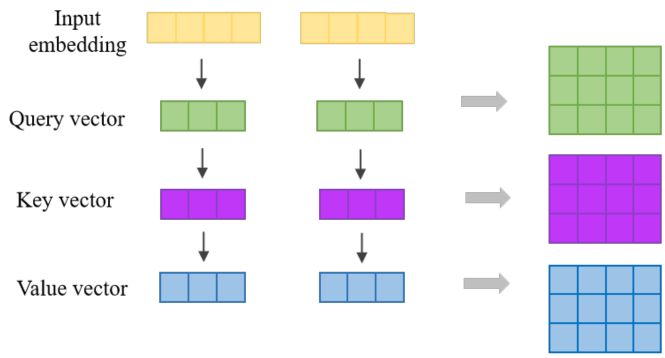

Fig. 2 Structure diagram of transformation matrix.

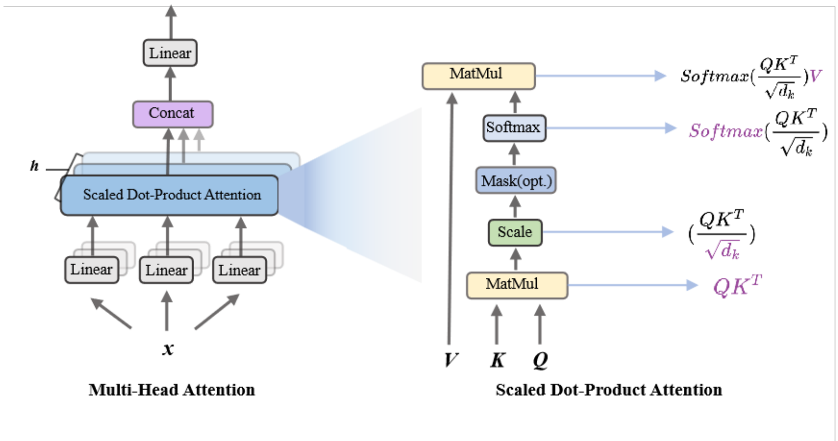

Fig. 3 Implementation process of multi-head self attention mechanism. This figure was adapted from [29].

In this study, we propose the capsule-transformer fusion model by utilizing the encoder of the transformer to extract features and integrating them with the capsule network model. First, we employ gated convolution to extract features from the input logmel spectrogram, generating embedding vectors along the time axis through feature mapping in the convolution. We then pass these vectors to the transformer for refined feature representation with improved global information. The aforementioned capsule network is applied to the output of the transformer, incorporating an improved dynamic routing mechanism to predict the probability of the existence of a sound class. This approach facilitates audio feature extraction, thereby improving performance in sound event detection.

3 Proposed Method

In this section, we present a method of integrating these architectures for polyphonic sound event detection using weakly labeled data. The collection of strongly labeled data is a time-consuming task in traditional SED methods, due to the substantial effort required for annotation. Consequently, our approach leverages a weakly labeled dataset.

3.1 Model architecture

The proposed model architecture is illustrated in Fig. 4, which consists of three parts: the gated convolutional layer, the transformer layer, and the capsule layer. The first two parts are used for feature extraction, while the third part is employed for classification and detection of acoustic events. In the convolutional layer, we use gated convolution [30], with which a dynamic feature selection mechanism can be applied to each channel and spatial position, enabling local feature selection for different audio instances. Three parallel gated convolutional neural network blocks are utilized to extract information from the input features. Each parallel block consists of three convolutional layers. After each block, a two-dimensional max pooling (Max Pool) is applied along the frequency axis for dimension reduction, while the time axis remains unchanged to match the target length. In addition, the convolutional kernels of same size are used within the convolutional layers to extract information from input features.

The encoder part of the transformer is incorporated following the convolutional layers. This addition helps the system to capture global information within audio signals. The encoder structure consists of self-attention and feed-forward layers. Section 2.2 outlines the self-attention mechanism, and

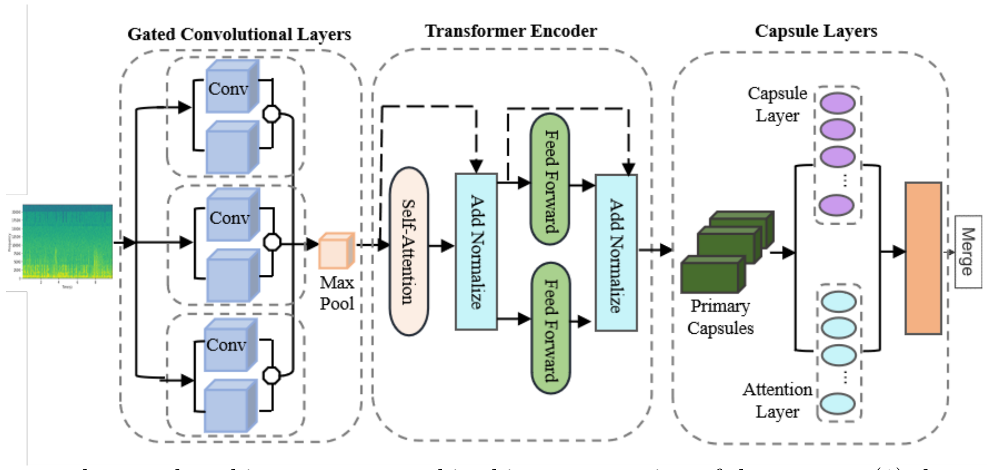

Fig. 4 The neural network architecture proposed in this paper consists of three parts: (1) the parallel gated convolutional layer, (2) the transformer encoder layer, and (3) the improved capsule layer. The traditional capsule layer and the temporal attention (TA) layer are learned in parallel to estimate the probability of the entity represented by the capsule.

the residual connections and normalization incorporated after the self-attention layer to improve convergence speed. Subsequently, the features are input to the feed-forward neural network layer, where a fully connected network with the GELU activation function is applied to enhance the model's generalization capability. Then, residual connections and normalization are applied before the final output.

In the capsule layer, we utilize an improved capsule network structure, incorporating a temporal attention layer, and compute the output in parallel with the second capsule layer. The introduction of this layer effectively addresses the problem of reduced model performance caused by background noise in audio data, especially in complex datasets [26]. The features are input to the primary capsule layer with ReLU activation. After reshaping, the output becomes individual time slices, which are considered as separate inputs for subsequent layers. The time slices are then passed to the second capsule layer and a temporal attention (TA) layer. In the second capsule layer, a dynamic routing algorithm is used to train the features and calculate the output. In contrast to the original capsule routing mechanism, the TA layer, inspired by the attention schemes outlined in [31, 32], employs the attention weights on the audio frames, i.e. attending the vital frames while attenuating irrelevant ones. Finally, the outputs of the second capsule layer and the TA layer are merged to obtain the predicted values of the data features. These predicted values can be seen as the expected length of the capsules relative to the probability distribution derived from the TA layer. Experimental results demonstrate that using the TA layer yields better performance compared to the original routing mechanism.

3.2 Parallel gated convolutional layer

We incorporate three parallel paths of gated convolution. Gated convolution allows for the automatic learning of soft masks from the data, as demonstrated by the following formula:

$$Gating_{y,x} = \sum \sum W_g \cdot I \tag{14}$$

$$Feature_{y,x} = \sum \sum W_f \cdot I \tag{15}$$

$$O_{y,x} = \emptyset(Feature_{y,x}) \odot \sigma(Gating_{y,x}) \tag{16}$$

where the subscripts x and y denote the coordinates of each channel in the input features, W_g represents the convolution kernel that operates on the input to generate the soft mask, W_f represents the convolution kernel that operates on the input to generate the feature map, and σ represents the sigmoid activation function applied to the outputs in the gated convolution. The soft mask, activated by this function, ranges between 0 and 1. Finally, $\emptyset$ represents the activation function applied after the convolution, and we use ReLU for $\emptyset$ in this paper. Fig. 5 illustrates the comparison between traditional partial convolution and gated convolution. In the case of partial convolution, the ReLU Update represents convolving features by updating the mask.

In each pathway, we perform three convolution operations using 128 filters, which consist of 64 linear filters and 64 sigmoid filters, with a stride of 1. We extract features by employing symmetric convolution kernels of size 3. After each convolutional block, we apply 2×2 average pooling to extract high-level features. The input feature has a shape of $T \times F$, where T represents the number of time frames, and F represents the number of frequency bins in the input feature. The output dimension

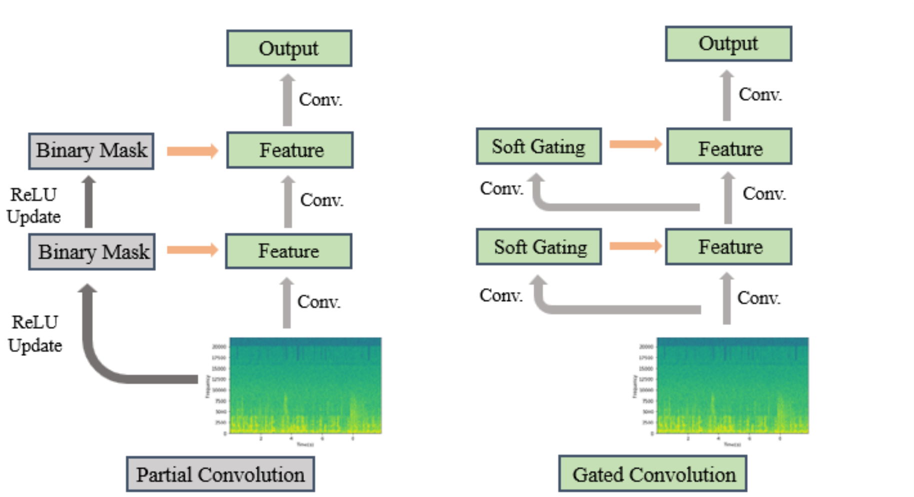

Fig. 5 Illustration of the partial convolution (left) and the gated convolution (right).

of the convolutional layer is $T_1 \times F_1 \times M$, with M representing the number of feature maps obtained after concatenating the outputs of the three parallel convolutional blocks. The values T_1 and F_1 correspond to the number of time frames and frequency bins, respectively, after feature extraction in the gated convolutional layer.

3.3 Transformer encoder

The transformer architecture we employ is illustrated in Fig. 6. In this structure, we employ 3 layers of encoders. After feature extraction in the convolutional layer, the data is initially passed to the self-attention module within the encoder structure. This module, comprising a linear layer and a single-headed attention mechanism, captures the inter-dependencies among features. At this stage, the dimensions of the data representation is $B \times N \times T_1$, where B denotes the batch size, N represents the sequence length of the input features, and T_1 is the dimension of each input vector. The output of the multi-head attention is subsequently normalized using layer normalization, preserving a dimension of $B \times N \times T_1$. Subsequently, the output is fed into the feed-forward neural network layer, which comprises two fully connected layers separated by an activation function. The first fully connected layer reduces the dimension to $B \times N \times 2T_1$, while the second fully connected layer restores it to $B \times N \times T_1$. Following another round of layer normalization, the output is directed to the capsule layer.

3.4 Capsule layer

The structural flowchart of the capsule layer we have used is shown in Fig. 7. The first layer of the capsule layer is the primary capsule layer, which is essentially a ReLU convolutional layer. The output from the transformer layer is first passed to the primary capsule layer. The output features are reshaped into a tensor of size $T_1 \times \cdot \times U$ and compressed [33]. Here, T_1 represents the time dimension before reshaping, and U denotes the capsule size, which is set to 4 in our case. This layer uses 64 filters with a kernel width of 3, and the time and frequency dimensions are set to 1 and 2, respectively.

The time slices after the output of the primary capsule layer are passed to the second capsule layer and the TA layer. Within the capsule layer, the output is calculated using the inter-layer dynamic routing mechanism, with $U = 8$. The length of each output vector is computed, and $o(t) \in \mathbb{R}^L$ is used to represent the activation vector for each time slice t. The TA layer is connected to L units and a sigmoid activation function, resulting in an output of $z(t) \in \mathbb{R}^L$, where L represents the number of classes (sound events). Finally, for class l, we combine $o(t)$ and $z(t)$ as follows [26]:

$$y_l = \frac{\sum_{t=1}^{T} o_l(t) z_l(t)}{\sum_{t=1}^{T} z_l(t)} = E_{t\sim q_l(t)}[o_l(t)] \tag{17}$$

where $q_l(t) = softmax(logZ_l)$, $\mathbb{Z}_l \in \mathbb{R}^T$ and $\{z_l(t)\}_{t=1,\dots,T}$. We select a probability threshold τ_1 for the constructed time slices [26]. If the final prediction y_l is greater than the specified threshold τ_1, it indicates the presence of the sound event. Otherwise, it is considered as absence of the sound event. In addition, we set a threshold for the probability of τ_2 with respect to $o_l(t)$ to calculate the onset and offset time. To mitigate noise, we employ morphological closing operations, which involves processing the regions of interest through convolution, utilizing their starting and ending points to determine the onset and offset time.

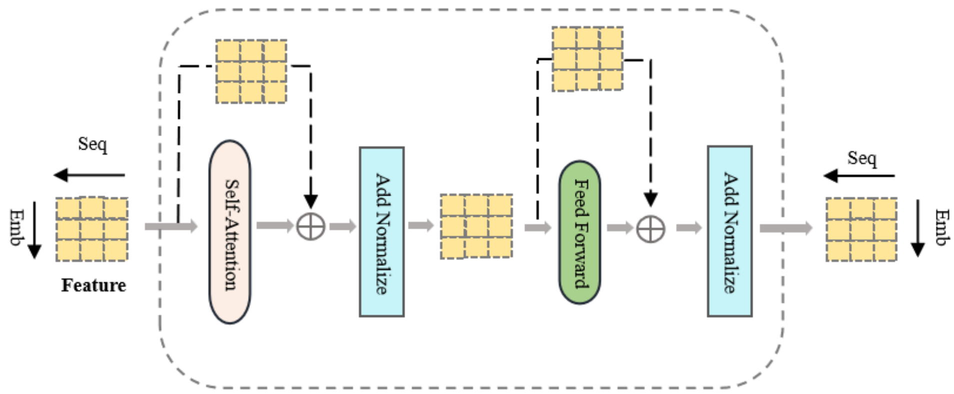

Fig. 6 The network architecture of the transformer encoder.

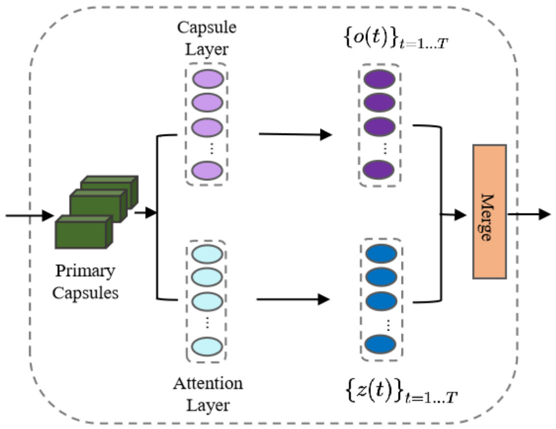

Fig. 7 The network architecture of the capsule layer.

4 Experiments

4.1 Dataest

Since our proposed method focuses on weakly labeled polyphonic event detection, we conducted an evaluation using the DCASE 2017 Task 4 dataset titled "Large-scale weakly supervised sound event detection for intelligent vehicles". This dataset is a subset of Google AudioSet, encompassing 17 sound events classified into two categories: "Warning" and "Vehicle". We selected this dataset due to its extensive nature, encompassing more than 140 hours of weakly labeled audio data segments that cover a wide range of environmental sounds. The dataset is divided into three subsets: a training subset with 51,172 audio clips, a validation subset with 488 audio clips, and an evaluation subset with 1,103 audio clips. The majority of the audio segments have a duration of 10 seconds.

To assess the performance of these tasks, we utilized metrics such as precision, recall, and macro-averaged F-score. Additionally, for the SED task, we calculated the frame-level error rate at a one-second time resolution. The sed_eval toolbox [32] was employed for evaluating the SED task.

4.2 Baseline system

We conducted a comparative analysis of our proposed method with the following baseline systems:

GCCaps [26]: refers to gated convolution capsule. This system comprises three gated convolutional network blocks, two capsule layers, and a TA layer that is run in parallel to the high-level capsule layer. Normalization is applied after each gated convolutional layer and the primary capsule layer. Each convolutional block consists of three gated convolutional layers.

GCRNN [35]: refers to gated convolutional recurrent neural network. In this system, the ReLU activation function after each audio classification layer of the CNN is replaced with learnable gated linear units.

GCNN [26]: refers to gated CNN. This system is similar to the GCRNN model [35] as it replaces traditional convolutional neural networks with gated convolutions. However, it does not include recurrent layers.

CNN-transformer [36]: refers to an integrated CNN and transformer model. This system consists of four convolutional blocks, each containing two convolutional layers. Normalization and ReLU non-linearity are applied after each convolutional layer. The model utilizes the Adam optimizer with a learning rate of 0.001 and incorporates mixup with an alpha value of 1 to mitigate overfitting during training. The final output is obtained by averaging the frequency-axis output of the last convolutional layer and predicting the presence probability of sound events for each time frame using a fully connected layer with sigmoid non-linearity.

4.3 Experimental setup

Prior to feature extraction, we employed mel spectrograms as input features. Each audio clip was resampled to 16 kHz and subjected to mel filter banks and a logarithmic non-linearity operation. Log mel features were computed with a frame length of 64 ms, a 20 ms overlap, and mel frequency units per frame. Consequently, a feature vector of size 240 × 64 was generated for each audio sample.

Tables 3 and 4 provide the hyperparameters used at different stages of the model, where "Tf" refers to the transformer and the number following Tf indicates the index of transformer layer. Within each gated convolutional network section, we utilized 64 filters of size 3×3. The pooling size for both the audio tagging subtask and the sound event detection subtask was set to 2×2. To address overfitting and expedite convergence, we used batch normalization after each convolutional layer and the primary capsule layer. In the transformer structure, the data feature input to the encoder had a sequence length of 64, and the vocabulary size was 3840. We employed an encoder structure with one attention head.

For optimization, we employed the Adam optimizer [37] as the gradient descent algorithm, maintaining a fixed learning rate of 0.001. The routing iteration was set to 4, and the learning rate was decayed by a factor of 0.9 every two epochs. Binary cross-entropy was used as the loss function, and gradients were calculated accordingly. The mini-batch size was set to 44, and we trained the system for a total of 30 epochs.

To mitigate the issue of significant class imbalance within the dataset, we implemented data balancing techniques as suggested in [32]. This ensured that our training, testing, and evaluation sets encompassed samples from each class of the audio dataset, thereby preventing classification bias.

During the inference process, we averaged the predictions from the top five epochs, based on their highest accuracy on the validation set, to obtain the final result. In our system, the detection thresholds for sound event detection (SED) were set to 0.3 and 0.6. Additionally, we set the expansion and corrosion sizes for SED to 10 and 5, respectively. These hyperparameters were determined through experiments conducted on the validation set.

Apart from the SED results, we also show the audio tagging results, by aggregating the detection results over the whole signal. Audio tagging is a multi-label classification problem by identifying the audio classes from the audio clip, while the SED task focuses on detecting the presence or absence of target sound events in continuous audio recordings. With the SED results, it is straightforward to obtain the tagging results, by dropping the information related to onset/offset time of the sound events.

4.4 Comparative experiment

In this section, we conducted comparative experiments between the GCCaps model mentioned in [26] and the proposed model in this paper. Specifically, we focused on the case where the convolutional layer has a size of 3. The comparison graph of different metrics including F1 Score and Precision at batch_size 30-44 is shown in Fig. 8. It is clear that our proposed model achieved higher F1 score and precision.

To further demonstrate the effectiveness of the feature extraction part in our model and highlight the differences between our proposed model and the baseline models, we provide t-distributed stochastic neighbor embedding (t-SNE) cluster visualizations of the feature extraction outputs from both the baseline system and our proposed system. t-SNE is an unsupervised nonlinear technique [38] widely employed in various fields, including image and audio analysis. Its primary purpose is to visualize high-dimensional data by mapping it to a lower-dimensional space, thereby observing the relationships between data points. In the t-SNE algorithm, similarity in the high-dimensional space is represented by a Gaussian distribution, while similarity in the low-dimensional space is represented by a t-distribution. The closer the points are, the higher their similarity.

We trained both models using the same training samples, and the results are depicted in Fig. 9. From the figure, it can be observed that our proposed model exhibits denser clusters and higher similarity among samples of the same class compared to the baseline model. This suggests that it can extract features with greater accuracy for samples with ambiguous characteristics. In other words, the proposed network architecture can better identify samples based on their distinctive features.

In addition to our system, we also evaluated the GCCaps model proposed in [24], along with GCRNN and GCNN, as part of a comparative study for ablation experiments. Fig. 10 compares the loss for different epochs of the four models against the different baseline systems. It can be observed that our proposed model has lower loss compared to the other models.

4.5 Results and discussion

Table 5 presents the F-score, accuracy, and recall of different methods on the evaluation set for the audio tagging task. In the audio tagging task, our proposed system achieved an F-score of 60.6% on the evaluation set, surpassing other methods in the same task. The fusion of the transformer and capsule models yielded the best performance, slightly outperforming the use of the GC-

Table 3 Model parameters (feature extraction)

	Feature extraction					
	Conv1	**Tf1**	**Conv2**	**Tf2**	**Conv3**	**Tf3**
Kernel size	64@3×3	-	64@3×3	-	64@3×3	-
Stride	1×1	-	1×1	-	1×1	-
Pooling size	2×2	-	2×2	-	2×2	-
Num_head	-	1	-	1	-	1
Dropout rate	0.2	0.3	0.2	0.3	0.2	0.3
Activation function	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU

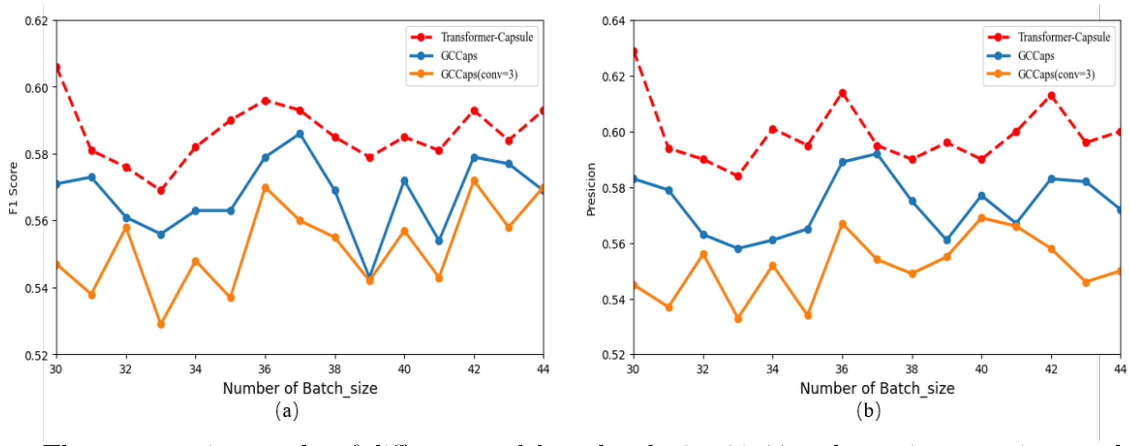

Fig. 8 The comparative graphs of different models at batch_size 30-44 under various metrics are shown. (a) represents the comparison among the three models based on the F1 Score metric, while (b) represents the comparison among the three models based on the Precision metric.

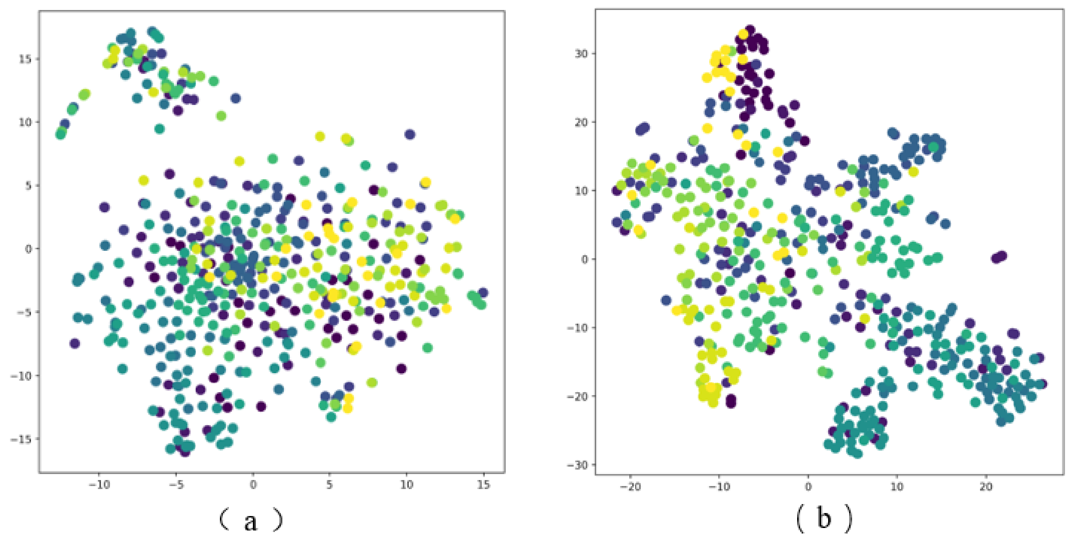

Fig. 9 The t-SNE visualization of the output features from different models. (a) represents the feature output of the gated convolutional layer in the GCCaps model, and (b) represents the feature output of the encoder layer in the proposed Transformer-Capsule model.

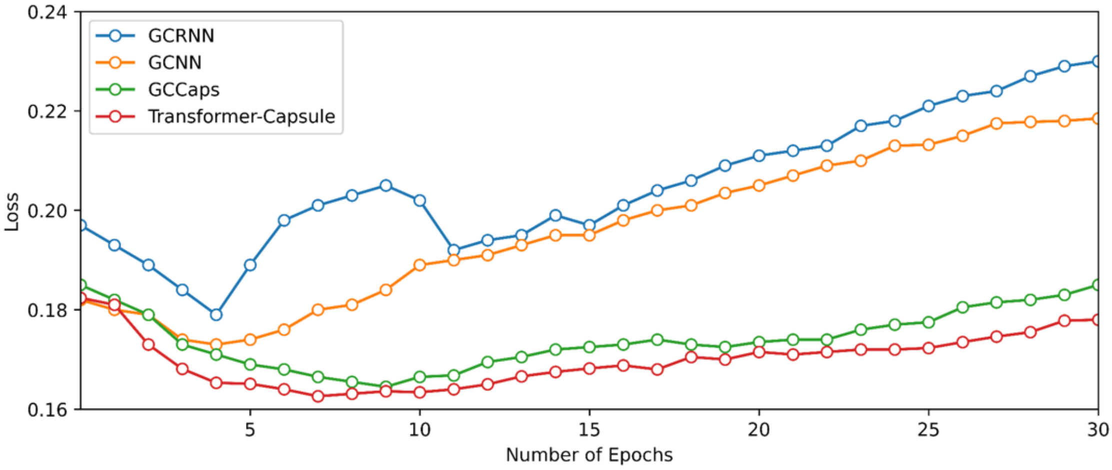

Fig. 10 Comparison of the loss function at different epochs for four different models including the proposed method and three baseline systems. The proposed Transformer-Capsule model exhibits the lowest loss and minimal deviation.

Table 4 Model parameters (capsule layers)

	Capsule layers	
	Primary capsule layer	Second capsule layer
Kernel size	32@3×3	-
Stride	1×1	-
Dropout	0.5	-
Activation function	Squashing	Squashing
Capsule dimension	8	16

Caps model. GCRNN and GCNN demonstrated comparable performance in this subtask. However, the CNN-Transformer model had the lowest F-score of 55.7%.

Table 6 presents the F-score, accuracy, recall, and error rate of different methods on the evaluation set for the sound event detection task. For the sound event detection subtask, the fusion of the transformer and capsule models achieved the lowest error rate of 0.75 and a good F-score of 47.9%, slightly outperforming the GCCaps model. The performance of GCCaps was slightly better than that of GCRNN, with an F-score of 46.3% and an error rate of 0.76. The inclusion of recurrent layers enhanced the temporal localization ability of the GCRNN model, as its score was significantly higher than that of GCNN, and its error rate was relatively low. Although the CNN-Transformer model had the highest F-score, its error rate was higher at 0.91.

Table 7 presents the F-scores of various events in the audio tagging subtask achieved by our proposed model, while Table 8 shows the error rates of various events in the sound event detection subtask. For the audio tagging subtask, events such as "Civil defense siren" and "Screaming" exhibited higher classification accuracy, while events like "Car passing by" and "Bus" demonstrated lower classification accuracy. In the sound event detection subtask, events such as "Civil defense siren" and "Train horn" had lower error rates, while events like "Bicycle" and "Truck" had higher error rates.

Table 5 Different performance results of audio tagging subtask

Method	F score	Precision	Recall
Transformer-Capsule	**60.6%**	62.9%	57.6%
GCCaps	58.6%	59.2%	59.6%
GCRNN	57.3%	53.6%	59.6%
GCNN	57.2%	59.0%	57.2%
CNN-Transformer	55.7%	55.4%	56.1%

Table 6 Different performance results of sound event detection subtask

Method	F score	Precision	Recall	Error rate
Transformer-Capsule	47.9%	68.7%	29.1%	**0.75**
GCCaps	46.3%	58.3%	38.4%	0.76
GCRNN	43.3%	57.9%	34.8%	0.79
GCNN	37.5%	46.6%	31.1%	0.88
CNN-Transformer	48.3%	-	-	0.91

To better observe the accuracy and relevance of the model, we conducted a paired-sample t-test between the baseline model GCCaps and the proposed model to compare the differences between the two sets of samples. The formula is as follows:

$$t = \frac{M_d - 0}{s_d/\sqrt{n}} \sim t(n-1) \tag{18}$$

where M_d represents the mean of the differences between samples, s_d represents the standard deviation of the differences between samples, n is the number of differences, n_d represents the sample size, and the t-statistic follows the t-distribution with degrees of freedom $n-1$. With a threshold set at $p = 0.05$, when $|t| > t_{\frac{\alpha}{2},n-1}$, we reject the null hypothesis and conclude that there is a significant difference

Table 7 F score of audio tagging subtask for each event

Train horn	Air horn, Truck horn	Car alarm	Reversing beeps	Bicycle	Skateboard	Ambulance	Fire engine, fire truck	Civil defense siren
61.1%	62.2%	66.0%	45.0%	49.6%	65.5%	50.4%	57.4%	82.0%
Police car	Screaming	Car	Car passing by	Bus	Truck	Motorcycle	Train	Micro average
48.1%	87.6%	65.7%	30.1%	43.5%	53.5%	58.9%	76.8%	60.6%

Table 8 Error rate of sound event detection subtask for each event

Train horn	Air horn, Truck horn	Car alarm	Reversing beeps	Bicycle	Skateboard	Ambulance	Fire engine, fire truck	Civil defense siren
0.66	0.71	0.67	0.79	1.20	0.89	0.88	0.93	0.31
Police car	Screaming	Car	Car passing by	Bus	Truck	Motorcycle	Train	Micro average
0.9	0.68	0.93	1.00	1.04	1.05	0.72	0.67	0.75

between the two sets of samples representing the overall results. Calculating the value of t as -0.383, we looked up the corresponding t-value in the t-table using the degrees of freedom and found that the calculated t-value is greater than the value in the table. Therefore, we reject the null hypothesis, indicating that there are significant differences in the results obtained by the two methods.

In the proposed model, we incorporated the improved capsule network model proposed in [26] and introduced the encoder structure of the transformer. In the experiments, we found that this fusion method can effectively improve the performance of the model on the test and evaluation sets. Specifically, the introduction of parallel gated convolution with symmetric convolutional kernels allows for effective utilization of the original feature information in the data, thereby improving the performance of model. At the same time, using the transformer to extract features from the input at a higher level reduces the computational complexity of the model and improves its overall performance. Finally, the use of capsule routing mechanism and attention mechanism enables the model to recognize the correlation between parts and wholes, enhancing its generalization ability, and also effectively suppresses background noise and mitigates potential overfitting issues, thereby improving the overall performance of the model.

We also referenced the asymmetric kernel convolutional neural network mentioned in [39] and tested the performance of convolutional network models with different kernel sizes. Ultimately, we found that selecting a symmetric convolutional kernel with a size of 3×3 yielded the best model performance. We also conducted experiments comparing different numbers of layers in the gated convolution and found that the model performed better when the number of layers was 3. It is worth noting that although incorporating the transformer encoder into the traditional capsule model has improved the performance, its model size in terms of the parameter count has also been increased to 523,873, which is higher than that of the GCCaps model, with a parameter count of 448,225.

While the proposed model has shown improvement, there is still a gap compared to the performance of the CNN-Transformer model proposed in [36]. This disparity arises from the fact that the threshold used in this study is a fixed value, instead of an automated threshold optimization system used in [36]. The performance of the proposed model is similar to that of the CNN-Transformer model, if a fixed threshold is used in both models. Therefore, in future research, we will focus on optimizing and improving the transformer aspect and the routing mechanism to achieve better detection performance.

5 Conclusion

This paper has presented a new method for polyphonic sound event detection based on the Capsule-Transformer network, building upon previous research. Firstly, we employ parallel gated convolutions to extract features at different frequencies, then a transformer encoder to extract features at a higher level. Then, we use the attention layers within the traditional capsule network to merge weights and generate final predictions.

The proposed system is evaluated using the weakly labeled dataset of the DCASE 2017 Challenge Task 4. It demonstrates superior performance in both the sound event detection subtask, with an error rate of 0.75, and the audio tagging subtask, achieving an F-score of 60.6%, as compared with baseline systems. Our future research directions include exploring more effective methods to improve the routing mechanism, aiming to enhance the model's training speed and efficiency. Furthermore, we will investigate techniques for feature augmentation to enhance the model's robustness.

Abbreviation

SED: Sound event detection
AT: Audio tagging
DNN: Deep neural networks
GNN: Gaussian mixture models
MFCC: Mel-frequency cepstral coefficients
NMF: Nonnegative matrix factorization
CNN: Convolutional neural networks

CRNN: Convolutional recurrent neural networks
GCCaps: Gated convolution capsule

Author affiliations

Kanghao Li: College of Mathematics and Physics, Qingdao University of Science and Technology, 99 Songling Road, Qingdao 266061, China.
Shuguo Yang: College of Mathematics and Physics, Qingdao University of Science and Technology, 99 Songling Road, Qingdao 266061, China.
Li Zhao: Qingdao Hospital of Traditional Chinese Medicine, China.
Wenwu Wang: Center for Vision Speech and Signal Processing, Department of Electrical and Electronic Engineering, Faculty of Engineering and Physical Sciences, University of Surrey, Guildford GU2 7XH, UK.

References

[1] Z. Zhao, S. Zhang, Z. Xu, K. Bellisario, N. Dai, H. Omrani, and B. C. Pijanowski, Automated bird acoustic event detection and robust species classification. Ecological Informatics, 2017, pp. 99-108.

[2] R. Abinaya, Acoustic based scene event identification using deep learning CNN, Turkish Journal of Computer and Mathematics Education (TURCOMAT), 2021, pp. 1398–1405.

[3] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, Attention-based models for speech recognition, in Advances in Neura Informati-on Processing Systems, 2015, pp. 577-585.

[4] J. Xu, J. Zhu, and Y. Yang, Disappeared command: spoofing attack on automatic speech recognition systems with sound masking, arXiv: 2204.08977, 2022.

[5] G. Ning, Z. Zhang, X. Ren, H. Wang, and Z. He, Rate-coverage analysis and optimization for joint audio-video multimedia retrieval, International Conference on Acoustics, Speech, and Signal Processing, 2017.

[6] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen, Audio context recognition using audio event histograms, in 2010 18th European Signal Processing Conference, 2010, pp. 1272-1276.

[7] N. Degara, M. E. P. Davies, A. Pena, and M. D. Plumbley, Onset event decoding exploiting the rhythmic structure of polyphonic music, in IEEE Journal of Selected Topics in Signal Processing, 2011, pp. 1228-1239.

[8] O. Dikmen and A. Mesaros, Sound event detection using non-negative dictionaries learned from annotated overlapping events, in 2013 IEEE Workshop Applications Signal Process. Audio Acoustics, Oct. 2013, pp. 1-4.

[9] V. Bisot, S. Essid, and G. Richard, Overlapping sound event detection with supervised nonnegative matrix factorization, in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Mar. 2017, pp. 31-35.

[10] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, Detection and classification of acoustic scenes and events, in IEEE Transactions on Multimedia, 2015, pp. 1733-1746.

[11] A. Mesaros, T. Heittola, and T. Virtanen, A multi-device dataset for urban acoustic scene classification, Workshop Detect. Classific. Acoust. Scenes Events, 2018.

[12] F. Font, G. Roma, and X. Serra, Sound sharing and retrieval, in Computational Analysis of Sound Scenes and Events, 2018, pp. 279-301.

[13] S. Krstulovi, Audio event recognition in the smart home, in Computational Analysis of Sound Scenes and Events, 2018, pp. 335-371.

[14] P. Foster and T. Heittola, DCASE2016 baseline system, IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE 2016)challenge. [Online]. Available: https://github.com/pafoster/dcase2016_task4/tree/master/baseline

[15] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, Convolutional recurrent neural networks for polyphonic sound event detection, IEEE Press, 2017, pp. 1291-1303.

[16] A. Mesaros, T. Heittola, and T. Virtanen, Tut database for acoustic scene classification and sound event detection, in 2016 24th European Signal Processing Conference (EUSIPCO), 2016, pp. 1128-1132.

[17] M. Valenti, S. Squartini, A. Diment, G. Parascandolo, and T. Virtanen, A convolutional neural network approach for acoustic scene classification, in 2017 International Joint Conference on Neural Networks (IJCNN), 2017, pp. 1547-1554.

[18] K. J. Piczak, Environmental sound classification with convolutional neural networks, in 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), 2015, pp. 1-6.

[19] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, Sound event localization and detection of overlapping sources using convolutional recurrent neural networks, in IEEE Journal of Selected Topics in Signal Processing, Mar. 2019, pp. 34-48.

[20] S. Deshmukh, B. Raj, and R. Singh, Multi-Task learning for interpretable weakly labelled sound event detection, arXiv: Audio and Speech Processing, 2020.

[21] T. K. Chan, C. S. Chin, and Y. Li, Non-Negative matrix factorization-convolutional neural network (NMF-CNN) for sound event detection, arXiv: Audio and Speech Processing, 2020.

[22] K. Wakayama and S. Saito, CNN-Transformer with self-attention network for sound event detection, ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022, pp. 806-810.

[23] Y. Mao, Y. Zeng, H. Liu, W. Zhu, and Y. Zhou, ICASSP 2022 L3DAS22 Challenge: Ensemble of Resnet-Conformers with ambisonics data augmentation for sound event localization and detection, ICASSP 2022-2022 IEEE International Conference on Acoustics, 2022, pp. 9191-9195.

[24] S. Sabour, N. Frosst, and G. E. Hinton, Dynamic routing between capsules, in Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 3859-3869.

[25] Y. Liu, J. Tang, Y. Song, and L. Dai, A capsule based approach for polyphonic sound event detection, in 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2018, pp. 1853-1857.

[26] T. Iqbal, Y. Xu, Q. Kong, and W. Wang, Capsule routing for sound event detection, in 2018 26th European Signal Processing Conference (EUSIPCO), 2018, pp. 2255-2259.

[27] Y. Xu, Q. Kong, Q. Huang, W. Wang, and M. Plumbley, Attention and localization based on a deep convolutional recurrent model for weakly supervised audio tagging, in Interspeech 2017, 2017, pp. 3083-3087.

[28] DCASE 2017 Task4, 2017. [Online]. Available: http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-large-scale-sound-event-detection.

[29] A.Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, Attention is all you need, in Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 5998-6008.

[30] G. Liu, F. A. Reda, and K. Shih, Image inpainting for irregular holes using partial convolutions, in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 85-100.

[31] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, Language modeling with gated convolutional networks, in Proceedings of the 34th International Conference on Machine Learning (ICML), 2017, pp. 933-941.

[32] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, Large-scale weakly supervised audio classification using gated convolutional neural network, in 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 121-125.

[33] F. Font, G. Roma, and X. Serra, Sound sharing and retrieval, in Computational Analysis of Sound Scenes and Events, 2018, pp. 279-301.

[34] A. Mesaros, T. Heittola, and T. Virtanen, Metrics for polyphonic sound event detection, Applied Sciences, 2016, pp. 162.

[35] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, Large-scale weakly supervised audio classification using gated convolutional neural network, in 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 121-125.

[36] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, Sound event detection of weakly labelled data With CNN-Transformer and automatic threshold optimization, in IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2020, pp. 2450-2460.

[37] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, in 3rd Int. Conf. Learn. Repr. (ICLR), 2014.

[38] Laurens, V. D. Maaten, and Hinton. G, Visualizing data using t-sne, Journal of Machine Learning Research, 2008, pp. 2579-2605.

[39] Y. C. Wu, P. C. Chang, C. Y. Wang, and J. C. Wang, Asymmetrie kernel convolutional neural network for acoustic scenes classification, in 2017 IEEE International Symposium on Consumer Electronics (ISCE), 2017, pp. 11-12.