A novel attention model across heterogeneous features for stuttering event detection

Abedal-Kareem Al-Banna[a,c] (a.al-Banna@lboro.ac.uk), Hui Fang[a] (h.fang@lboro.ac.uk), Eran Edirisinghe[b] (e.edirisinghe@keele.ac.uk)

[a] Department of Computer Science, Loughborough University, Loughborough, LE11 3TU, UK

[b] School of Computing & Mathematics, Keele University, Keele,Newcastle ST5 5BG, UK

[c] Department of Artificial Intelligence and Data Science, University of Petra, 961343 Amman 11196 Jordan

Corresponding Author:

Abedal-Kareem Al-Banna

Department of Computer Science, Loughborough University, Loughborough, LE11 3TU, UK

Email: a.al-Banna@lboro.ac.uk

A novel attention model across heterogeneous features for stuttering event detection

Abedal-Kareem Al-Banna[a,c,*], Hui Fang[a], Eran Edirisinghe[b]

[a] *Department of Computer Science, Loughborough University, Loughborough, LE11 3TU, UK*
[b] *School of Computing & Mathematics, Keele University, Keele,Newcastle ST5 5BG, UK*
[c] *Department of Artificial Intelligence and Data Science & University of Petra, 11196, Jordan*

Abstract

Stuttering is a prevalent speech disorder affecting millions worldwide. To provide an automatic and objective stuttering assessment tool, Stuttering Event Detection (SED) is under extensive investigation for advanced speech research and applications. Despite significant progress achieved by various machine learning and deep learning models, SED directly from speech signal is still challenging due to stuttering speech's heterogeneous and overlapped nature. This paper presents a novel SED approach using multi-feature fusion and attention mechanisms. The model utilises multiple acoustic features extracted based on different pitch, time-domain, frequency domain, and automatic speech recognition feature to detect stuttering core behaviours more accurately and reliably. In addition, we exploit both spatial and temporal attention mechanisms as well as Bidirectional Long Short-Term Memory (BI-LSTM) modules to learn better representations to improve the SED performance. The experimental evaluation and analysis convincingly demonstrate that our proposed model surpasses the state-of-the-art models on two popular stuttering datasets, with 4% and 3% overall F1 scores, respectively. The superior resultsChee et al. (2009) indicate the consistency of our proposed method, supported by both multi-feature and

*Corresponding author.
Email addresses: a.al-Banna@lboro.ac.uk (Abedal-Kareem Al-Banna), h.fang@lboro.ac.uk (Hui Fang), e.edirisinghe@keele.ac.uk (Eran Edirisinghe)

attention mechanisms in different stuttering events datasets.

1. Introduction

Speech is commonly used in human communication to share thoughts, concepts, knowledge and ideas between individuals and groups (Alim & Rashid, 2018). Speech waves are represented by acoustic energy, which humans perceive and interpret (Ferrand, 2017). Producing this energy is a complex process involving integration between the nervous, respiration, phonation and articulation systems; therefore, any abnormality in this collaboration may cause stuttering (Ronald B. Gillam, 2022). Stuttering is a neuro-developmental speech disorder affecting 1% of the global population (Al-Banna et al., 2022a). Automatic detection of stuttering behaviours, known as stuttering event detection (SED), can facilitate applications like stuttering severity assessment. However, SED remains challenging due to the heterogeneous and overlapped nature of stuttering behaviors in speech. The World Health Organisation (WHO) describes stuttering as "speech that is characterised by frequent repetition or prolongation of sounds or syllables or words, or by frequent hesitations or pauses that disrupt the rhythmic flow of Speech. It should be classified as a disorder only if its severity is such as to markedly disturb the fluency of Speech."(World Health Organisation, 2010).

The main symptoms in the speech of People Who Stutter (PWS) include repetition, interruptions in the flow of speech (blockages), and prolonged speech. PWS often use physical concomitants, such as eye blinking, head movement, foot tapping, nose flaring, and facial grimaces, to conceal their dysfluency (Riley et al., 2004). As listed in Table 1, stuttering events are categorised into four groups, Speech Language Pathologists (SLPs) regularly monitor and observe the four groups of stuttering symptoms to determine the severity of the stuttering. In evaluation sessions, the SLP exerts efforts to manually observe and analyse stuttering events from different speech contexts such as monologues, dialogues or reading. Early childhood stuttering evaluation is vital, especially

during the preschool years between three and five. It helps diagnose stuttering and may help 20% of Children Who Stutter (CWS) to treat stuttering before being chronic (Villegas et al., 2019). Thus, developing an automatic stuttering detection technique is useful to help the SLP in stuttering evaluation sessions.

Stuttering event	**Example**	**Type**
Repetition	**Th-th-this** Stuttering is not simple	Primary
Interjection	Your voice **um ah** should be heard	Secondary
Prolongation	Your **vvvvvvoice** should be heard	Primary
Block	**k(involuntary stop)** Kareem	Primary

Table 1: The main categories of speech symptoms in stuttering and the sub-categories of repetition, such as word and sound, are listed in the table.

Automatic SED is a challenging task due to its diverse determining factors that can impact its performance, such as the nature of stuttering speech and the lack of reliable training data (Barrett et al., 2022; Lea et al., 2021; Kourkounakis et al., 2021). The existing models of SED can be broadly categorised into two classes, i.e., signal processing approaches and Automatic Speech Recognition (ASR) approaches. Different methods have been utilised in the latter, such as Hidden Markov Model (HMM) Tan et al. (2007), Task-oriented lattices (Alharbi et al., 2018), and pre-trained ASR models (Mohapatra et al., 2022). While in the signal approaches, previous research employed traditional Machine Learning (ML) algorithms such as Support Vector Machine (SVM) and k-Nearest Neighbours (k-NN) to detect stuttering events from the speech signal directly. Recently, Deep Learning (DL) approaches such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and ConvLSTM methods are commonly used as new signal processing approaches. The literature shows that the DL methods outperformed the traditional ML approach (Al-Banna et al., 2022b) in SED. However, conventional methods that rely on acoustic parametric and nonparametric feature extraction and acoustic processing techniques (Mahesha & Vinod, 2016) efficiently detect certain stuttering events, such as

block and prolongation (Afroz & Koolagudi, 2019). On the other hand, the ASR approach is considered to be better than the signal approach for detecting repetition events (Lea et al., 2021).

It is well known that speakers have different speech rates and vocal tracts, which may affect speech production. e.g., females typically have a higher pitch than males because females have a shorter vocal tract. In another instance, the speech rate of the elderly normally is slower than children. Consequently, different characteristics of speakers, including age, gender, and accent, may affect the detection performance of the SED and its generalisation ability. Moreover, people who stutter tend to have fairly different kinds of stuttering events (Ronald B. Gillam, 2022). Stuttering events could be heterogeneous and overlapped. In repetition, for instance, the Person Who Stutter (PWS) may produce three to four repetitions at the beginning of the word or even within the words, e.g. (foot,b,b,b, ball) or (ka, ka, Kareem). Detecting these variations of stuttering events is difficult owing to the nature of stuttering, the reliability of training data and the effectiveness of specific parametric and nonparametric acoustic processing techniques.

To solve the above-mentioned issues, the primary goal of this paper is to propose a novel multi-feature-based deep learning model to improve the detection performance of the SED. Therefore, we exploit the attention mechanism and fuse multiple features, including a set of pitch, time domain, auditory-based spectral and ASR features, in order to extract more reliable feature representations to improve the detection performance of SED as well as detect stuttering events core-behaviours. The main contributions of the proposed work are summarised below:

- A novel attention-based model is proposed to effectively learn frame-level and temporal representations by considering contextual, pitch, time-domain and auditory-based spectral features.
- Our multi-feature fusion approach, using time-domain features (Zero Crossing Rate (ZCR), Spectral Flux Onset strength envelope (SFO) and

Fundamental Frequency (FF)), ASR embeddings, and auditory-based spectral features, is capable of improving SED performance significantly, which outperforms state-of-the-art methods.

- A convolutional block with attention maps along two separate dimensions based on Convolutional Block Attention Module (CBAM) (Sanghyun Woo, Jongchan Park, Joon-Young Lee, 2018) was introduced for SED. The proposed work demonstrates the effectiveness of this lightweight module in performing automatic feature selection by assigning shared weights to the intermediate feature map and eventually focusing on the salient features of speech regions, leading to improved performance in SED.

The rest of the paper is structured as follows: Section 2 presents the motivation for the study, Section 3 reviews previous studies on stuttering events detection and classification, Section 4 presents the methodology and the proposed architecture for the stuttering events detection model, Section 5 presents the experiments, results, and discussions, the conclusion and future works placed in Section 6.

2. Related work

There are existing studies on stuttering events detection and how parametric, nonparametric, hybrid features and acoustic processing techniques are leveraged with machine learning and deep learning (Barrett et al., 2022). The summary of stuttering detection and classification models that we have conducted is presented in Table 2.

2.1. Motivation and background

In recent years, more efforts have been devoted to advancing speech research and applications for people with motor problems. While previous studies employed DL and ML in speech interaction applications for blind people, the elderly, and limited hand dexterity applications (Li et al., 2019; Almutairi et al., 2022; Lea et al., 2022). Only a few previous research attempts have investigated

using ML and DL algorithms in SED. Leveraging SED in different applications may impact the quality of life for 70 million PWS worldwide, in different aspects. Firstly, integrating a robust SED with the current speech assistive technology may improve their understanding of stuttering speech. Secondly, SED plays a significant role in streamlining and easing the stuttering severity evaluation for CWS, which may impact the therapy plan for CWS.

Despite the gradually growing use of voice assistants over the years, these technologies did not generalise yet to understanding stuttering speech (Mitra et al., 2021). Current assistive technology, such as Google, Alexa, and SIRI, were initially trained on fluent speech data. Therefore, primary stuttering behaviours, such as audible block, repetitions and prolongation, will be pruned with current assistants, which could limit their performance. Thus, integrating existing ASR with SED may enhance their generalisation ability to understand stuttering events, and PWS will benefit from these technologies. Stuttering evaluation is another crucial application that needs an efficient SED.

Stuttering evaluation by SLP in early childhood (at preschool age), especially between three and five years old is required. It probably helps diagnose stuttering and may allow 20% of CWS to treat it before being chronic (Villegas et al., 2019). However, stuttering evaluation is not affordable for many, and it is tedious and time-consuming for SLPs and the PWS. In general, SLPs use two methods to evaluate stuttering severity, perceptual scaling and counting procedure. In the counting procedure, the SLP tracks and manually counts the percentage of frequency of Stuttered Syllables (%SS) or the Stuttered Words (%SW) and the duration of the stuttering event. Accordingly, automatic SED may help SLPs in stuttering evaluation sessions and streamline and ease the severity evaluation process.

2.2. Early ML approaches for SED

Various machine learning techniques have been suggested in scholarly works for SED. (Tan et al., 2007) created a set of tools called the "Malay Speech Therapy Assistance Tools" to aid speech therapists in diagnosing and training chil-

dren who stutter, a pre-emphasised Mel-Frequency Cepstral Coefficient (MFCC) feature vector was employed as input to the model. Hariharan et al. (2012) suggested repetitions and prolongations classification model based on k-NN and Linear Discriminant Analysis (LDA). In addition, MFCC and Linear Predictive Cepstral Coefficient (LPCC) feature extraction are developed to check model performance in recognising stuttering events. Mahesha & Vinod (2016) demonstrated a SED model based on the MFCC as an acoustic feature and Gaussian Mixture Model (GMM) where the GMM parameters were estimated. In Sheikh et al. (2021), A lightweight model architecture was proposed for stuttering detection, which utilised a Time-delay Neural Network (TDNN) with MFCC as the sole acoustic feature. The author of Al-Banna et al. (2022b) in the comparative study argued that the Random Forest (RF) and MFCC as a hand-crafted feature outperform other ML approaches.

2.3. Deep learning approaches for SED

Kourkounakis et al. (2020) proposed a hybrid deep neural network model that utilises a Bidirectional Long Short-Term Memory (BI-LSTM) and a residual network. The model trained on different types of repetitions and interjection, STFT utilised as a sole acoustic feature. Kourkounakis et al. (2021) suggested an end-to-end network for SED in the proposed model, the input signal was clipped into a fixed-size 4-second audio clip with a 16 kHz sample rate and then converted to an STFT with 256 frequency bins. These spectrograms passed through Squeeze-and-Excitation Residual Network to learn frame-level spectral representations. A global attention mechanism was added to the final recurrent layer to focus on the more salient parts of the speech. The author in (Al-Banna et al., 2022a) developed a SED model by employing an atrous convolutional network with dilation rates of 2, 4, 8, and 16.

The author of Mohapatra et al. (2022) proposed a deep learning approach based on a convolutional neural network and Contextual and Data Distillation (CDD) using Wav2Vec2.0 hidden states. The proposed method evaluated different data sizes, using a few samples of SEP-28k and FluencyBank datasets;

the authors argued that the model was generalised across different speakers. The author of Liu et al. (2023) proposed a single task model using CNN and transformer models. The proposed model utilised Wav2Vec2.0 embeddings as unique feature representations, the model improved by considering the variable length of the stuttering speech.

2.4. Multi-feature fusion

Lea et al. (2021) proposed a multi-task learning Conv-LSTM stuttering detection model based on LSTM and a convolutional neural network. The baseline model consists of a single RNN layer with LSTM. While the enhanced model created based on a single Convolutional layer and batch normalisation layer produces weights for the model features followed by an LSTM layer and classification layer with two branches. Different time domain, articulatory and frequency domain features are used in this model. Mel-filterbank energy with 40 dimensions was used with a cut-off frequency of zero Hz and 8000 Hz, and 100 Hz as a sample rate. In addition to the filter bank, three F0 dimensions, vector size with 41 phoneme probabilities extracted from a pre-trained acoustic model and articulatory vocal-tract features were utilised. The paper shows that employing multi-feature may improve SED generalisation and robustness.

The author of Jouaiti & Dautenhahn (2022) suggested a neural network that combines MFCC and phoneme classes and probabilities to detect four stuttering events. The model also contains another branch for dysfluency classification. After the $3 - second$ speech signal was downsampled to 8 kHz, A 20×47 vector of MFCC coefficient, phoneme class probability with (18×299 dims) and phoneme estimation 1×299 dims was extracted. The extracted features are then fed into three parallel models. Each model contains a bi-directional LSTM layer followed by a Rectified linear Activation Function (ReLU), time distribution and dense layers. The output of each model is merged and passed via two dense layers, followed by batch normalisation and dropout layers. At the end of the network, two separate classifiers are employed to detect and classify stuttering events. The author utilised the undersampling technique to resolve

the imbalanced nature of stuttering speech data by randomly deleting speech samples.

Reference	R	I	P	B	M-F	Attention	Feature	Method	Dataset
Tan et al. (2007)	✓	×	✓	✓	×	×	MFCC	HMM	Artificially generated
Ravikumar et al. (2009)	✓	×	×	×	×	×	MFCC	SVM	Manually collected
Hariharan et al. (2012)	✓	×	✓	×	✓	×	LPC LPCC WLPCC	k-NN LDA	39 Samples from UCLASS
Mahesha & Vinod (2016)	✓	✓	✓	×	×	×	MFCC	GMM	50 selected sample from UCLASS
Dash et al. (2018)	✓	×	✓	×	×	×	STT Amplitude	STT	Manually collected
Kourkounakis et al. (2020)	✓	✓	✓	×	×	×	STFT	RESNET BILSTM	25 speech samples UCLASS
Kourkounakis et al. (2021)	✓	✓	✓	×	×	✓	STFT	SE-NET BILSTM Global Attention	Artificially generated
Lea et al. (2021)	✓	✓	✓	✓	✓	×	Filter banks Pitch articulatory	ConvLSTM	SEP-28k FluencyBank
Sheikh et al. (2021)	✓	✓	✓	✓	×	×	MFCC	TDNN	138 speaker UCLASS
Al-Banna et al. (2022b)	✓	✓	✓	✓	×	×	MFCC	RF	SEP-28k FluencyBank
Jouaiti & Dautenhahn (2022)	✓	✓	✓	✓	✓	×	MFCC Phoneme classes Phoneme probabilities	BI-LSTM	SEP-28k FluencyBank
Sheikh et al. (2022)	✓	✓	✓	✓	×	×	Wav2Vec2.0 embeddings	ECAPA-TDNN	SEP-28k
Al-Banna et al. (2022a)	✓	✓	✓	✓	×	×	Log- mel-spectrogram	Atrous CNN	UCLASS FluencyBank
Bayerl et al. (2022)	✓	✓	✓	✓	×	×	Wav2Vec2.0 embeddings	SVM	FluencyBank//KSoF
Mohapatra et al. (2022)	✓	✓	✓	✓	×	×	Wav2Vec2.0 embeddings	CNN	SEP-28k FluencyBank
Liu et al. (2023)	✓	✓	✓	✓	×	✓	Wav2Vec2.0 embeddings	CNN Transformer	SEP-28k PSC-PS-DF
Proposed model	✓	✓	✓	✓	✓	✓	MFCC ZCR FF SFO Contextual features	CNN BI-LSTM CBAM	SEP-28k FluencyBank

Table 2: Previous Works in Stuttering Classification and Detection: Comparing the Detection of Stuttering Core Behaviours – Prolongation (**P**), Block (**B**), Repetition (**R**), Interjection (**I**), Utilising Multi-Feature (M-F) and Attention Across Different Datasets.

The related works show that different time-domain, frequency domain and ASR features are essential for accurate stuttering events detection. However, most studies used only one view of speech signals, such as auditory-based spectral, ASR, or time-domain features. The authors of Mahesha & Vinod (2016); Sheikh et al. (2021); Al-Banna et al. (2022b) used MFCC as a sole acoustic feature. While the authors of Ravikumar et al. (2009); Hariharan et al. (2012); Pálfy (2014) utilised time-domain features such as autocorrelation (ACF), envelope parameters, duration energy peaks and fundamental frequency. These features are important to detect specific stuttering events such as prolongation. The authors of Sheikh et al. (2022); Liu et al. (2023); Mohapatra et al. (2022) introduced, finetuned and used Wav2Vec2.0 contextual embeddings in SED. These embeddings are important to detect repetition events and boost SED performance. Limited research exists in the literature that fused multi-feature in SED. The author of Lea et al. (2021) used frequency domain and time domain features and pitch articulatory features in the detection task. The author of Jouaiti & Dautenhahn (2022) combined MFCC and phoneme classes and probabilities to detect four stuttering events. Thus in this work, a set of time domain, pitch, auditory-based spectral and ASR features are considered to propose an Attention-based multi-feature deep learning model that best learns the optimal features representation to increase the detection performance of SED.

3. Methodology

This section introduces a novel multi-feature attention model for stuttering event detection across heterogeneous features, as shown in Fig 1. The model mainly consists of three blocks. The model feature block, a convolutional block CNN and RNN block. The model features block extracts the pitch, temporal, contextual and auditory-based spectral features. The CNN block uses attention maps along two separate dimensions to improve the intermediate feature map with minimal computation and concentrate on the essential features of the stuttering speech. While RNN block aims to learn the temporal changes

of the speech signal.

3.1. Architecture overview

A high-level overview architecture of the proposed model for stuttering events detection is shown in Fig. 1. It employs a multi-feature deep learning model to effectively learn a frame-level and temporal feature representation of contextual, time-domain and auditory-based spectral features. These features are passed to the five parallel ConvLSTM branches. Combining CNN and BI-LSTM into a single architecture is suitable for SED in which temporal sequence modelling is advantageous (Mesaros et al., 2021; Kourkounakis et al., 2021; Sleem & Elhenawy, 2022). In the ConvLSTM, the CNN block is employed as a feature extractor to learn discriminative features through consecutive convolutions and non-linear transformations. At the same time, the RNN block aims to learn the temporal changes of the speech signal. After each convolutional block, attention maps along two different dimensions based on CBAM (Sanghyun Woo, Jongchan Park, Joon-Young Lee, 2018) has been used to improve the intermediate feature map with minimal computation and concentrate on the essential features for detecting stuttering.

The CBAM is a lightweight that outperforms the Squeeze and Excitation method on image classification and objects detection tasks (Sanghyun Woo, Jongchan Park, Joon-Young Lee, 2018). The module consists of two attention blocks as shown in Fig. 2, spatial and channel attention. The channel attention emphasises essential features in a feature map by weighing the importance of each channel. In addition, spatial attention highlights important regions and suppresses irrelevant regions by using a convolutional layer and skipping connection (Sanghyun Woo, Jongchan Park, Joon-Young Lee, 2018). The outputs of the parallel networks are combined and fed through a dense layer with separate sigmoid activation functions for each output, producing probability for each dysfluency class. A detailed description of the model is discussed below.

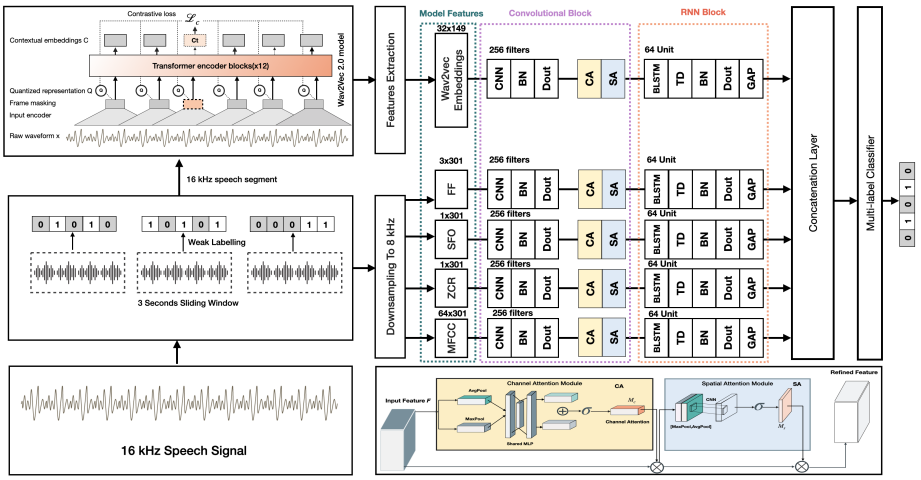

Figure 1: High-level overview of the proposed model architecture for Stuttering Event Detection SED: (a) ZCR - Zero Crossing Rate, (b) FF - Fundamental Frequency, (c) SFO - Spectral Flux Onset, (d) MFCC - Mel Frequency Cepstral Coefficient, (e) CNN - Convolution Neural Network, (f) BN - Batch Normalization, (g) CA - Channel Attention, (h) SP - Spatial Attention, (i) BLSTM - Bi-directional LSTM, (j) TD - Time Distribution layer, (k) GAP - Global Average Pooling, (l) Dout - Dropout

3.2. Attention mechanisms

Convolutional Neural Network (CNN) is utilised to extract discriminative features through a sequence of convolutions and non-linear transformations to learn a frame-level representation of contextual, pitch, temporal and auditory-based spectral features. This work employs five parallel convolutional blocks, both 1D and 2D. Each block consists of a single CNN layer with 256 filters and a size of 3 for 1D CNN and 3×3 for 2D CNN, followed by batch normalisation and a dropout of 0.4. CNN block detects spatial patterns via sliding and overlapping techniques on the extracted features. As shown in Eq. 1, The kernel operation K is applied for each window of the input featurfes I to generate a feature map $f(t)$ for each window (i, j) . This process is repeated for all possible windows, resulting in multiple feature maps that capture different characteristics of the input features. The feature maps pass through ReLU activation function that maintains positive values and changes the negative values to zero.

The generated feature maps of each convolutional block are then channelled through attention maps along two separate dimensions based on CBAM (Sanghyun Woo, Jongchan Park, Joon-Young Lee, 2018) to refine the intermediate feature map with minimal computation and concentrate on the essential features for detecting stuttering. The CBAM consists of two attention modules, as illustrated in Fig 2, channel and spatial attention. Channel attention emphasizes essential features in a feature map by weighing the importance of each channel. In addition, spatial attention highlights important regions and suppresses irrelevant regions by using a convolutional layer and skipping connection (Sanghyun Woo, Jongchan Park, Joon-Young Lee, 2018). The outputs of the parallel networks are combined and fed through a dense layer with separate sigmoid activation functions for each output, producing probability for each dysfluency class.

$$f(t) = (I * K)(i, j) = \sum_{m} \sum_{n} I(m, n) K(i - m, j - n) \tag{1}$$

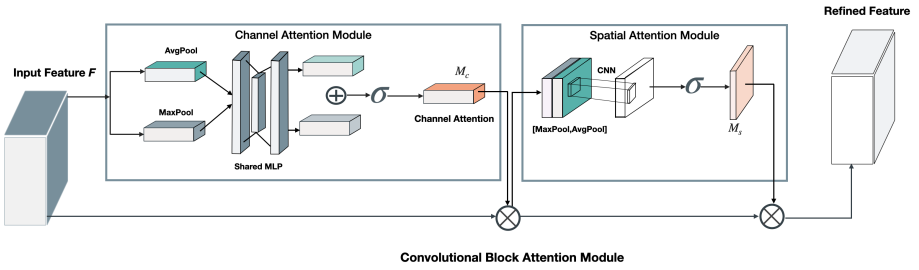

Figure 2: Overview of the attention maps along two separate dimensions based on CBAM .

The channel attention module processes the channel of the feature map (Sanghyun Woo, Jongchan Park, Joon-Young Lee, 2018). It starts by applying the maximum and average pooling to the input feature map F. The output of these pools is then fed into a multi-layer perceptron network. The sum of the two outputs is passed through a sigmoid activation function to calculate the attention values, as shown in Eq. 2. These values are then multiplied with the input features to generate the final channel attention.

$$
\begin{aligned}
M_c(F) &= \sigma(MLP(max\_pool(F)) + MLP(avg\_pool(F))) = \\
&\sigma(W_1(W_0(F^c_{max})) + W_1(W_0(F^c_{avg})))
\end{aligned}
\tag{2}
$$

In the equation, σ represents the sigmoid function, where the MLP is a multi-layer perceptron with one hidden layer, the W_0, and W_1 are shared weights within the MLP network. The average and maximum pooling output are represented by F^c_{avg} and F^c_{max}, respectively.

The spatial attention module extracts attention in the spatial dimension by first averaging and maximum pooling the input feature map in the spatial dimension. This produces two feature maps, which are then concatenated along the channel dimension. These feature maps are then passed through a convolutional layer and a sigmoid activation function. The resulting feature values are multiplied with the original input feature to generate the final refined feature maps, as shown in Eq. 3.

$$
\begin{aligned}
M_s(F) &= \sigma(f^{7*7}([max\_pool(F); avg\_pool(F)])) = \\
&\sigma(f^{7*7}([F^s_{max}; F^s_{avg}]))
\end{aligned}
\tag{3}
$$

In the equation, the symbol σ represents the sigmoid function, and the term f^{7*7} indicates that a convolution operation is being performed with a filter of size $7 * 7$ and the output denoted by F^s_{avg} corresponds to the average pooling, while F^s_{max} represents the output obtained from maximum pooling.

3.3. Learning temporal features using recurrent block

In order to detect stuttering events, it is vital to analyse the temporal features of speech segments. A recurrent neural network (RNN) is well suited for analysing sequential data such as speech, and can accurately identify stuttered speech event (Lea et al., 2021; Jouaiti & Dautenhahn, 2022). In this work, a recurrent block is used to learn the temporal features of refined feature maps. The RNN block, as illustrated in Fig 1, begins with permuting and reshaping

the refined feature maps produced by the CBAM to be processed by a recurrent neural network (RNN). A bidirectional LSTM layer with 64 units is then applied to the reshaped feature representation.

The bidirectional LSTM processes the input sequence in both forward and backward directions. It allows the model to consider past and future information, resulting in a more comprehensive understanding of the speech segment. After the LSTM layer, a time-distributed dense layer is employed, allowing the model to learn different feature representations at different time steps. This layer applies a fully connected dense layer to each time step of the input sequence. In addition, to improve the stability and speed of the model's training, batch normalisation is applied to the output of the time-distributed dense layer. A dropout layer with a rate of 0.4 is applied to the output of the batch normalisation layer. Dropout is a regularisation technique that can help to prevent overfitting. Finally, the output of the dropout layer is passed through a global average pooling layer, which collapses the entire 2D tensor into 1D. This is used to reduce the dimensionality of the output and make it more convenient for the final stages of the network.

3.4. Feature representation

A temporal acoustic region within the speech signal is annotated in a binary manner with one or more stuttering events. These annotations indicate if the stuttering events are active or inactive in that region. The acoustic regions may have more than one stuttering event. In addition, these annotations include extra metadata describing that region and contain start and end intervals in the original speech signal. In SED models, the duration of the temporal region is generally between three and five seconds. Using the fixed-sized region makes supervised learning methods suitable for SED task, where the acoustic regions and the annotations are used to train the model. Therefore, the monophonic three seconds acoustic regions were sampled at 8000 Hz and 16000 Hz, as shown in Fig 1 and then fed into two layers to extract four groups of features: auditory-based spectral, pitch, temporal feature and contextual representations.

3.4.1. Auditory-based spectral features

The acoustic energy over a set of frequencies can be detected using spectral representation (McFee et al., 2015). As described in Algorithm 1, this paper's experiments extracted the auditory-based spectral feature using MFCC. The MFCC is a widely used feature extraction technique for speech signal analysis, which emulates the human auditory system. It involves converting audio signals into coefficients that capture spectral characteristics on a quasi-logarithmic frequency scale. The 8 Khz downsampled time-domain temporal region refined using a Signal-to-Noise ratio (SNR) estimator with $\alpha = 0.95\%$ to distinguish and quantify the speech signal from background noise, Table 3 shows the impact of alpha ratio on the performance of MFCC brach in term of F1 score.

The refined signal is then converted into a frequency domain using the Short-time Fourier Transform (STFT) with hop length with (0.010×sampling rate) and (0.025× number of FFT). Subsequently, 64 filter banks were applied to the frequency domain signal to extract MFCC coefficients vector representations for each acoustic region. Despite the importance of the aforementioned spectral feature representation in detecting frequency information, pitch and time information is equally essential for SED.

Algorithm 1 Compute MFCC coefficients

function GETMFCC(signal)

Input: 3 seconds downsampled time domain signal

Output: Computed MFCC coefficients $mfcc$

$S(t) \leftarrow \text{signal}[0 : \text{int}(3 \times 8000\,\text{Hz})]$

//Apply signal to noise estimator

$\alpha \leftarrow 0.95$

$\gamma(t) \leftarrow \text{S}(t) - \alpha\,\text{S}(t-1)$

Framing signal $\gamma(t)$ into **N** frames of size 25 ms with a stride of 10 ms

//Compute the Hamming window $W(n)$

$W(n) \leftarrow 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right)$

//Compute power spectrum for each frame x:

$P_s \leftarrow \frac{|\text{FFT}(x_i)|^2}{N}$

//Convert the power spectrum to mel-scale:

$m \leftarrow 2595\log_{10}\left(1 + \frac{f}{700}\right)$ by applying 64 triangular filters

//Apply Discrete Cosine Transform on filter-banks:

$mfcc \leftarrow \text{dct(m)}$

return $mfcc$

end function

Alpha	Prolongation	**Block**	**Sound**	**Word**	**Interjection**	**Avg.F1**
0.90	63.99	55.89	69.56	62.81	76.63	65.78
0.91	62.76	58.10	70.12	61.88	76.37	65.85
0.92	66.65	61.11	67.06	63.34	74.36	66.50
0.93	68.10	56.59	66.12	64.10	77.02	66.38
0.94	66.75	58.88	70.59	65.07	73.95	67.05
0.95	68.41	57.46	67.68	66.46	76.19	**67.24**
0.96	66.84	57.91	66.75	63.57	76.11	66.24
0.97	66.66	59.64	70.50	62.69	73.84	66.66

Table 3: The impact of alpha ratio on the performance of MFCC brach in terms of F1 score on a subset of SEP-28k dataset.

3.4.2. Pitch and Temporal Features

Pitch and temporal features are commonly used to detect unvoiced periods and sudden changes in stuttering speech. Quantifying the unvoiced periods in stuttering speech may enhance the detection of blocks and prolongation events since the voiced speech is smoother than unvoiced (Bäckström et al., 2022). Three pitch and temporal features were extracted from the temporal region ZCR and the spectral flux strength envelope autocorrelation. As described in algorithm 2, the ZCR of the temporal region frames is the ratio of changing the signal sign between negative to positive values during consecutive samples x_i within a given time window ω and vice versa (Bäckström et al., 2014). The ZCR in stationary signal is defined in Eq. 4 and Eq. 5:

$$Z(i) = \sum_{n=\infty}^{-\infty} \frac{1}{2} \mid sgn[x_i(n)] - sgn[x_i(n-1)] \mid \omega[(n-i)], \tag{4}$$

$$sgn[x_i(n)] = \begin{cases} 1, & x_i(n) \geq 0, \\ -1, & x_i(n) < 0. \end{cases} \tag{5}$$

Algorithm 2 Compute Zero Crossing Rate (ZCR) for 3-second Signal

function GETZCR(signal)

Input: 3 seconds downsampled time domain signal

Output: Compute Zero Crossing Rate Z

Initialize parameters:

Sample rate: $S_r \leftarrow 8000\,\text{Hz}$

Window size: $N \leftarrow \text{int}(S_r \times 0.025)$

Overlap size: $\text{overlap} \leftarrow \text{int}(N \times 0.1)$

Total frames: $\text{num\_frames} \leftarrow \text{int}\left(\frac{\text{length of signal}}{N - \text{overlap}}\right)$

Initialize $Z \leftarrow [\,]$

for $i = 0$ to $\text{num\_frames} - 1$ **do**

$\text{start} \leftarrow i \times (N - \text{overlap})$

$\text{end} \leftarrow \min(start + N, \text{length of signal})$

Frame $x_i \leftarrow \text{signal[start:end]}$

Calculate $Z(i)$ using Eq. (4 and 5)

Append Z(i) to Z

end for

return Z

end function

In addition to the ZCR, The autocorrelation of a spectral flux onset (SFO) strength envelope is computed based on (Böck et al., 2012). The SFO is calculated by comparing the frame's power spectrum $|X(n,k)|$ of two consecutive frames and sums all positive deviations with $H(x) = \frac{x+|x|}{2}$ to return a vector containing the onset strength envelope representing the increasing spectral energy at each frame as shown in Eq. 6 where n is the frame number and k is the frequency bin index.

$$SF(n) = \sum_{k=1}^{k=\frac{N}{2}} H(|X(n,k)| - |X(n-1,k)|) \tag{6}$$

The pYIN (Mauch & Dixon, 2014) Algorithm 3 is employed to compute the

fundamental frequency (F0) of the temporal region. The output of this algorithm is three frame vectors representing the F0 candidate, Viterbi decoding and voicing flags where the Viterbi decoding estimates the most likely F0 sequence and voicing flags.

Algorithm 3 Compute Fundamental Frequency (F0) using pYIN Algorithm

function GETF0(signal)

Input: 3 seconds downsampled time domain signal

Output: Fundamental Frequency (F0) ,Voicing Flags,Voicing Probability

Initialize parameters:

Sample rate: $S_r \leftarrow 8000\,\text{Hz}$

Frame length: $N \leftarrow \text{int}(S_r \times 0.025)$

Overlap size: $\text{overlap} \leftarrow \text{int}(N \times 0.1)$

//Extract F0 using pYIN algorithm

f0, voiced_flag, voiced_probs = LIBROSA.PYIN(signal, S_r, overlap, N)

Replace NaN values in $f0$ with zeros

return $f0, \text{voiced\_flag}, \text{voiced\_probs}$

end function

3.4.3. Contextual features

The logits of size 32×149 produced by the WAV2Vec2.0 model (Baevski et al., 2020) were fused with proposed features to explore the effect of transformer embedding on the experimental results. A pre-trained base WAV2Vec2.0 model (Baevski et al., 2020) illustrated in Fig 3 trained on 960 of fluent speech was used as a feature extractor for the contextual features. Wav2Vec2.0 is a family of models for learning the representation of audio data. Wav2Vec2.0 model employs mask language modelling MLM of PERT (Devlin et al., 2019) on audio data, which involves training a model to predict masked (or "hidden") segments. In mask language modelling, the model is given an input audio signal, and a portion of the signal is masked (i.e., hidden from the model). The model is then trained to predict a token of audio data that gives the context of

the unmasked portion of the signal. Wav2Vec2.0 consists of two self-supervised learning tasks, a contrastive and a quantization task. In the contrastive task, the model is trained to maximise the similarity between related examples (positive samples) and minimise the similarity between unrelated examples (negative samples) in different masked time steps. The contrastive task is typically implemented using a contrastive $\mathcal{L}_c$ loss function, which compares the similarity between pairs of examples and adjusts the model parameters to minimise the loss. The contrastive task trains the model to perform various downstream tasks, such as stuttering events detection.

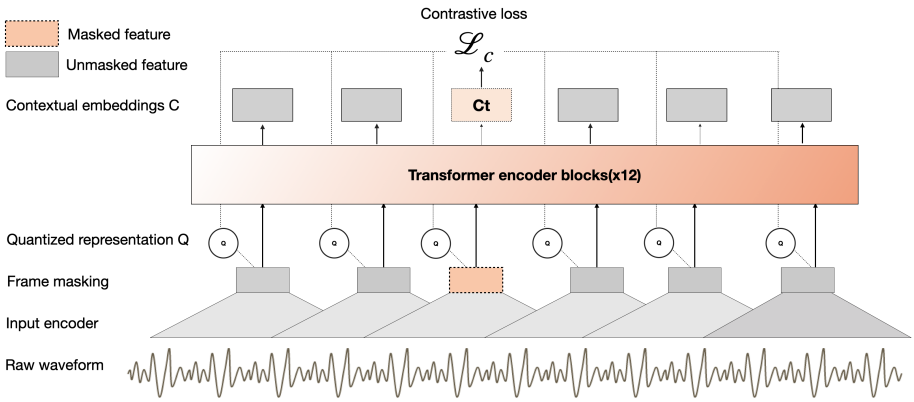

Figure 3: A pre-trained base WAV2Vec2.0 model trained on 960 of fluent speech used as a feature extractor for the transformer embeddings features.

4. Experiments, results, and discussions

This section presents the experimental results of the proposed model for stuttering event detection. The model was trained and evaluated on two datasets on stuttering core behaviours, SEP-28K (Lea et al., 2021) and FluencyBank (Bernstein Ratner & MacWhinney, 2018). In addition, the experimental setup, evaluation metrics, and results of the experiments are described. Furthermore, a comparison of the proposed model performance with existing models, and the implications and significance of the results in the context of stuttering detection are discussed.

4.1. Description of stuttering datasets

In order to rigorously evaluate the proposed model for stuttering events detection, two datasets are used in this work, SEP-28K and FluencyBank. The FluencyBank is a dataset provided for the studies of fluency development, and it is part of a more extensive open-access repository called TalkBank. This dataset includes monolingual and bilingual speech for children and adults who stutter (C/AWS), clutter (C/AWC), and second language learners. Speech sample data for this dataset are collected from stuttering assessment sessions of interview and reading tasks. In the interview task, a common set of interview questions were asked by SLP to the PWS, while in the reading task, the PWS read "Friuli," a reading passage from the Stuttering Severity Instrument-4 (SSI-4). The FluencyBank dataset comes with two separate sub-datasets, Voices-CWS and Voices-AWS. This paper used the speech utterances annotated by (Lea et al., 2021).

The SEP-28K dataset includes $28,137$ speech segments curated from 385 episodes of eight stuttering YouTube podcasts. For each episode, three-second intervals ranging from 40 to 250 were identified near the speech pause segment and subsequently extracted and annotated. In (Lea et al., 2021) also, Fleiss Kappa's inter-annotator agreement was utilized to validate and annotate $4,144$ (3.5) hours of the audio clip taken from the FluencyBank dataset. Table 4 presents the stuttering events and their respective total number of observations and data size in hours in both datasets.

Dataset	Prolongation	Block	Sound Rep	Word Rep	Interjection	Total of Obs	Total Hours
SEP-28K	3480	1679	2517	2295	4322	14293	11.91
FluencyBank	526	292	421	361	754	2354	1.96

Table 4: Description of the main categories and sub-categories of stuttering events and their respective total number of observations and data size in hours in FluencyBank and SEP-28k datasets.

The model's reliability and performance are correlated to the annotation process's quality. Therefore, the annotation process should be measured and

achieved by SLP (Al-Banna et al., 2022a) because people who stutter tend to have fairly different kinds of stuttering events. The stuttering events could be heterogenic and overlapped (Ronald B. Gillam, 2022) . However, In the annotation process of SEP-28k and FluencyBank, which at least three well-trained annotators achieved, the inter-annotator agreement for stuttering events needed to be more consistent across categories. For example, word repetitions, interjections, and sound repetitions showed higher agreement levels (Fleiss Kappa scores of 0.62, 0.57, and 0.40, respectively), while blocks and prolongations had only fair or slight agreement (Fleiss Kappa scores of 0.25 and 0.11 respectively) (Lea et al., 2021). Therefore, in this paper, the disagreement in the annotation is processed by taking the agreed stuttering core behaviours by three annotators. as described in Table 5. Previous research handled this issue using concordance correlation coefficient (CCC) loss using the inter-annotator agreement for each utterance (Lea et al., 2021) while Mohapatra et al. (2022) used the same approach followed.

Dataset	Prolongation	Block	Sound Rep	Word Rep	Interjection	Total of Obs	Total Hours
SEP-28K	1957	2096	1454	2165	1911	9583	7.985
FluencyBank	286	329	438	385	415	1853	1.544

Table 5: Description of the main categories and sub-categories of stuttering events and their respective total number of observations and data size in hours in FluencyBank and SEP-28k datasets after resolving the disagreement.

4.2. Model evaluation

The performance of the proposed model is evaluated in this paper using k-fold cross-validation and hold-out test. Moreover, multiple statistical metrics listed in Table 6 are considered to evaluate the proposed model. In the hold-out method, stuttering datasets were split into training and test sets to observe how the trained model performs on unseen stuttering events. Random splitting with a 10 % and 90 % test and training sets was followed to overcome cherry-picked debate. In 10-fold cross-validation, the training set is randomly divided into

10-fold. One fold is employed to validate the model performance, while the remaining is used to train the model. The paper results are the mean of the 10 folds on the test dataset. The following statistical metrics are considered in this paper.

Metric	Description
F1-Score	Weighted average of Recall and Precision
Recall	Sensitivity, ratio of TP to all stuttering events in the actual class
UAR (Unweighted Average Recall)	Combination of Sensitivity and Specificity
EER (Equal Error Rate)	Metric measuring misclassified predictions

Table 6: Statistical metrics for model evaluation.

- *F1-Score* is the weighted average of *Recall* and *Precision*. It is used to measure the performance of imbalanced datasets and computed as

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{7}$$

- *Recall* represents the model's sensitivity. It measures the ratio of correctly predicted positive events TP to all stuttering events in the actual class $TP + FN$. Recall is calculated as

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

- Unweighted Average Recall (UAR) is a combination of Sensitivity $\frac{TP}{TP+FN}$ and Specificity $\frac{TN}{FP+TN}$. In a binary classification task, Sensitivity is the ratio of the total predicted positive to the total number of positives, and it is called the Recall of the positive class. While Specificity is the total correctly predicted negative to the total number of negatives, it is called Recall on the negative class. The UAR can be computed as

$$UAR = \frac{specificity + sensitivity}{2.0} \tag{9}$$

- Equal Error Rate (EER) is a statistical metric that measures the misclassified predictions by the model. It is calculated as Eq 10 by dividing

the number of inaccurate (FP+FN) predictions by the total number of predictions(FP+FN+TN+TP).

$$EER = \frac{FP + FN}{FP + FN + TN + TP} \tag{10}$$

4.3. Implementation details

TensorFlow 2.6 framework (Abadi et al., 2016) is utilised to build and train the proposed model with different parameters listed in Table 7. TensorFlow is an open-source machine learning framework developed by Google that allows researchers to build and deploy machine learning models efficiently. In addition, Librosa Python library (McFee et al., 2015), which uses in audio and speech analysis, is utilised to extract the auditory-based spectral, pitch and temporal features from the speech signals explained in Section 3.4. Moreover, the sklearn Python library (Pedregosa et al., 2011) is used to implement k-fold cross-validation. Furthermore, a pre-trained Wav2vec2 base model is incorporated to generate contextual features for the speech signals. Through the use of these tools, the effective processing and detection of speech signals with high accuracy are achieved.

Parameter	**Value**
Optimiser	Adam (learning rate: 0.0001)
Convolutional Layers	1D and 2D with 256 filters, kernel size of 3 for 1D, and 3×3 for 2D
Activation Function	ReLU
Dropout	Dropout rate of 0.4 for all Convolutional Layers
Recurrent Layers	Bidirectional LSTM with 64 units
Dense Layers	TimeDistributed with 64 output units
Loss Function	Sigmoid Focal Cross Entropy
Early Stopping	Enabled with Patience of 5
Model Checkpoint	Save best models based on validation loss
Training Batch Size	128 for k-fold cross-validation

Table 7: Summary of model Parameters used in our proposed model.

The training datasets of the temporal regions and their corresponding labels are utilised for the model's training. k-fold cross-validation with 10 folds are employed, with data shuffled before each fold and balanced class weights being used. The sklearn library's k-fold class splits the data into training and validation sets through the training process. The Adam optimiser with a learning rate of 0.0001 and binary cross-entropy loss function is utilised in training. Additionally, the best model weights during training are saved, and early stopping patience set to 5 is implemented to prevent overfitting.

The time complexity and processing time are vital in implementing the proposed approach in real-life applications. The Psutil Python library on Google Colab Tesla T4 GPU with a total memory of 12.68 GB has been used to perform inference on 500 samples of speech data. We calculated the average processing time of each feature extraction method as explained in Table 8.

Feature	**Time complexity**	**Processing time** (ms)
MFCC	$\mathcal{O}(M * NlogN)$	19.56
ZCR	$\mathcal{O}(N)$	2.1
SFO	$\mathcal{O}(M * N + S)$	39.32
FF	$\mathcal{O}(M * N)$	3120.27
ASR embeddings	$\mathcal{O}(T * L * L)$	3682.27

Table 8: Summary of time complexity and processing time in milliseconds (ms) for each feature extraction method where N is the number of frames, M is the frame length,S represents the maximum lag length for the autocorrelation, T represents the number of transformers, and L is the speech segment length.

4.4. Experiments for ablation study

The multi-feature fusion approach in this paper involves combining multiple acoustic features extracted from speech signals using time-domain features (ZCR, SFO and FF), ASR embeddings, and auditory-based spectral features. In the proposed model, four groups of experiments were conducted to evaluate

the impact of various feature extraction techniques on the model's performance, as presented in Table 9. Mel-Frequency Cepstral Coefficients (MFCCs), logits obtained from the Wav2vec2 model, Zero Crossing Rate (ZCR), Spectral flux onset (SFO) and fundamental frequency were used as feature representations in the experiments. The effect of attention mechanisms was also a concern in all the experiments.

		Features	**Model structure**	**Attention**
Group1				
	Test1	MFCC	Conv Block+RNN Block	w/o attention
	Test2	MFCC	Conv Block+CBAM+RNN Block	w/. attention
Group2				
	Test1	MFCC + Wav2Vec2.0 logits	Conv Block+RNN Block	w/o attention
	Test2	MFCC + Wav2Vec2.0 logits	Conv Block+CBAM+RNN Block	w/. attention
Group3				
	Test1	MFCC + Wav2Vec2.0 logits+ZCR+FF+SFO	Conv Block+RNN Block	w/o attention
	Test2	MFCC + Wav2Vec2.0 logits+ZCR+FF+SFO	Conv Block+CBAM+RNN Block	w/. attention
Group4				
	Test1	MFCC + Wav2Vec2.0 logits+ZCR+FF+SFO	Conv Block+CBAM+RNN Block	w/. attention

Table 9: Summary of experimental groups, attention, and key features.

The first experiment uses a baseline model of one CNN block followed by an RNN block, with MFCC described in algorithm 1 as the sole acoustic feature. This experiment aimed to examine the behaviour of the baseline model using MFCC. Accordingly, the CNN block consists of a single 2D CNN layer with 256 filters of a 3×3 size, followed by batch normalisation and dropout of 0.4. The feature maps are then processed by attention maps based on CBAM. The RNN block includes a bidirectional LSTM layer with 64 units, a time-distributed dense layer and a global average pooling layer. The output is passed through a dense layer with sigmoid activation functions, producing probability for each stuttering event. The baseline model was tested without CBAM, as presented in group 1, to observe the impact of attention on model performance.

Fig 5 compares the F1 scores for the baseline model and the model w/o the CBAM attention mechanism. As observed, using the CBAM module slightly

increased the F1 score for prolongation, from 66.41% to 66.68%, and block, from 58.08% to 60.38%, which is a significant improvement. However, a decrease in the F1 score was observed for sound repetition, from 68.39% to 63.91%; on an investigation, it was found that MFCC had a high correlation with other classes, which led to overfitting. This outlier was handled by adding regularisation and reducing the feature set dimensionality. Furthermore, no significant difference was observed for interjection, with the F1 score remaining at 75.4% and 75.11%, respectively. These findings suggest that incorporating the CBAM module can lead to a significant improvement in the performance of the block, while it might have a negative effect on the performance of sound repetition. In addition, employing more features than a hand-crafted MFCC as an exclusive acoustic feature may enhance the performance of SED.

Experiment	Prolongation	Block	Sound	Word	Interjection	Avg.F1	Avg.ERR	Avg.UAR	Avg.recall
Group1+Test1	66.41	58.08	68.39	63.76	75.40	66.41	36.41	61.31	63.59
Group1+Test2	66.68	60.38	63.70	65.20	75.11	66.21	36.71	61.04	63.29
Group2+Test1	67.57	61.40	72.97	67.52	76.44	69.18	33.16	61.53	66.84
Group2+Test2	68.22	61.27	72.34	67.74	77.13	69.34	32.96	61.50	67.04
Group3+Test1	68.64	61.08	75.69	69.06	77.70	70.44	31.51	61.82	68.49
Group3+Test2	69.80	62.84	76.95	69.18	78.17	71.39	30.39	62.55	69.61
Group4+Test1	**72.23**	**68.66**	**79.89**	**72.07**	**79.40**	**74.45**	**23.43**	**64.29**	**76.57**

Table 10: Results of four experiment and test groups with different model structures and features on SEP-28k on the test dataset.

In the second experiment (group 2), the logits produced by the pre-trained base WAV2Vec2.0 model were fused with MFCC to explore the effect of contextual features on the experimental results. This group combines two identical branch structures described in Table 9. The model structure was tested without CBAM. The results demonstrate that the model's performance improved in all the classes except prolongation. The sound repetition recorded the most substantial improvement in the F1 score, rising by 4% from 68.39% to 72.97%. This suggests that the model could effectively learn and generalise to sound repetition. In addition, the block and the word repetition also recorded substantial enhancement, increasing by 3% from 58.08% to 61.4% and 63.76% to 67.52%,

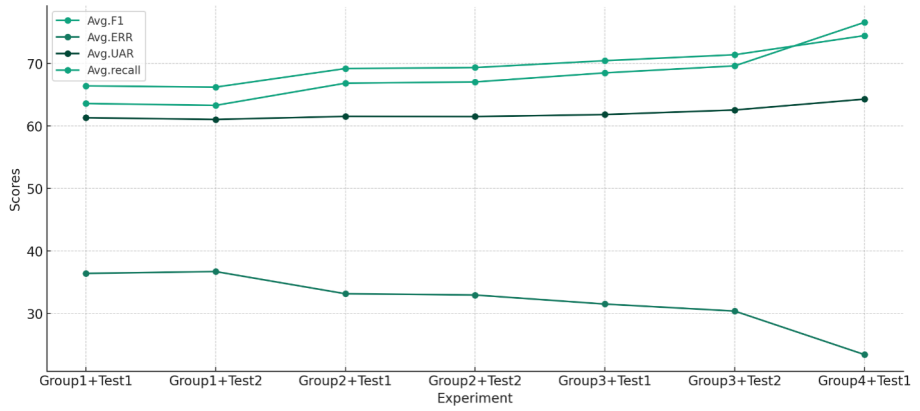

Figure 4: Results of four experiments and test groups.

respectively. Moreover, the interjection class recorded a slight but notable improvement from 75.40% to 76.44%. On the other hand, prolongation slightly increased by less than 1% from 66.41% to 67.57%. The experiment results show that including CBAM in the model had a small but insignificant impact on its performance. Specifically, the model's F1-score increased by 0.16%. However, the F1-score of the interjection and the prolongation recorded 1% enhancement.

In the third experiment (group 3), multi-feature were fused on the model to effectively learn a frame-level and temporal features representation of contextual, time-domain and auditory-based spectral features as described in Table 9. These features are passed to the five parallel branches with the same structure as the baseline model. The results demonstrate an enhancement in performance across all five classes of a multi-feature model with CBAM. The sound repetition class has the highest improvement of 4.61% among the stuttering events, indicating that the model struggled to detect sound repetition in the previous groups without pitch and temporal features. However, the F1 score of interjection slightly increased by 1%. Moreover, the F1 score of the block, word repetition and prolongation increased by 1.5%.

In the fourth experiment (group 4), the sigmoid activation function with a Focal Loss (FL) with $\alpha = 0.25$ and $\gamma = 2.0$ was employed to modify the

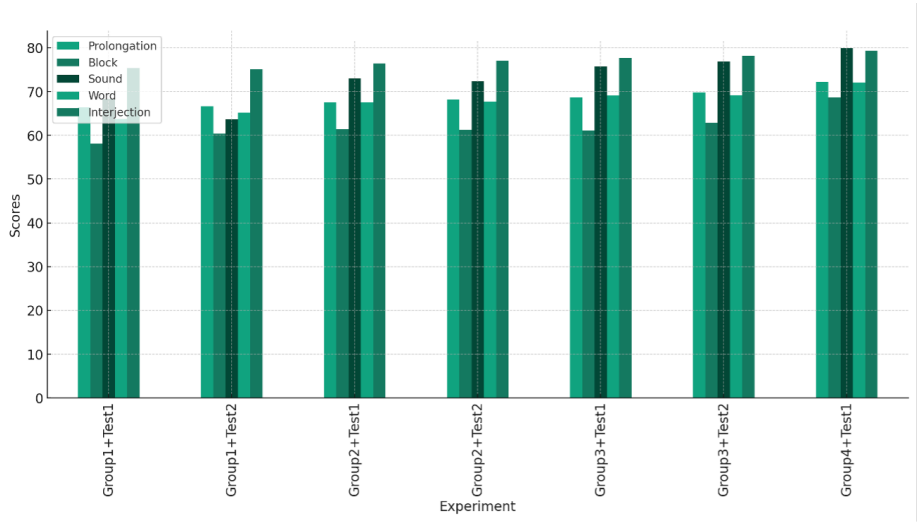

Figure 5: Results of four experiments and test groups for each stuttering event.

standard cross-entropy loss used in multi-label classifier by adding a focal weight to address the class imbalance problem. It can be clearly seen that along with FL hyper tuning, the overall F1 score of the testing results increased by 3%, from 71.39% to 74.45%. In addition to the improvement of F1 score, UAR increased by 2%, EER decreased by 7% and the recall increased approximately by 7% as illustrated in Fig 4, respectively.

4.5. Results comparisons and implications

As presented in Table 11, the proposed model achieves an average F1 score of 74.44% on the SEP-28k dataset, an absolute improvement of 4% over the 70.60% F1 score of (Mohapatra et al., 2022) with large effect size. On the FluencyBank dataset, the model obtains a 3% absolute gain in average F1 score compared to (Lea et al., 2021) (71.41% versus 68.2%), with a moderate effect size. This difference is statistically significant ($p<0.01$) with a Cohen's d effect size of 0.85. Statistical testing indicates significant gains over the prior art, with moderate to large effect sizes (Cohen's d of 0.4 to 0.7). This demonstrates substantial practical improvements in detection performance.

The model demonstrates improved block and word repetition performance on both datasets, outperforming the current state-of-the-art models. Specifically, the block event achieved 68.66% on the SEP-28k and 66.20% on FluencyBank. On the other hand, for word repetition, the F1 score was 72.02% and 69.58% on SEP-28k and FluencyBank, respectively. The outperformance of existing models highlights the potential of the proposed model in stuttering event detection and demonstrates a noticeable enhancement of block.

SEP-28k	**F1**					
	Prolongation	**Block**	**Sound**	**Word**	**Interjection**	**Avg.F1**
Baseline (MFCC)	66.41	58.08	68.39	63.76	75.40	66.41
Baseline (MFCC+ASR embeddings)	68.22	61.27	72.34	67.74	77.13	69.34
Baseline (MFCC+ASR embeddings+ZCR+SFO+FF)	69.80	62.84	76.95	69.18	78.17	71.39
Lea et al. (2021)	68.50	55.90	63.20	60.40	71.30	63.86
Liu et al. (2023)(Train 1/2 Test All)	67.00	58.00	66.00	53.00	79.00	64.60
Mohapatra et al. (2022)	**73.00**	58.00	72.00	71.00	79.00	70.60
Improved model	72.23	**68.66**	**79.89**	**72.02**	**79.40**	**74.44**

FluencyBank	**F1**					
	Prolongation	**Block**	**Sound**	**Word**	**Interjection**	**Avg.F1**
2D with local pooling Al-Banna et al. (2022a)	25.80	44.10	47.00	NA	52.70	42.4
CNN+LSTM Al-Banna et al. (2022a)	42.30	40.60	54.30	NA	53.30	47.63
BILSTM Al-Banna et al. (2022a)	41.00	44.20	45.00	NA	54.70	46.23
Al-Banna et al. (2022a)	44.50	45.20	49.00	NA	52.50	55.10
Bayerl et al. (2022)	60.00	33.00	60.0	43.00	**84.0**	56.00
Mohapatra et al. (2022)	73.00	63.00	61.00	67.00	60.00	64.80
Lea et al. (2021)	67.90	56.80	**74.30**	59.30	82.60	68.18
Improved model	**80.38**	**66.20**	66.41	**69.58**	74.46	**71.41**

Table 11: Comparison of the Proposed Model's F1 Score with State-of-the-Art Models on SEP-28k and FluencyBank Datasets. The F1 scores demonstrate the superior performance of our model across both datasets.

The improved performance of the model on block and word repetition in both datasets is attributed to three main factors. Firstly, the proposed model addresses the reliability issues of the datasets by using the intra-rater agreement with at least three annotators since the annotator of this dataset is not SLPS, and the agreement on this event is 25%, as described in the data section. The intra-rater agreement interprets the results of Al-Banna et al. (2022a) in block and other events, while Mohapatra et al. (2022) handles this issue with the same

approach followed in the proposed model.

The second factor is the fusion of temporal, contextual and pitch features. This finding is compatible with the previous studies (Lea et al., 2021; Al-Banna et al., 2022b) that argued the importance of these features to the detection block. Finally, the model effectively tackles the class imbalance problem by applying the best-fit focal weights after massive experiments. The model shows promising results and achieved state-of-the-art results only on 8 hours and 1.5 hours of reliable data of SEP-28k and FluencyBank, respectively, on all stuttering events.

Regarding the prolongation class, the model outperforms state-of-the-art by 7.38% on the FluencyBank dataset; however, Mohapatra et al. (2022) performs slightly better than the proposed model by 73.00% on SEP-28k. Concerning sound repetition, the model demonstrates promising results and surpasses the previous studies by 7.8% on SEP-28k; it also achieved 66.41% on FluencyBank. The baseline results of Al-Banna et al. (2022a) were analysed since the performance of the sound repetition is considerably less than the previous studies by 11%; this may be due to merging sound and word repetition into one core behaviour class and using F1 score on minority classes. Hence, It may be advisable to classify word and sound repetition as separate categories. The Lea et al. (2021) shows strong performance on interjection on the FluencyBank dataset with 82.60% and sound repetition with 74.30%; however, it struggled to detect other stuttering events. Therefore, phoneme probabilities and articulatory vocal tract features in Lea et al. (2021) are viable for these stuttering events. In addition, using three annotators' agreements to resolve inter-rater agreements can result in missing data on interjection and sound, making them misclassified in a relatively small dataset. For FluencyBank, the proposed model showed a drop in performance compared to SEP28-k. On closer inspection, the annotation quality for SEP28-k was found to be lower, especially for the block class. To handle this, the model was retrained after filtering low-confidence samples to improve results.

4.6. Limitations of the proposed approach

Employing a multi-feature fusion approach, using time-domain features (ZCR, SFO and FF), ASR embeddings, and auditory-based spectral features on reliable data with multi-label time annotation, is capable of improving SED performance significantly, which outperforms state-of-the-art methods As discussed in the previous section. Despite the promising results of the proposed model, this model imposes two limitations. The first limitation is data limitation. The model was trained on only two English published stuttering datasets, SEP-28k and FluencyBank (Lea et al., 2021), recently curated by Apple company. Unfortunately, the scarcity of training data in English and other languages, such as Arabic, is the main challenge in the stuttering detection domain (Barrett et al., 2022). To overcome this limitation, we rigorously tested the proposed model on unseen samples with exclusive speakers from SEP-28k and FluencyBank datasets to ensure its generalisation ability.

Although the number of parameters of the proposed model is approximately eight million, the second limitation is the implementation of the proposed model in real-life applications. Our next paper stuttering Auto-Annotate tool utilised our multi-feature fusion approach to ease and streamline the annotation process for the stuttering speech data.

5. Conclusion and future work

In conclusion, this work proposes a novel attention-based multi-feature fusion model for robust stuttering event detection. The approach combines diverse acoustic features and employs channel and spatial attention mechanisms to learn effective representations across heterogeneous inputs. Extensive experiments demonstrate consistent and significant F1 score gains of $3-4\%$ over prior state-of-the-art methods on two datasets. The results highlight the importance of complementary speech features and attention modeling for improved detection of varied stuttering core behaviuors.

Despite the promising results, the proposed model is trained and evaluated on datasets with fixed-length segments. In Sep-28k and FluncyBank datasets, $40 - 250$ three-second intervals near the speech pause segment are extracted and annotated for each stuttering speech. In severe stuttering cases, the repetition, block, and prolongation might exceed the segment duration and cause those classes to be misclassified. Unlike fixed-length, which may truncate or pad stuttering events, variable-length segments can capture these events and may enhance the performance of SED. Therefore, employing a multi-feature fusion approach, using time-domain features (ZCR, SFO and FF), ASR embeddings, and auditory-based spectral features with variable-length segment processing provides a future direction for this work. Although efforts are devoted to advancing speech research and applications for stuttering events detection in the English language, the availability and reliability of training data is the main challenge affecting existing models' performance. Using generative and diffusion models to generate more reliable data suggests directions for future research.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. et al. (2016). Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)* (pp. 265–283).

Afroz, F., & Koolagudi, S. G. (2019). Recognition and Classification of Pauses in Stuttered Speech Using Acoustic Features. In *2019 6th International Conference on Signal Processing and Integrated Networks, SPIN 2019* (pp. 921–926). Institute of Electrical and Electronics Engineers Inc. doi:10.1109/SPIN.2019.8711569.

Al-Banna, A.-K., Edirisinghe, E., & Fang, H. (2022a). Stuttering detection using atrous convolutional neural networks. In *2022 13th International*

Conference on Information and Communication Systems (ICICS) (pp. 252–256). doi:10.1109/ICICS55353.2022.9811183.

Al-Banna, A.-K., Edirisinghe, E., Fang, H., & Hadi, W. (2022b). Stuttering Disfluency Detection Using Machine Learning Approaches. *Journal of Information & Knowledge Management*, *21*. doi:10.1142/S0219649222500204.

Alharbi, S., Hasan, M., Simons, A. J., Brumfitt, S., & Green, P. (2018). A lightly supervised approach to detect stuttering in children's speech. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* (pp. 3433–3437). International Speech Communication Association volume 2018-Septe. doi:10.21437/Interspeech.2018-2155.

Alim, S. A., & Rashid, N. K. A. (2018). Some commonly used speech feature extraction algorithms. In R. Lopez-Ruiz (Ed.), *From Natural to Artificial Intelligence* chapter 1. Rijeka: IntechOpen. doi:10.5772/intechopen.80419.

Almutairi, M., Gabralla, L. A., Abubakar, S., & Chiroma, H. (2022). Detecting elderly behaviors based on deep learning for healthcare: Recent advances, methods, real-world applications and challenges. *IEEE Access*, *10*, 69802–69821. doi:10.1109/ACCESS.2022.3186701.

Baevski, A., Zhou, H., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. arXiv:arXiv:2006.11477.

Barrett, L., Hu, J., & Howell, P. (2022). Systematic Review of Machine Learning Approaches for Detecting Developmental Stuttering. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *30*, 1160–1172. URL: https://ieeexplore.ieee.org/document/9729855/. doi:10.1109/TASLP.2022.3155295.

Bayerl, S. P., Wagner, D., Noeth, E., & Riedhammer, K. (2022). Detecting Dysfluencies in Stuttering Therapy Using wav2vec 2.0. In *Interspeech 2022* (pp. 2868–2872). ISCA: ISCA. doi:10.21437/Interspeech.2022-10908.

Bernstein Ratner, N., & MacWhinney, B. (2018). Fluency Bank: A new resource for fluency research and practice. *Journal of Fluency Disorders*, *56*, 69–80. doi:10.1016/j.jfludis.2018.03.002.

Böck, S., Krebs, F., & Schedl, M. (2012). Evaluating the online capabilities of onset detection methods. In *ISMIR* (pp. 49–54).

Bäckström, T., Räsänen, O., Zewoudie, A., Zarazaga, P. P., Koivusalo, L., Das, S., Mellado, E. G., Mansali, M. B., & Ramos, D. (2014). *Introduction to Audio Analysis*. Oxford: Academic Press. URL: https://www.sciencedirect.com/science/article/pii/B9780080993881000091. doi:https://doi.org/10.1016/B978-0-08-099388-1.00009-1.

Bäckström, T., Räsänen, O., Zewoudie, A., Zarazaga, P. P., Koivusalo, L., Das, S., Mellado, E. G., Mansali, M. B., & Ramos, D. (2022). *Introduction to Speech Processing*. (2nd ed.). URL: https://speechprocessingbook.aalto.fi. doi:10.5281/zenodo.6821775.

Chee, L. S., Ai, O. C., Hariharan, M., & Yaacob, S. (2009). Automatic detection of prolongations and repetitions using lpcc. In *2009 International Conference for Technical Postgraduates (TECHPOS)* (pp. 1–4). doi:10.1109/TECHPOS.2009.5412080.

Dash, A., Subramani, N., Manjunath, T., Yaragarala, V., & Tripathi, S. (2018). Speech Recognition and Correction of a Stuttered Speech. In *2018 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2018* (pp. 1757–1760). doi:10.1109/ICACCI.2018.8554455.

Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In J. Burstein,

C. Doran, & T. Solorio (Eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (pp. 4171–4186). Association for Computational Linguistics. URL: https://doi.org/10.18653/v1/n19-1423. doi:10.18653/v1/n19-1423.

Ferrand, C. T. (2017). *Speech Science: An Integrated Approach to Theory and Clinical Practice (Pearson Communication Sciences and Disorders)*. (4th ed.). Pearson.

Hariharan, M., Chee, L. S., Ai, O. C., & Yaacob, S. (2012). Classification of speech dysfluencies using LPC based parameterization techniques. *Journal of Medical Systems*, *36*, 1821–1830.

Jouaiti, M., & Dautenhahn, K. (2022). Dysfluency Classification in Stuttered Speech Using Deep Learning for Real-Time Applications. *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (pp. 6482–6486). URL: https://ieeexplore.ieee.org/document/9746638/. doi:10.1109/ICASSP43922.2022.9746638.

Kourkounakis, T., Hajavi, A., & Etemad, A. (2020). Detecting Multiple Speech Disfluencies Using a Deep Residual Network with Bidirectional Long Short-Term Memory. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* (pp. 6089–6093). Institute of Electrical and Electronics Engineers Inc. volume 2020-May. doi:10.1109/ICASSP40776.2020.9053893. arXiv:1910.12590.

Kourkounakis, T., Hajavi, A., & Etemad, A. (2021). FluentNet: End-to-End Detection of Stuttered Speech Disfluencies with Deep Learning. *IEEE/ACM Transactions on Audio Speech and Language Processing*, *29*, 2986–2999. doi:10.1109/TASLP.2021.3110146.

Lea, C., Huang, Z., Jain, D., Tooley, L., Liaghat, Z., Thelapurath, S., Findlater, L., & Bigham, J. P. (2022). Nonverbal sound detection for dis-

ordered speech. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 7397–7401). doi:10.1109/ICASSP43922.2022.9747227.

Lea, C., Mitra, V., Joshi, A., Kajarekar, S., & Bigham, J. P. (2021). SEP-28k: A Dataset for Stuttering Event Detection from Podcasts with People Who Stutter. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6798–6802). IEEE. URL: https://ieeexplore.ieee.org/document/9413520/. doi:10.1109/ICASSP39728.2021.9413520.

Li, B., Muñoz, J. P., Rong, X., Chen, Q., Xiao, J., Tian, Y., Arditi, A., & Yousuf, M. (2019). Vision-based mobile indoor assistive navigation aid for blind people. *IEEE Transactions on Mobile Computing*, *18*, 702–714. doi:10.1109/TMC.2018.2842751.

Liu, J., Wumaier, A., Wei, D., & Guo, S. (2023). Automatic speech disfluency detection using wav2vec2.0 for different languages with variable lengths. *Applied Sciences*, *13*. URL: https://www.mdpi.com/2076-3417/13/13/7579. doi:10.3390/app13137579.

Mahesha, P., & Vinod, D. (2016). Gaussian mixture model based classification of stuttering dysfluencies. *Journal of Intelligent Systems*, *25*, 387–399. URL: https://doi.org/10.1515/jisys-2014-0140. doi:doi:10.1515/jisys-2014-0140.

Mauch, M., & Dixon, S. (2014). Pyin: A fundamental frequency estimator using probabilistic threshold distributions. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 659–663). doi:10.1109/ICASSP.2014.6853678.

McFee, B., Raffel, C., Liang, D., Ellis, D. P., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and music signal analysis in python. In *Proceedings of the 14th python in science conference*. volume 8.

Mesaros, A., Heittola, T., Virtanen, T., & Plumbley, M. D. (2021). Sound Event Detection: A tutorial. *IEEE Signal Processing Magazine*, *38*, 67–83. URL: https://ieeexplore.ieee.org/document/9524590/. doi:10.1109/MSP.2021.3090678.

Mitra, V., Huang, Z., Lea, C., Tooley, L., Georgiou, P., Kajarekar, S., & Bigham, J. (2021). Analysis and tuning of a voice assistant system for dysfluent speech. In *Interspeech*. URL: https://arxiv.org/pdf/2106.11759.pdf.

Mohapatra, P., Pandey, A., Islam, B., & Zhu, Q. (2022). Speech Disfluency Detection with Contextual Representation and Data Distillation. In *Proceedings of the 1st ACM International Workshop on Intelligent Acoustic Systems and Applications* (pp. 19–24). New York, NY, USA: ACM. URL: https://dl.acm.org/doi/10.1145/3539490.3539601. doi:10.1145/3539490.3539601.

Pálfy, J. (2014). Analysis of dysfluencies by computational intelligence. In *Computer Science*.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, *12*, 2825–2830.

Ravikumar, K. M., Rajagopal, R., & Nagaraj, H. C. (2009). An approach for objective assessment of stuttered speech using mfcc features. In *Computer Science*.

Riley, J., Riley, G., & Maguire, G. (2004). Subjective Screening of Stuttering severity, locus of control and avoidance: Research edition. *Journal of Fluency Disorders*, *29*, 51–62. doi:10.1016/j.jfludis.2003.12.001.

Ronald B. Gillam, T. P. M. (2022). *Communication Sciences and Disorders: From Science to Clinical Practice: From Science to Clinical Practice*. (4th ed.). Jones & Bartlett Learning.

Sanghyun Woo, Jongchan Park, Joon-Young Lee, I. S. K. (2018). *CBAM: Convolutional Block Attention Module*. Springer Nature.

Sheikh, S. A., Sahidullah, M., Hirsch, F., & Ouni, S. (2021). StutterNet: Stuttering Detection Using Time Delay Neural Network. *European Signal Processing Conference*, *2021-August*, 426–430. doi:10.23919/EUSIPCO54536.2021.9616063. arXiv:2105.05599.

Sheikh, S. A., Sahidullah, M., Hirsch, F., & Ouni, S. (2022). Introducing ecapa-tdnn and wav2vec2.0 embeddings to stuttering detection. arXiv:arXiv:2204.01564.

Sleem, A., & Elhenawy, I. (2022). An Attentive Convolutional Recurrent Network for Fake News Detection. *International Journal of Advances in Applied Computational Intelligence*, *2*, 08–14. URL: https://www.americaspg.com/articleinfo/31/show/1802. doi:10.54216/IJAACI.020101.

Tan, T. S., Helbin-Liboh, Ariff, A. K., Ting, C. M., & Salleh, S. H. (2007). Application of Malay speech technology in Malay speech therapy assistance tools. *2007 International Conference on Intelligent and Advanced Systems, ICIAS 2007*, (pp. 330–334). doi:10.1109/ICIAS.2007.4658401.

Villegas, B., Flores, K., Pacheco-Barrios, K., & Elias, D. (2019). Monitoring of respiratory patterns and biosignals during speech from adults who stutter and do not stutter: A comparative analysis. In *International Symposium on Medical Information and Communication Technology, ISMICT*. volume 2019-May. doi:10.1109/ISMICT.2019.8743844.

World Health Organisation (2010). International statistical classification of diseases and related health problems 10th revision (icd-10) version for 2010. https://icd.who.int/browse10/2010/en#/F98.5, Accessed on 6 February 2021.