MethodsX 13 (2024) 103050

Contents lists available at ScienceDirect

MethodsX

journal homepage: www.elsevier.com/locate/methodsx

Evaluative comparison of machine learning algorithms for stutter detection and classification ☆

Ramitha V, Rhea Chainani, Saharsh Mehrotra, Sakshi Sah, Smita Mahajan*

Department of Artificial Intelligence and Machine Learning, Symbiosis International University Symbiosis Institute of Technology, Pune, Maharashtra, India

ARTICLE INFO

Method name:
Support Vector Classifier, Random Forest, Decision Tree, K-Nearest Neighbors, Logistic Regression

Keywords:
Automatic dysfluency detection
Comparative analysis
Machine learning
Stuttering
Speech disorder

ABSTRACT

Stuttering is a neuro-developmental speech disorder that interrupts the flow of speech due to involuntary pauses and sound repetitions. It has profound psychological impacts that affect social interactions and professional advancements. Automatically detecting stuttering events in speech recordings could assist speech therapists or speech pathologists track the fluency of people who stutter (PWS). It will also assist in the improvement of the existing speech recognition system for PWS. In this paper, the SEP-28k dataset is utilized to perform comparative analysis to assess the performance of various machine learning models in classifying the five dysfluency types namely Prolongation, Interjection, Word Repetition, Sound Repetition and Blocks.

- The study focuses on automatically detecting stuttering events in speech recordings to support speech therapists and improve speech recognition systems for people who stutter (PWS).
- The SEP-28k dataset is used to perform a comparative analysis of different machine learning models.
- The research examines the impact of key acoustic features on model accuracy while addressing challenges such as class imbalance.

Specifications table

Subject area:	*Computer Science*
More specific subject area:	Machine Learning
Name of your method:	Support Vector Classifier, Random Forest, Decision Tree, K-Nearest Neighbors, Logistic Regression
Name and reference of original method:	Bayerl, Sebastian P., Dominik Wagner, Elmar Nöth, Tobias Bocklet, and Korbinian Riedhammer. "The influence of dataset partitioning on dysfluency detection systems." In International Conference on Text, Speech, and Dialogue, pp. 423–436. Cham: Springer International Publishing, 2022 doi: https://doi.org/10.48550/arXiv.2206.03400
Resource availability:	*The URL for the dataset is given below:* https://machinelearning.apple.com/research/stuttering-event-detection

Background

A detailed review of the past two decades of research reveals a concerted effort to develop automated methods for identifying stuttering. Researchers have meticulously examined datasets, acoustic features and classification methodologies to elucidate the chal-

☆ **Related research article:** None.

* Corresponding author.
E-mail address: smita.mahajan@sitpune.edu.in (S. Mahajan).

https://doi.org/10.1016/j.mex.2024.103050
Received 31 October 2024; Accepted 12 November 2024
Available online 14 November 2024

lenges and opportunities in this domain. Key among these efforts is the extraction of acoustic features, where methods ranging from autocorrelation to spectral analysis have been explored. Recent advancements have seen the integration of convolutional neural networks (CNNs) with spectrogram representations, showcasing the potential for enhanced accuracy in stuttering identification tasks. The research meticulously outlines the various statistical machine learning techniques employed for stuttering detection and classification. In most cases, Support vector machines (SVM) [1], artificial neural networks (ANNs) [2] and hidden Markov models (HMM) [3] have been proved to be most effective. SVMs have emerged as the most prevalent classifier, demonstrating high accuracy across different types of stuttering [1,4,5]. Methods including k-nearest neighbor (k-NN) and linear discriminant analysis (LDA) have shown further promise [6], enriching the vast scope of stutter identification methodologies.

However, the past research also underscores a paradigm shift towards deep learning techniques, particularly CNNs [7] and recurrent neural networks (RNNs), in stuttering identification. These approaches offer automatic feature extraction capabilities and have exhibited superior performance compared to traditional methods. Recent studies, including the introduction of models like FluentNet [8] and StutterNet [9], have addressed data scarcity concerns and achieved remarkable accuracies on larger datasets. It is noteworthy that the accuracies achieved by these models are indeed impressive, with SVMs reporting high accuracies ranging from 75 % to 98 % across various stuttering types [10]. Similarly, CNN models have displayed average accuracies of 91.15 % to 91.75 % on different datasets [11]. While challenges persist in generalizing models to larger datasets and capturing diverse stuttering patterns effectively, the advancements in machine learning and deep learning offer renewed hope for the automated identification of stuttering.

The study of Stuttering Detection lies within the domain of speech and language processing. The main focus of the research is the identification and classification of the speech dysfluencies vis-a-vis stuttering. This paper essentially uses signal processing techniques to analyze the acoustic properties of speech. This includes extracting key features from speech signals such as the spectral features in order to develop high-level models which are capable of monitoring stuttered speech patterns.

The traditional method of speech therapists or pathologists analyzing the speech or the recordings of PWS manually to assess the type and severity of stuttering [12]. This method of detecting is time consuming and intense and requires PWS to regularly meet the speech therapists [13]. This calls for an automated system for this task.

The methodology proposed through this paper as shown in Fig. 1 aims to contribute to this area of research in the following manner:

1. Developing robust machine learning models for classifying the five classes of Stutter- Interjection, Prolongation, Blocks, Sound Repetitions and Word Repetitions.
2. Conducting analysis on the impact of various features extracted manually as well as using pre-trained models.
3. Performing comparative analysis on each class of stutter using various machine learning models.

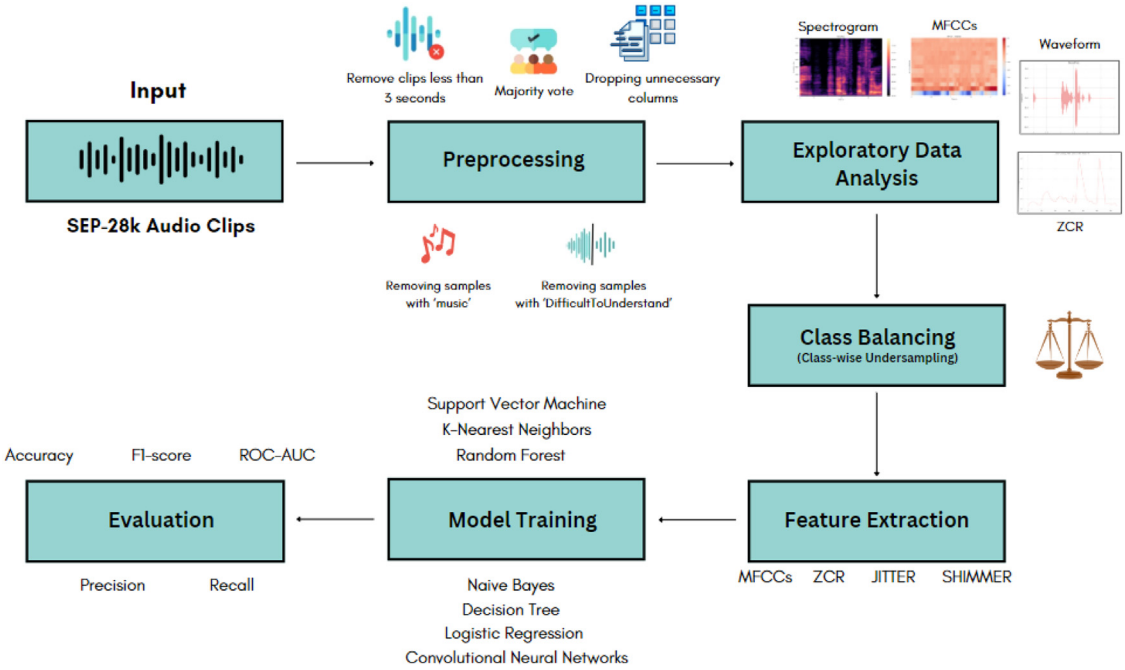

Fig. 1. Methodology Diagram.

Table 1
Stuttering labels.

Stuttering Labels	Definition	SEP-28k(%)
Block	Gasps for air or stuttered pauses.	12.0
Prolongation	Elongated syllable (e.g., [mmm]ommy).	10.0
Sound Repetition	Repeated syllables (e.g., I [pr-pr-pr-]prepared dinner).	8.3
Word/Phrase Repetition	The same word or phrase is repeated (e.g., I made [made] dinner).	9.8
No dysfluencies	Confirmation that none of the above is true.	56.9
Interjection	Common filler words such as “um” or “uh” or person-specific filler words that individuals use to cope with their stutter (e.g., some users frequently say “you know” as a filler).	21.2
Non-dysfluent Labels		
Natural Pause	There is a pause in speech that is not considered a block or other disfluency.	8.5
Difficult To Understand	It is difficult to understand the speech.	3.7
Unsure	An annotator selects this if they are not confident in their labeling.	0.1
No Speech	There is no speech in this clip. It is either silent or there is just background noise.	1.1
Poor Audio Quality	It is difficult to understand due to, for example, microphone quality.	2.1
Music	There is background music playing.	1.1

Method Details

Dataset Acquisition

In the methodology proposed SEP-28k dataset, which is publicly available in the Apple Machine Learning Research [14], has been used. Following the download of the podcast audio files in .mp3 format, a conversion process was employed to convert the files from .mp3 into .wav format. Subsequently, 3-second clips were extracted from the audio files for each podcast episode. The classes available in the dataset have been explained in Table 1.

Preprocessing

The initial step of preprocessing was deleting the audio clips which were not 3 s long and also to check if all the audio clips have a sampling rate of 16 kHz. Additionally, significant preprocessing was required due to the disparities between the annotators. Initially the dataset consisted of 11 columns dedicated for labeling purposes which made the dataset multi-label. Several of the labels were dropped due to their lack of any meaningful contribution to the model. The Unsure column was dropped. Moving forward, the audio clips labeled as music, which had introductory or concluding music from the podcast, were removed from the dataset, following which the music column was also dropped. Furthermore, for the DifficultToUnderstand column, the audio samples where two or more annotators agreed that the audio clip was difficult to understand were removed and then the DifficultToUnderstand column was also removed from the dataset.

Handling Class Imbalance

The SEP-28k data suffers from severe class imbalance. The NoStutteredWords class is overrepresented compared to the others. As can be seen in Fig. 2, the count of samples in the NoStutteredWords class alone equals the total count of samples in all the other classes combined. Within each of the individual classes also, class imbalance can be noticed. The negative class has a significantly greater number of samples than the positive class. Class imbalance is one of the most important problems that has to be addressed prior to training a model. Class imbalance results in the underrepresentation of the minority class where the learning is heavily dominated by the majority class [15]. In some scenarios, minority class samples are not detected by the model and hence results in the poor performance of the chosen models.

The proposed methodology addresses the class imbalance problem by considering each of the five classes — interjection, prolongation, word repetition, sound repetition and blocks as individual binary classifications. In each of the classes, undersampling was used i.e., after identifying the minority class, a subset of the majority class is randomly sampled to match the number of samples in the minority class. By sampling them in the method mentioned above, it is ensured that both the negative and the positive class samples are equally represented.

Feature Extraction

Features are a crucial part of any machine learning algorithm. Hence it is crucial to choose the desired features [16]. The primary features considered are Mel-Frequency Cepstral Coefficients (MFCCs) [17], Zero Crossing Rate, Jitter and Shimmer. Other sets of features that were considered are Pitch, Energy and Perceptual Linear Prediction (PLP) [18]. These features were not considered as the combination of these features with the primary features brought down the accuracy of the models. Experimentation was also done with the pretrained wav2vec 2.0 [19] model for feature extraction. The various features considered are:

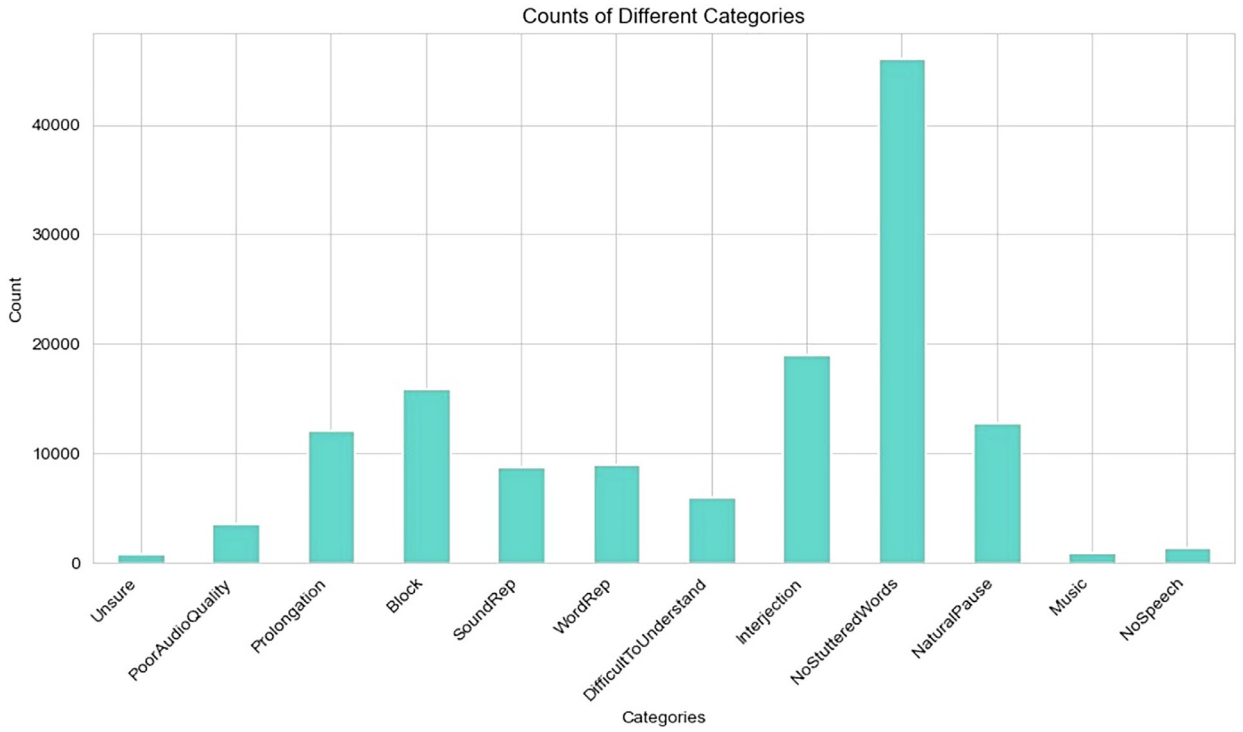

Fig. 2. Representation of Class Imbalance.

1. MFCCs:
   Mel-frequency cepstral coefficients or MFCCs represent the spectral characteristics extracted from the audio clips. The calculation of MFCCs involves conversion of audio clips into the frequency domain using Fourier transform (FT) and then mapping the spectrum to the mel scale. Further, logarithm of the mel spectrum is calculated and then Discrete Fourier Transform (DCT) is calculated to reduce the dimensionality.
2. Zero Crossing Rate:
   Zero Crossing Rate or ZCR represents the measure at which the audio signal changes its sign. ZCR is interpreted as the measure of noisiness of a signal. A high ZCR implies a rapid change in the waveform which is associated with a high frequency whereas a low ZCR implies steady or sustained changes.
3. Jitter:
   Jitter represents the variation of pitch periods in the consecutive cycles. Jitter typically represents the irregularity in the fundamental frequency (pitch) over time. Higher Jitter values indicate a greater variability in the fundamental frequency and corresponds to the presence of vocal disorders.
4. Shimmer:
   Shimmer represents the variation of amplitude or intensity of pitch periods in the consecutive cycles. Shimmer typically represents the irregularity in the intensity/loudness over time. Higher shimmer values indicate a greater variability in the intensity which also implies the presence of vocal disorders.

Model Training and Hyperparameter Tuning

After the extracted features are concatenated into an array, these features are given to the models for training. Here individual models have been considered for each class - namely, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Random Forest, Decision Tree and Naive Bayes. All the above mentioned models are trained for all the classes and the model which gives the highest accuracy is chosen for that particular class.

Hyperparameter tuning and cross validation techniques were applied on each of the models i.e., KNN, Logistic Regression, Decision Tree, Random Forest, and SVM, and just cross validation (with ten-fold cross-validation) for Naïve Bayes as it is a non-parametric model.

In the Random Forest model, the hyperparameter tuning was done by adjusting the number of estimators as well as the maximum depth. Number of estimators controls the number of decision trees in the random forest ensemble model. Here, the number of estimators is set to 100, which provides a balance between the stability of the model as well as the accuracy. The next hyperparameter is the maximum depth, which controls the limit of each tree. Here, the maximum depth is set to 10, which prevents the decision tree from being too deep and therefore prevents it from being overfitted.

Next is the Decision Tree model, where the hyperparameters that were tuned are maximum depth and criterion. Here, the maximum depth is set to 8, which helps the model generalize better, as maximum depth more than 8 tends to induce noise. The next hyperparameter is the criterion, which is the measure of the "impurity" or the randomness in the data. Here, the criterion is set as 'Gini,' which improves the homogeneity within each branch of the Decision Tree.

The K-Nearest Neighbors (KNN) model has two hyperparameters which were tuned, namely - number of neighbors (k) as well as the distance metric. The number of neighbors decides how many of the neighboring feature points in the feature space would be considered while a class is being predicted. Here, k is set as 5, which balances the model's sensitivity to outliers and generalization. The next hyperparameter is the distance metric, which here is 'Minkowski' with the default parameter as 2, which is equivalent to the Euclidean distance.

In the Support Vector Classifier (SVC), two hyperparameters were primarily focused on, namely - Kernel Type and the Regularization Parameter (C). The Kernel type determines the method by which the data is mapped to a higher-dimensional space. Here, a linear kernel is chosen because the data is linearly separable, and hence, using a more complex kernel like 'rbf' would lead to overfitting. The next hyperparameter is the Regularization Parameter (C), which is set to 1.0 here. The Regularization Parameter is responsible for controlling the trade-off between maximizing margin and reducing the error.

In the Naive Bayes model, the hyperparameter tuned is the Prior Distribution. The Naive Bayes chooses a different type of distribution depending on the nature of the data used. Here, Gaussian Naive Bayes was selected, which assumes the data to be normally distributed. In this type of distribution, the data follows a bell-shaped curve and is best suited for continuous data.

In the model Logistic Regression, the hyperparameter Regularization Strength (C) is set to 1.0, which is responsible for controlling the inverse of the strength of regularization. The higher the value of C, the lower is the regularization, which in turn allows for the model to be fitted very closely in the training data. This results in the model generalizing better by capturing the important patterns while ignoring the noise.

Overall, through these hyperparameter tuning steps, the precision and recall have increased, which implies the model was effective in minimizing the false positives and false negatives.

Method Validation

After the creation of unique subset for each dysfluency class by undersampling the majority class, six different machine learning models: Support Vector Machine (SVC), Logistic Regression, Naive Bayes, Random Forest and K-nearest Neighbors were trained on the dataset. Undersampling ensured that data used to train the model to identify one dysfluency was not biased towards a single class.

During the training of all the models except Naive Bayes, RandomizedSearchCV [20] was used in combination with ten-fold cross validation for hyperparameter tuning, resulting in 500 fits being performed for one model. This approach made sure the best possible hyperparameters were chosen for each model. For Naive Bayes, only ten-fold cross validation was performed.

After training, the models were evaluated on an unseen test set. The test set comprised 30 % of the subset created for that particular dysfluency. Out of the six models, the model that achieved the highest accuracy on the test data was selected as the final model for detecting that particular dysfluency in audio files. The validation and testing accuracy were extremely close indicating optimal learning and guaranteeing similar results on unseen real world data.

In this approach, the model generalization has been improved by addressing the class imbalance issues, choosing a specific model for each type of stutter as well as by choosing appropriate features. Class imbalance is one of the main issues which affects the generalization of a model. Here, the training data was balanced by undersampling the majority class in the binary classification. The next method that improves the generalization is choosing the individual models for each type of stutter which results in a tailored approach for detecting specific types of stutter. Finally, the generalization is improved by choosing appropriate features such as MFCCs, Zero Crossing Rate, Jitter as well as Shimmer while other features such as Pitch, Energy and PLP were excluded.

This rigorous process ensured that the final models resulting from this method of creation were trained on balanced data and were well-tuned and robust.

Results and Discussions

The results shown in Table 2 reveal the performance of various machine learning models across various types of stutter. The highest accuracy for each type of stutter was obtained through a different model which suggests that no single model is adequate for all dysfluency types.

Table 2
Accuracy scores of different models.

	Prolongation	SoundRep	WordRep	Interjection	Block
RandomForest	0.66	0.63	0.61	0.65	0.58
SVM	0.63	0.63	0.63	0.67	0.60
KNN	0.60	0.63	0.60	0.64	0.60
Naive Bayes	0.60	0.57	0.56	0.58	0.51
Logistic Regression	0.55	0.56	0.56	0.60	0.54
Decision Trees	0.58	0.58	0.55	0.57	0.56

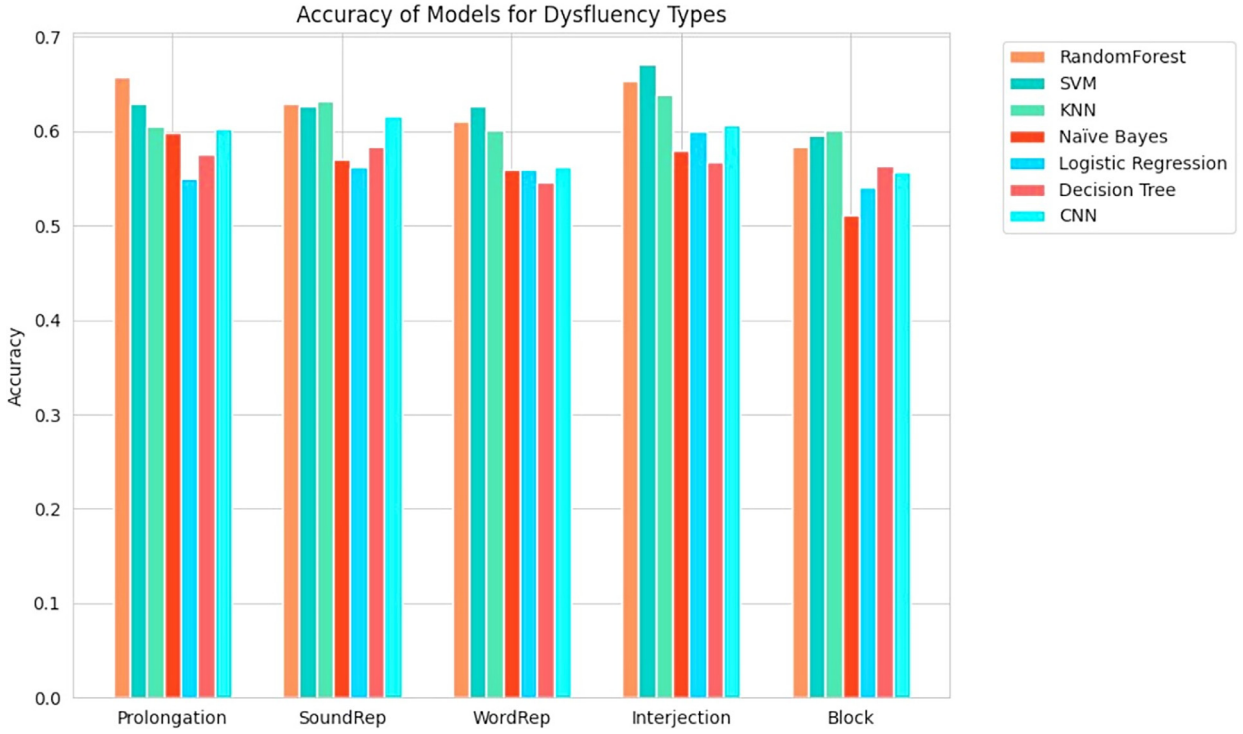

Fig. 3. Comparison of Accuracy of different models for all the dysfluencies.

Table 3
Comparison with base paper.

	ML Model Selected	F1-score	F1-score from [1]
Blocks	KNN	**0.55**	0.40
Interjection	SVM	0.67	0.73
Prolongation	Random Forest	**0.64**	0.54
Sound Repetition	KNN	0.65	0.70
Word Repetition	SVM	**0.63**	0.50

From Table 2, it can be inferred that the best accuracy was achieved using RandomForest for Prolongation (0.66), KNN for Sound Repetition (0.63), SVM for Word Repetitions (0.63), SVM for Interjection (0.67), and KNN for Blocks (0.60).

From Fig. 3, it can be inferred how certain models such as RandomForest and SVM consistently outperform the other models in various types of stutter. This variation highlights the unique characteristics of each type of dysfluency, suggesting that using different models for specific types of stuttering may be more effective than relying on a single model for all types.

The model used in [1] is SVM, meanwhile each dysfluency in the proposed work has been implemented by using a different model. From Table 3, it can be inferred that the proposed model outperforms the models used in [1] for the classes - Blocks (0.55), Prolongations (0.64) and Word Repetitions (0.63) in terms of F1-score.

From Fig. 4 it can be observed that the Word Repetitions class shows the most balanced performance with approximately equal counts in both correct identifications (True positives: 443, True negatives: 444) and errors (False Positives: 264, False Negatives: 264). Following closely is the Interjection class, which also exhibits a nearly equal distribution of correct identifications and errors (True positives: 951, True negatives: 964, False Positives: 487, False Negatives: 500). The model used for both of these dysfluencies was SVC, suggesting its effectiveness in achieving such balance compared to other models. The remaining classes show moderate performance with varying degrees of imbalance between correct identifications and errors.

The ROC AUC values from Fig. 5 indicate variations in discriminatory performance among classifiers across dysfluency types. Among the highest accuracy models chosen for each class, the Support Vector Classifier (SVC) generally performs well, particularly evident in its higher ROC AUC scores for interjections (0.73) and word repetitions (0.68). In comparison, Random Forest and KNN demonstrate moderate performance, with ROC AUC scores ranging from 0.66 to 0.71 across different dysfluencies. Notably, SVC consistently outperforms other models in discrimination ability across dysfluency types, suggesting its effectiveness in classification tasks for stuttering events.

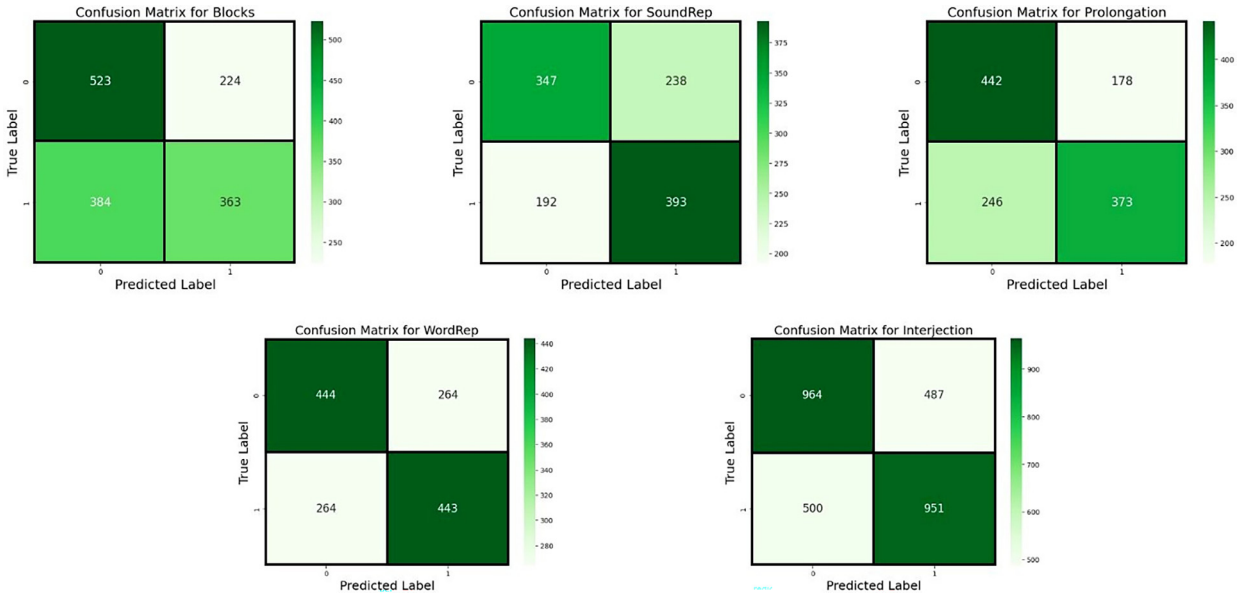

Fig. 4. Confusion Matrix of the best model for each of the dysfluencies.

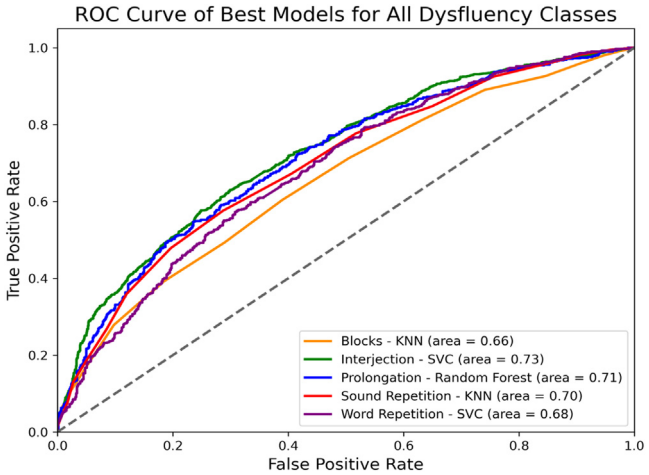

Fig. 5. ROC Curve of the best model for each of the dysfluencies.

Conclusion

In conclusion, this study presents a comprehensive approach to automated stutter detection and classification in audio files, utilizing the SEP-28k dataset and comparing various machine learning models. Our approach involved addressing various issues such as class imbalance through undersampling of the majority class. A thorough analysis of the impact of features such as MFCCs, Zero Crossing Rate, Jitter and Shimmer on the performance of the classification models was conducted.

Further improvements can be made by experimenting with other pre-trained models for transfer learning such as wav2vec 2.0 which could further improve the accuracy as well as the generalizability. Synthetic data generation and data augmentation could be applied to handle the class imbalance by expanding the dataset that retains the original characteristics of the dataset. The architecture of Convolutional Neural Network (CNN) could be made more complex by including regularization techniques suchas dropout. Experimentation could be done using various ensemble methods such as stacking, majority vote as well as soft vote which could create a more robust model.

Future scopes include exploring alternative feature sets and deep learning architectures to further enhance accuracy, and correction of dysfluencies based on speech samples which fosters accessibility in everyday communication among PWS and smart voice assistants.

Limitations

The methodology proposed has a few limitations. First, the inter-annotator disagreement in the SEP-28k dataset [14] leads to inconsistencies which introduces noise into the training data potentially affecting the accuracy of the model. Second, the limited diversity within the dataset leads to a very narrow representation of patterns in stuttering. This may cause the model to struggle with generalization when applied to different demographics or languages.

Ethics statements

Not applicable.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Ramitha V: Conceptualization, Methodology, Formal analysis, Investigation, Writing – original draft. **Rhea Chainani:** Conceptualization, Methodology, Formal analysis, Investigation, Writing – original draft. **Saharsh Mehrotra:** Conceptualization, Methodology, Formal analysis, Investigation, Writing – original draft. **Sakshi Sah:** Conceptualization, Methodology, Formal analysis, Investigation, Writing – original draft. **Smita Mahajan:** Conceptualization, Supervision, Writing – review & editing.

Data availability

Data will be made available on request.

Acknowledgments

We would like to extend our sincere gratitude to Symbiosis Institute of Technology, Symbiosis International (Deemed University), Pune, India, for providing the invaluable support and platform necessary to conduct our research work. We are thankful for providing the nurturing environment where we had the privilege to learn and explore.

References

[1] S.P. Bayerl, D. Wagner, E. Nöth, T. Bocklet, K. Riedhammer, The influence of dataset partitioning on dysfluency detection systems, in: International Conference on Text, Speech, and Dialogue, 2022, pp. 423–436. Cham: Springer International Publishing.
[2] P. Howell, S. Sackin, Automatic recognition of repetitions and prolongations in stuttered speech, in: Proc. of the first World Congress on Fluency Disorders, Vol. 2, University Press Nijmegen Nijmegen, The Netherlands, 1995, pp. 372–374.
[3] M. Wi´sniewski, W. Kuniszyk-J´ó´zkowiak, E. Smo lka, W. Suszy´nski, Automatic detection of disorders in a continuous speech with the hidden markov models approach, in: Computer Recognition Systems 2, Springer, 2007, pp. 445–453.
[4] P. Mahesha, D. Vinod, Classification of speech dysfluencies using speech parameterization techniques and multiclass svm, in: Proc. International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, Springer, 2013, pp. 298–308.
[5] S.P. Bayerl, D. Wagner, E. N¨oth, K. Riedhammer, Detecting dysfluencies in stuttering therapy using wav2vec 2.0, arXiv preprint arXiv:2204.03417.
[6] L.S. Chee, O.C. Ai, M. Hariharan, S. Yaacob, MFCC based recognition of repetitions and prolongations in stuttered speech using k-NN and LDA, in: Proc. 2009 IEEE Student Conference on Research and Development (SCOReD), IEEE, 2009, pp. 146–149.
[7] Y. Prabhu, N. Seliya, A CNN-based automated stuttering identification system, in: 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA), Nassau, Bahamas, 2022, pp. 1601–1605.
[8] Tedd Kourkounakis, Amirhossein Hajavi, Ali Etemad, arXiv preprint, 2020.
[9] M. Abubakar, M. Mujahid, K. Kanwal, S. Iqbal, M. Nabeel Asghar, A. Alaulamie, StutterNet: stuttering disfluencies detection in synthetic speech signals via mel frequency cepstral coefficients features using deep learning, in: IEEE Access, 12, 2024, pp. 99308–99320.
[10] J. P´alfy, J. Posp´ıchal, Recognition of Repetitions Using Support Vector machines, in: Signal Processing Algorithms, Architectures, Arrangements, and Applications SPA 2011, IEEE, 2011.
[11] T. Kourkounakis, A. Hajavi, A. Etemad, Detecting Multiple Speech Disfluencies Using a Deep Residual Network with Bidirectional Long Short-Term Memory, ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 2020, pp. 6089–6093, doi:10.1109/ICASSP40776.2020.9053893.
[12] B. Guitar, Stuttering: An Integrated Approach to Its Nature and Treatment, Lippincott Williams & Wilkins, 2013.
[13] M.Y. Roberts, Using Emprical Benchmarks to Assess the Effects of a Parentimplemented Language Intervention For Children With Language Impairments, Vanderbilt University, 2011.
[14] Colin Lea, Vikramjit Mitra, Aparna Joshi, Sachin Kajarekar, Jeffrey P. Bigham, 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), 2021, pp. 6798–6802. IEEE.
[15] Xinjian & Guo, Yilong Yin, Cailing Dong, Gongping Yang, Guangtong. Zhou, On the class imbalance problem, Fourth International Conference on Natural Computation, ICNC '08, 4, 2008, doi:10.1109/ICNC.2008.871.
[16] McKinney, Martin, and Jeroen Breebaart. "Features for audio and music classification." (2003).
[17] Z.K. Abdul, A.K. Al-Talabani, Mel frequency cepstral coefficient and its applications: a review, in: IEEE Access, 10, 2022, pp. 122136–122158.
[18] Shakeel A. Sheikh, Md Sahidullah, Fabrice Hirsch, Slim Ouni, Machine learning for stuttering identification: review, challenges and future directions, Neurocomputing 514 (2022) 385–402.
[19] A. Baevski, Y. Zhou, A. Mohamed, M. AuliH. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, H. Lin (Eds.), wav2vec 2.0: a framework for self-supervised learning of speech representations, Advances in Neural Information Processing Systems (2020) 460 3312 449–12.
[20] James & Bergstra, Y. Bengio, Random search for hyper-parameter optimization, J Mach Learn Res 13 (2012) 281–305.