**An Enhanced Stuttering Detection Using Wav2vec and BERT model**

G. Narayanamma Institute of Science and Technology

**Abstract**

Stuttering is a speech disorder that affects the fluency of speech, posing significant communication challenges. Machine learning and deep learning approaches have shown promise in the automated detection and analysis of stuttering, providing valuable tools for speech therapists and researchers. This study proposes an advanced system for automated stuttering detection, combining both auditory and linguistic information. The system utilizes the Wav2vec model to convert audio recordings into matrix representations and Agnostic BERT to generate vector representations from transcripts. These vectors are merged to form a comprehensive feature set that captures both the acoustic and textual aspects of speech. A binary classifier is then employed to differentiate between "stutterers" and "non-stutterers." The model is trained using the SEP-28K dataset and tested on real-time data. The system not only identifies the presence of stuttering but also quantifies the percentage and duration of stuttering occurrences, pinpointing affected characters in the speech. The proposed approach achieved a 68% accuracy rate, with a detailed performance evaluation based on accuracy, precision, recall, and F1-score metrics, highlighting its potential for stuttering severity assessment and therapeutic applications.

**I. Introduction**

Pronunciation is the way in which a word or language is spoken and it is an aspect of effective communication. It involves intonation, sounds, rhythm. Accurate pronunciation ensures that the intended message is conveyed clearly as clear and comprehensive speech is the vital backbone for strong communication and is understood by listeners. There are variations in pronunciation due to factors like regional accents, dialects, and individual speech habits. Proper pronunciation is not just about individual sounds but also about the overall fluency and rhythm of speech. Fluency and rhythm contribute to the naturalness of spoken language. Disruptions in these elements can result in misunderstandings and miscommunication.

Stuttering, or stammering, is a speech disorder characterized by disruptions like sound repetitions, prolonged sounds, and blocks. It often begins in early childhood and can persist into adulthood, affecting communication and quality of life. Anxiety about speaking can worsen these disfluencies, leading to social withdrawal. Early detection is essential for effective intervention, which can include modifying speech patterns, managing anxiety, and enhancing communication skills. Accurate stuttering detection also helps reduce the social stigma associated with the disorder, contributing to better emotional well-being. Automated stuttering detection systems are a significant advancement in this field.

**II. Related work**

Recent advancements in detecting stuttered speech have leveraged various deep learning models and techniques, leading to significant improvements in accuracy and efficiency. Among these, FluentNet, a deep neural network that integrates a Squeeze-and-Excitation Residual CNN, bidirectional LSTM layers, and an attention mechanism, has achieved state-of-the-art results on the UCLASS dataset. Its performance was further validated with the newly introduced LibriStutter dataset [7].

Bayerl et al. [13] demonstrated the effectiveness of fine-tuning Wav2vec 2.0 for stuttering classification, achieving significant improvements on the FluencyBank and KSoF datasets. By adapting the pre-trained Wav2vec 2.0 model to the specific task of stuttering detection, they were able to leverage its powerful feature extraction capabilities to better capture the nuanced patterns of stuttered speech. Their results highlight the value of fine-tuning large pre-trained models for domain-specific applications, contributing to the advancement of automated stuttering detection techniques.

A study utilized MFCC and phoneme probabilities in a BLSTM network to detect and classify stuttering dysfluencies, though it faced challenges with prolongations and word repetitions due to dataset imbalances [6]. Ravikumar and Rajagopal achieved high accuracy in detecting syllable repetitions using MFCC features with an SVM classifier [9]. By extracting MFCC features from the speech signals, they effectively captured the acoustic characteristics associated with stuttering dysfluencies, specifically syllable repetitions. The SVM classifier, known for its robustness in handling small datasets and high-dimensional spaces, was able to accurately distinguish between fluent and dysfluent speech patterns. The success of their model underscores the potential for integrating classic feature extraction methods with efficient classifiers to address specific stuttering phenomena. StutterNet, a model based on a deep neural network and time-delay neural network, was introduced to detect stuttering from acoustic signals without relying on ASR systems [12].

**III. Proposed System**

A diagram of a company

Description automatically generatedThe proposed system as shown below in Fig 3.1., leverages advanced machine learning models to accurately identify individuals who stutter. The core of this system integrates Wav2vec and Agnostic BERT to comprehensively analyse speech by combining auditory and linguistic information. Wav2vec, a groundbreaking approach, pre-trains a model on large, unlabelled datasets of raw audio waveforms, enabling it to learn meaningful speech representations without heavily relying on transcriptions. This process begins with the Wav2Vec2Tokenizer, which prepares audio data by tokenizing, encoding, and managing audio input lengths, transforming the audio into numerical representations suitable for analysis.

***Fig 3.1*** *Flow diagram of proposed system*

Agnostic BERT enhances the system's linguistic analysis capabilities by generating vector representations from transcripts, adapting to multiple languages without language-specific training. The combination of audio and linguistic vectors is fed into a binary classifier, differentiating between "stutterers" and "non-stutterers," with the alignment refined through Connectionist Temporal Classification (CTC) for improved performance.

**Wav2ec 2.0**

Wav2vec is an innovative model designed for pre-training on large, unlabelled datasets of raw audio waveforms. The architecture of Wav2Vec 2.0 model is presented below in Fig 3.2. Unlike traditional automatic speech recognition (ASR) models that require extensive transcribed audio data, Wav2vec leverages self-supervised learning to predict future waveform samples based on past audio. The Wav2vec model is utilized to effectively handle raw audio waveforms and extract meaningful speech representations. By leveraging Wav2vec's self-supervised learning capabilities, we were able to train the model on large, unlabelled datasets of natural speech.

A diagram of a transformer

Description automatically generatedThis enabled the model to predict future waveform samples based on past audio, capturing nuanced auditory features without needing extensive transcriptions. This made Wav2vec an essential tool for our project, as it provided a robust foundation for detecting stuttering instances in diverse speech samples.

***Fig 3.2*** *Wav2Vec Architecture*

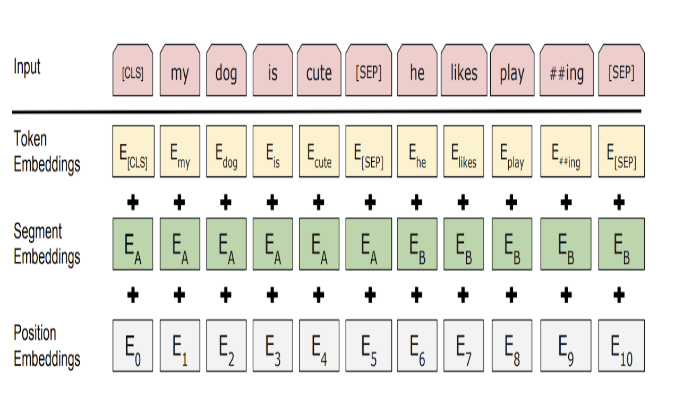
To prepare the audio data for analysis, we used the Wav2Vec2Tokenizer, which played a crucial role in transforming the raw audio waveforms into numerical representations.

The tokenizer segmented the audio, encoded these segments, and managed the audio input lengths to ensure they were suitable for processing by the Wav2vec model. This preprocessing step allowed the model to effectively analyse and extract essential audio features, facilitating the detailed examination of speech characteristics necessary for accurate stuttering detection.

After pre-training the Wav2vec model on unlabelled audio data, we fine-tuned it using labelled datasets with stuttering instances. By employing Connectionist Temporal Classification (CTC) loss, we aligned the predicted labels with the audio inputs, enabling the model to adapt to specific stuttering detection tasks. This approach allowed us to generate high-fidelity transcriptions of the audio, which were crucial for identifying stuttering accurately. The detailed and precise account of the audio input provided by Wav2vec significantly enhanced the model's precision and recall, ensuring that both the presence and absence of stuttering were correctly identified in real-world scenarios.

**Agnostic BERT**

Agnostic BERT enhances the system's linguistic analysis capabilities by generating vector representations from textual transcripts. The architecture of Agnostic BERT is presented below in Fig 3.3. After Wav2vec processes the raw audio to produce transcripts, Agnostic BERT takes over to handle any errors due to stuttering. It tokenizes the transcript into individual words, pre-processes them, and then generates vector embeddings. The CLS token output summarizes the entire transcript into a single vector that encapsulates its semantic meaning and syntactic structure, providing crucial linguistic context for accurately identifying stuttering events.

By incorporating Agnostic BERT, we complement the transcriptions provided by Wav2vec with enhanced text processing and understanding. Developed to handle various natural language processing tasks, Agnostic BERT extracts contextual embeddings from the transcriptions, which are vital for understanding the nuances in speech. This model reduces biases and adapts to multiple languages without needing language-specific training, ensuring that the contextual information in the transcriptions is accurately captured. This thorough understanding of the text allows our system to analyse the flow and context of speech, improving the accuracy of stuttering detection.

***Fig 3.3*** *Agnostic BERT Architecture*

The integration of Agnostic BERT helps the model distinguish between fluent and stuttered speech segments, thereby enhancing the overall accuracy of stuttering detection. This synergy between high-quality transcriptions from Wav2vec and the contextual understanding provided by Agnostic BERT is vital for our project's success. The combination ensures that both auditory and linguistic features are effectively utilized, leading to a more robust and reliable stuttering detection system, as demonstrated by our experimental results.

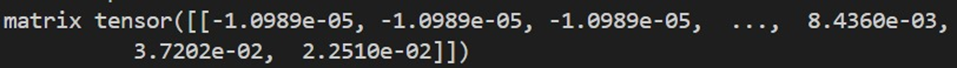
**IV. Methodology**

The complete details of the Methodology are shown below in Fig 4.3. Here is the detailed breakdown of the Methodology:

**Audio File Preparation**

First, the audio file is loaded using the torchaudio library, which retrieves its waveform and sample rate. Ensuring the audio sample rate matches the standardized 16,000 Hz is crucial for compatibility with the audio processing models. If the sample rate differs, the audio file is resampled to 16,000 Hz. This step ensures that subsequent processes, such as feature extraction and model input, are standardized and can perform optimally.

**Audio Feature Extraction**

Once the audio is prepared, it is tokenized into smaller segments using a tokenizer. These tokenized segments are then fed into the Wav2Vec2 model. Wav2Vec2, a powerful model for speech recognition, converts the audio waveforms into a matrix representation that captures essential audio features. The matrix formed for a sample is illustrated as shown in Fig. 4.1. This matrix representation distills the audio into a form that retains crucial information about the speech patterns and characteristics, facilitating accurate downstream processing.

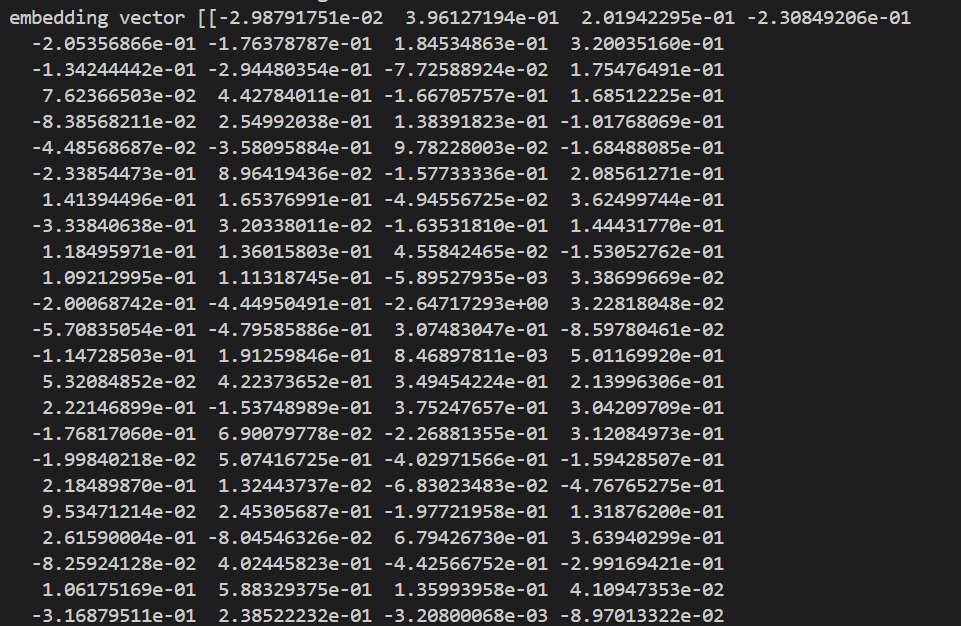
***Fig 4.1*** *Matrix representation of audio sample*

**Transcript Generation and Decoding**

The next step involves generating a transcript from the Wav2Vec2 model. The initial transcript may contain errors such as repeated or improperly expressed characters, which can occur in instances of stuttering. To refine this transcript, a tokenizer decodes it, matching tokens to their vocabulary to produce readable text. This step is critical for transforming the raw output of the model into a more understandable and user-friendly format.

**Generate Vector Embeddings with Agnostic BERT**

After obtaining the readable text transcript, it is tokenized into individual words. Pre-processing steps are applied to these tokenized words to prepare them for input into the Agnostic BERT model. Agnostic BERT is used to generate vector embeddings for the transcript. The embedded vector for a sample is illustrated as shown in Fig. 4.2. The embeddings capture the semantic meaning and syntactic structure of the text. The CLS (Classification) token output from Agnostic BERT is extracted, summarizing the entire transcript into a single vector. This vector encapsulates the overall meaning and context of the transcript, making it a valuable representation for further analysis.



***Fig 4.2*** *Embedded vector representation of audio sample*

**Stuttering Detection**

To detect stuttering, the audio is transcribed into text, forming a transcribed segment. The Longest Common Subsequence (LCS) algorithm is then used to compare the transcribed segment with the expected transcript. The LCS algorithm identifies the longest sequence of characters that appear in both the transcribed and expected texts, serving as a benchmark for comparison. Segments in the transcribed text where characters do not match the common sequence determined by the LCS algorithm are flagged as potential stuttering segments. This process helps isolate specific parts of the text where stuttering may have occurred.

**Calculate Stuttering Percentage**

The severity of stuttering is quantified by calculating the stuttering percentage (1). This involves comparing the total length of characters in the identified stuttering segments to the length of the expected transcript. The stuttering percentage provides a measurable indication of the extent of stuttering present in the audio. This metric is crucial for evaluating and monitoring stuttering over time or across different conditions.

A diagram of a diagram of a computer

Description automatically generated ……………(1)

***Fig 4.3*** *Proposed Architecture Design*

**Calculating Results**

The final step involves presenting the results. The stuttering percentage is provided as a measure of stuttering severity. Additionally, specific characters affected by stuttering are identified and listed. The full transcribed text from the audio is outputted, along with the expected transcription. This comprehensive output allows for a detailed analysis of the stuttering, facilitating further study or therapeutic interventions. The severity of stuttering is taken from the mapping of different ranges of stuttering percentages and the severity level as shown in Table 4.1.

***Table 4.1*** *Severity level of stuttering as per stuttering percentage*

|  |  |
| --- | --- |
| **Range of stuttering percentages** | **Severity level** |
| Less than 5% | Trivial |
| 5 – 10% | Trivial to Intermediate |
| 10-15% | Intermediate |
| 15-20% | Intermediate to Acute |
| Above 20% | Acute |

**V. Datasets**

**SEP-28k**

The SEP-28k dataset is a comprehensive collection of around 385 podcasts featuring individuals who stutter or have pronunciation problems. This dataset is publicly available under specific conditions, retaining the original copyrights of the podcast owners. It encompasses a diverse range of English speakers of different genders, voices, and speaking styles. The primary purpose of this dataset is to facilitate the classification of stuttering and pronunciation issues in speech. However, the dataset only includes audio clips of individuals who stutter, necessitating the creation of a complementary dataset of non-stuttering individuals for effective model training.

To build the non-stuttering dataset, a systematic approach was adopted. Initially, selected podcasts were divided into equal segments of 5 seconds each. This segmentation aimed to precisely identify stuttering occurrences within the audio clips. Subsequently, each audio segment was processed using the wav2vec model, which transcribed the audio. By comparing the transcriptions with the original audio segments, it was determined whether the segment featured a stuttering individual. This meticulous process ensured the creation of a balanced dataset, containing both stuttering and non-stuttering audio clips, essential for training robust and accurate speech recognition models.

**Real time Dataset**

The real-time dataset was collected from individuals to ensure the model accurately reflects natural speech patterns and stuttering instances. Volunteers were recorded speaking in various environments, capturing a wide range of audio qualities and speaking styles. This dataset includes diverse speech samples with different accents, speaking speeds, background noises, and disruptions (as shown in Fig. 5.1(a) and (b)), which are essential for training a robust stuttering detection model. By using real-world data, the model learns to handle the complexities of everyday speech, making it more effective and reliable in identifying stuttering across different contexts and speakers. This real-time collection approach ensures the model is well-equipped to function in practical, real-world applications.

(b)

***Fig 5.1*** *waveform of (a) non-stuttering sample (b) stuttering sample form real time data*

(a)

**VI. Results and analysis**

After training and testing the model, we achieved a 68% accuracy rate. The evaluation of the model's performance was conducted using several metrics: accuracy, precision, recall, and F1-score as shown in Table 6.1. Each metric offers a unique perspective on the model's effectiveness, providing a comprehensive assessment of its performance. The experiments were conducted with various activation functions and different numbers of layers. The final layer uses a 'sigmoid' function, indicating that the problem is a binary classification. Consequently, the model's output is a probability value ranging from 0 to 1. Below, we explain the methods used to evaluate the classifier.

**Precision** measures the proportion of correctly predicted positive cases out of all cases predicted as positive. This metric is particularly important when the cost of false positives is high, as it helps evaluate how accurately the model identifies positive instances.

Precision (Pre) = TP / (TP + FP)

**Recall** provides the proportion of correctly predicted positive cases out of all actual positive cases.

Recall (Re) = TP / (TP + FN)

**F1-score** considers both recall and precision, integrating these two metrics to strike a balance between reducing false predictions and correctly identifying positive cases.

F1-score = 2 \* (precision \* recall) / (precision + recall)

|  |  |  |  |
| --- | --- | --- | --- |
| Accuracy | Precision | Recall | F1-score |
| 68% | 0.83 | 0.45 | 0.59 |

***Table 6.1*** *Performance metrics*

**VII. Conclusions and Future Work**

This study presented a comprehensive approach to automated stuttering detection by integrating audio and text representations using Wav2vec and Agnostic BERT models. The methodology involved preparing audio files, extracting audio features, generating transcripts, and creating vector embeddings that combine auditory and linguistic information. Stuttering detection was performed using the Longest Common Subsequence (LCS) algorithm to compare the transcribed segment with the expected transcript, identifying stuttering segments and calculating the stuttering percentage. The stuttering severity was then mapped based on different ranges of stuttering percentages, providing a detailed analysis of stuttering severity levels.

The proposed system demonstrated its effectiveness with a 68% accuracy rate, evaluated through metrics such as precision, recall, and F1-score. By combining auditory and linguistic features, the model captures subtle stuttering patterns that might be missed when relying on a single modality, enhancing detection accuracy and robustness. This dual-modality approach enriches the system's capability, making it a valuable tool for stuttering detection and assessment.

While the results are promising, the study's potential could be further unlocked by expanding the dataset and exploring more advanced modeling techniques. Future work should focus on collecting larger and more diverse datasets and refining models to enhance performance and generalizability. Additionally, integrating advanced deep learning techniques and exploring real-time applications could further advance the system's practical utility in clinical and therapeutic settings.

**VIII. References**

[1] Al, S., Hasan, M., Simons, A., Brumfitt, S., and Green, P., “Detecting stuttering events in transcripts of children’s speech,” Proceedings of Interspeech, pp. 2651–2655, 2017.

[2] Andrzej Czyzewski and Kostek, B., “Intelligent processing of stuttered speech,” Journal of Communication Disorders, pp. 143–171, 2003.

[3] Dash, A., Subramani, N., Manjunath, T., Yaragarala, V., and Tripathi, S., “Speech recognition and correction of a stuttered speech,” in Proc. Int. Conf. Adv. Comput., Commun. Informat., pp. 1757–1760, 2018.

[4] Feng, F., Yang, Y., Cer, D., Arivazhagan, N., and Wang, W., “Language-agnostic BERT sentence embedding,” arXiv preprint arXiv:2007.01852, 2020.

[5] Heeman, P., Lunsford, R., McMillin, A., and Yaruss, J., “Using clinician annotations to improve automatic speech recognition of stuttered speech,” Proceedings of Interspeech, pp. 2651–2655, 2016.

[6] Jouaiti, M., and Dautenhahn, K., “Dysfluency classification in stuttered speech using deep learning for real-time applications,” 2021.

[7] Kourkounakis, T., Hajavi, A., and Etemad, A., “FluentNet: End-to-End Detection of Stuttered Speech Disfluencies With Deep Learning,” Vol. 29, 2021.

[8] Lee, J., and Watanabe, S., “Intermediate loss regularization for CTC-based speech recognition,” IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), pp. 6224–6228, 2021.

[9] Ravikumar, H. K. M., and Rajagopal, R., “An approach for objective assessment of stuttered speech using MFCC features,” Digital Signal Processing: A Review Journal, vol. 9, pp. 19–24, 2019.

[10] Rudzicz, F., “Using articulatory likelihoods in the recognition of dysarthric speech,” IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 54, no. 3, pp. 430–444, March 2012.

[11] Sheikh, S. A., and Sahidullah, M., “Machine learning for stuttering identification: Review, challenges and future directions,” Neurocomputing, 2022.

[12] Sheikh, S. A., and Sahidullah, M., “Stutternet: Stuttering detection using time delay neural network,” arXiv preprint arXiv:2105.05599v2, Jun. 2021.

[13] Sebastian P. Bayerl, Dominik Wagner, Elmar Noth, Korbinian Riedhammer, “Detecting Dysfluencies in Stuttering Therapy Using wav2vec 2.0,” arXiv:2204.03417v2 [eess.AS], 16 Jun 2022.

[14] Zhang, Y., Yu, H., and Liu, H., “FluentNet: End-to-End Detection of Stuttered Speech Disfluencies with Deep Learning,” arXiv preprint arXiv:1904.05189, 2021.