Evaluating Edge Speaker Verification ML Pipelines: A Multi-Variable Optimization Framework

Nitish Maindoliya ∗§, Chenxin Zhang ∗§, Jaden Rhee ∗§, and Andrew J. Mason ∗‡
∗Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824
§IEEE Student Member; ‡IEEE Senior Member; {maindoli, mason}@msu.edu

***Abstract*—As edge computing proliferates, enabling real-time speaker verification on resource-constrained devices has become critical to privacy and efficiency. However, the field's trajectory toward massive deep learning (DL) embeddings (e.g., x-vectors, ECAPA-TDNN) introduces severe memory and latency overhead, rendering them nonviable for ultra-low-power micro controllers. To address this, we present a comprehensive profiling framework that systematically evaluates 225 feature-classifier permutations, comparing traditional digital signal processing (DSP) features against modern DL embeddings under strict edge constraints. Experimental profiling demonstrates that, while DL embeddings dominate raw accuracy, traditional DSP representations coupled with lightweight classical classifiers can provide the optimal design choice for the extreme edge. Specifically, an RBF-kernel support vector machine (SVM) trained on functionally pooled MFCC and zero-crossing rate (ZCR) / RMS energy features achieves one of the definitive Pareto-optimal configurations for deployment on 512 KB memory architectures.**

***Index Terms*—speaker verification, embedded systems, edge AI, MFCC, support vector machines, TinyML**

I. Introduction

Enabling real-time speech intelligence on resource-constrained edge devices is a critical challenge. Traditional multi-class speaker identification relies on cloud processing or computationally massive deep neural networks (DNNs), which exceed the strict memory and compute limits of extreme edge platforms [1]. A more scalable approach is speaker verification (SV), a localized binary decision that determines whether the incoming speech belongs to a specific registered user versus the rest of the world.

State-of-the-art SV has shifted aggressively from traditional statistical models [2], [3] to deep learning (DL) embeddings such as x-vectors [4] and ECAPA-TDNN [5]. Consequently, the current literature focuses primarily on mathematically compressing these heavy models for embedded targets [6]–[11]. However, this trend overlooks a highly efficient baseline: standard digital signal processing (DSP) features paired with classical machine learning (ML) classifiers [12]. For ultra-low-power edge cases, where even quantized DL architectures remain structurally prohibitive, these classical pipelines might be preferable due to much lower compute overhead. [13].

This work presents a comprehensive profiling framework for optimized real-time edge SV. The framework evaluates diverse combinations of feature extraction methods and lightweight classifiers under strict hardware constraints, supporting continuous audio streaming and periodic inference. By introducing a multi-objective evaluation strategy that jointly balances verification accuracy, latency, and memory footprint, the ultimate result of this work is an adaptable meta-pipeline that empowers system designers to optimize edge architectures based on their specific design goals using a tailored figure of merit (FOM).

II. Methodology

A. Data Acquisition and Preprocessing

We utilized a curated subset of the VoxCeleb2 data set [14]. To evaluate generalization, the data set was restricted to 100 unique speakers, each possessing approximately 300 seconds of audio from the same speaking session. These data were divided into strict partitions to prevent data leakage between the enrollment and testing phases.

To align with standard hardware profiles, all audio was down-sampled to a 16 kHz sample rate [14] and converted to WAV format. Furthermore, a Silero voice activity detection (VAD) pass [15]–[17] was applied to ensure that the silent segments were removed, preventing the profiling framework from wasting compute cycles on empty audio.

To avoid overfitting from a static background subset and avoid class imbalance by utilizing the entire data set, we implemented a dynamic data-generation sequence. For each trial, background audio is randomly sampled from the non-target dataset until its total duration precisely matches the target user's limited enrollment audio. This approach guaranties a mathematically balanced 1:1 training ratio while maximizing the classifier's exposure to phonetic and speaker diversity.

B. Two-Tiered Striding Strategy

To support continuous, real-time SV while managing memory constraints and limited user enrollment data, we implemented a two-tiered striding strategy (Fig. 1).

1) Segment-Level Striding: We established a 2.0-second inference window across all trials, as audio durations below one second lack sufficient phonetic coverage for reliable verification. To facilitate continuous evaluation, we applied a sliding segment stride of 1.0 seconds. This configuration ensures that the system performs an inference iteration every second while analyzing the preceding two seconds of audio, thereby preserving a broader acoustic context for each decision.

2) Frame-Level Striding: For extracting DSP features, each 2.0-second segment was analyzed using standard 25ms hamming windows with a 10ms stride [17]–[19], maintaining a 60% overlap to capture quasi-stationary vocal characteristics.

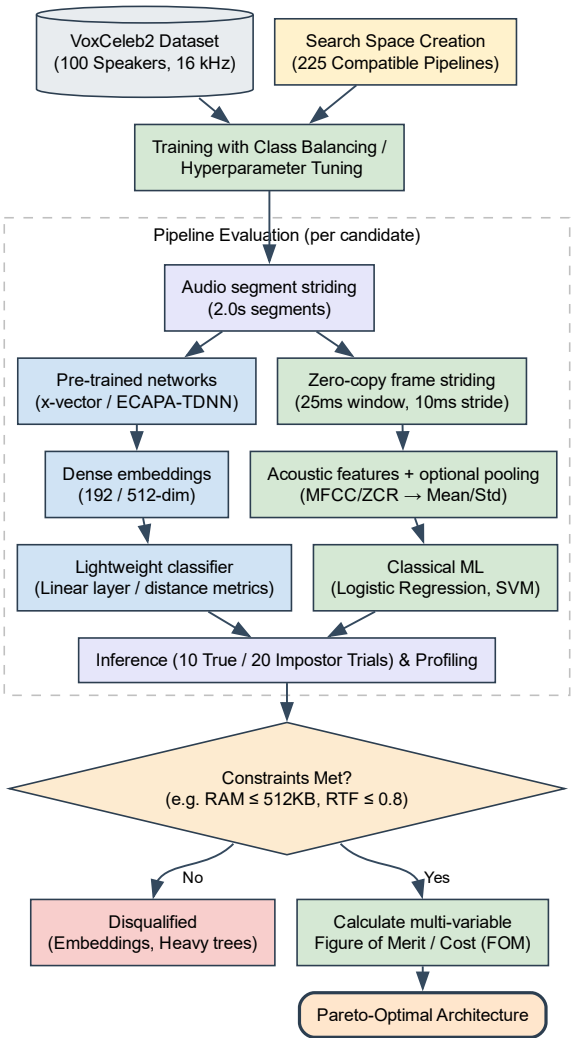

Fig. 1. Employed methodology for the systematic evaluation of SV pipelines. The framework profiles a search space of 225 feature-classifier permutations, filtering out hardware-incompatible models based on strict memory (RAM) and latency (RTF) thresholds. The remaining feasible candidates are then assessed using a multi-objective FOM to identify the Pareto-optimal architecture.

To simulate the memory constraints and pointer arithmetic of an embedded C++ environment, we implemented zero-copy memory striding using *numpy* instead of high-overhead Python libraries. In this way, we were able to pass frame pointers directly to feature extractors, eliminate redundant operations, and prevent RAM duplication. DL embeddings bypass this step, ingesting the raw 2.0 second segments directly.

C. Feature Extraction

To capture the acoustic characteristics of speech segments, we evaluated 10 distinct representations categorized under two primary extraction philosophies: DSP and DL embeddings. The explicit features and their extraction processes are defined as follows.

1) DSP and statistical pooling: Seven acoustic features were extracted at the frame level [17], encompassing both time-domain features (zero-crossing rate (ZCR) and root mean square (RMS) energy) and frequency-domain representations (mel-frequency cepstral coefficients (MFCCs), linear predictive cepstral coefficients (LPCCs), perceptual linear prediction (PLPs), pitch and dominant frequency). To convert the resulting temporal matrices (frames $\times$ features) into the fixed-length 1D arrays required by classical machine learning models, we either applied direct matrix flattening or utilized statistical functional pooling [20]. This pooling operation compressed the temporal dimension by extracting discrete statistical moments (mean and standard deviation) along the time axis, significantly reducing the overall feature footprint.

2) DL embeddings: Raw 2.0-second audio segments were passed directly through pre-trained DL architectures to extract three dense representations: ECAPA-TDNN, ECAPA2, and x-vectors. These networks manage temporal pooling internally, directly generating flat feature vectors [4], [5].

D. Classification

To determine the optimal inference engine under hardware constraints, we evaluated nine classifiers grouped into three categories.

1) Distance Metrics: Cosine similarity and Euclidean distance.

2) Classical & Statistical ML: Support vector machines (SVM) with RBF and linear kernels, random forest, logistic regression, and Gaussian mixture models (UBM- GMM) [2].

3) Neural Networks: 1D Convolutional Neural Networks (1D-CNN) and Multilayer Perceptrons (MLP).

E. Search Space Creation and Configuration Selection

Exhaustively combining up to three distinct features across nine classifier architectures yields an impractically large search space. To mitigate this combinatorial explosion, we constrained the pipeline permutations using architectural compatibility and algorithmic logic. This filtering yielded a targeted subset of 225 viable pipelines for rigorous evaluation, governed by the following rules:

1) Isolated Embeddings: DL embeddings (e.g., x-vectors, ECAPA-2) were evaluated exclusively in isolation, preventing heterogeneous concatenation with DSP features.

2) Latent Space Compatibility: Because DL embeddings are explicitly optimized for spatial separability, they were exclusively paired with distance-based metrics (e.g., cosine similarity) or linear classifiers (e.g., logistic regression).

3) Spectral Anchoring: Each DSP feature set was required to contain at least one frequency-domain representation to ensure a baseline capture of vocal tract characteristics.

4) Functional Routing: The 1D statistical functional vectors were strictly routed to classical ML algorithms (e.g., SVM, random forest) optimized for flat and dense input.

5) Matrix Routing: Unpooled temporal feature matrices were reserved for classifiers capable of frame-level processing or spatial feature extraction (1D-CNN, MLP, UBM-GMM).

F. Model Optimization and Edge-Constrained Tuning

Each statistical and DL model was hyperparameter tuned via a randomized search cross-validation (*RandomizedSearchCV*) approach. Crucially, to preserve the framework's viability for memory-constrained embedded environments, the hyperparameter search grids were intentionally bounded to prevent the models from becoming computationally heavy.

G. Profiling and Evaluation

To systematically benchmark the pipelines, models were trained on balanced 30-to-60-second subsets of Target and World audio. The verification phase consisted of 10 Target and 20 World trials, evaluating 5.0 seconds of audio per trial. To ensure statistical robustness and mitigate speaker bias, the final performance metrics were averaged across 20 independent iterations using randomized target speakers.

Because raw accuracy is dependent on a specific decision threshold and can be skewed by trial imbalances, we utilized the equal error rate (EER) as our primary threshold-independent metric [21]. The EER represents the specific operating point where the system's false acceptance rate (FAR) and false rejection rate (FRR) are strictly equivalent. These individual error rates are computed as follows.

$$FAR = \frac{FP}{FP+TN}, \quad FRR = \frac{FN}{FN+TP} \tag{1}$$

By sweeping the decision threshold τ across the continuous prediction scores produced during the trials, the EER is determined by interpolating the point where $FAR(\tau) = FRR(\tau)$, providing a balanced measure of discriminative capacity.

To accurately simulate embedded hardware constraints, execution was strictly limited to a single CPU core without any parallelism or vectorization. Computational latency was profiled by tracking total CPU cycles (for both feature extraction and inference) to enable precise real-time factor (RTF) scaling across diverse microcontroller clock frequencies. The peak theoretical memory footprint per trial was quantified by aggregating the byte sizes of the incoming audio buffer, intermediate extraction matrices, the final feature vector, static model parameters, and runtime state variables.

III. Results and Analysis

A. Constraint Formulation

Edge deployment introduces rigid hardware limitations that must be satisfied before evaluating biometric performance. Consequently, selection is framed as a constraint satisfaction problem (CSP). Let M denote the complete set of evaluated feature-classifier permutations. The feasible set F is defined as:

$$F = \{m \in M \mid RAM(m) \leq t_{RAM} \text{ and } RTF(m) \leq t_{RTF}\} \tag{2}$$

where $RAM(m)$ represents the peak memory footprint, and $RTF(m)$ represents the real-time factor. For the target platform (e.g., Arduino Nano 33 BLE), the maximum allowable

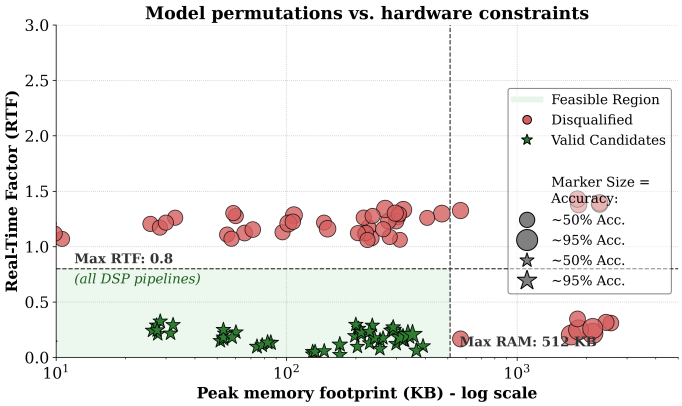

Fig. 2. The green shaded area (lower left) defines the feasible region, bounded by a Max RTF of 0.8 and a Max RAM of 512 KB. 118 valid candidates (green stars) are all DSP-based pipelines, while disqualified pipelines (red circles) fail to meet one or both constraints. Marker size indicates accuracy (1 - EER).

SRAM usage was constrained to 512 KB, while the latency constraint was defined as:

$$RTF = \frac{T_f + T_i}{T_{audio}} \tag{3}$$

where T_f is the feature extraction time, T_i is the classifier inference time, and T_{audio} is the 2.0-second inference window. To ensure stable real-time operation without buffer overflows, the constraint $RTF \leq 0.8$ was strictly enforced.

Architectures that violated either threshold were disqualified. Notably, all DL embedding approaches exceeded the available memory budget due to their massive parameter counts. As a result, only 118 DSP pipelines paired with classical ML classifiers or lightweight neural networks remained within the feasible region.

B. Figure of Merit Optimization

Although hardware constraint analysis isolates feasible models, it does not determine the optimal architecture. To quantitatively evaluate the trade-offs between biometric accuracy and computational efficiency, we formulate a multi-objective FOM. [22]. Each metric is first standardized using min-max normalization:

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{4}$$

The overall FOM is then defined as a minimization problem:

$$FOM = w_1 \cdot EER_{norm} + w_2 \cdot RTF_{norm} + w_3 \cdot RAM_{norm} \tag{5}$$

where the assigned weights satisfy $w_1 + w_2 + w_3 = 1$. To fully explore the capability spectrum of our model permutations, we conducted the FOM analysis in three distinct weighting scenarios (illustrated in Figure 3):

1) ***Balanced*** *($w_1 = 0.33, w_2 = 0.33, w_3 = 0.34$):* When equal priority is given to accuracy and system resources, highly compressed feature representations paired with lightweight non-linear classifiers prove most effective among the feasible candidates. The (MFCC, RMS) + SVM-RBF

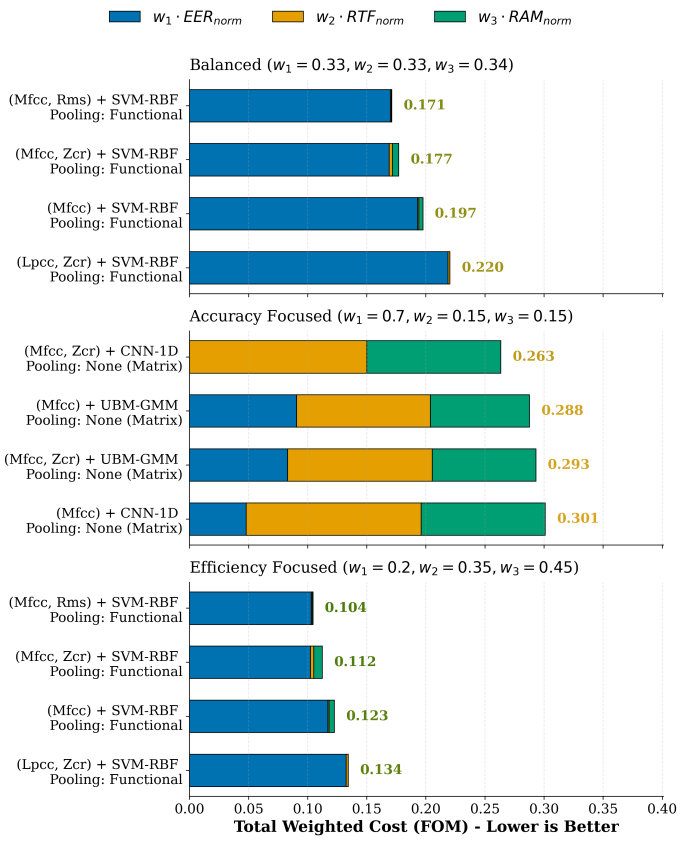

Fig. 3. Stacked bar charts comparing top 4 model performance across three weighting scenarios: Balanced, Accuracy Focused, and Efficiency Focused. The total weighted cost FOM is calculated using normalized values for EER, RTF, and RAM. Lower cost indicates a superior candidate.

pipeline with functional pooling emerged as the optimal architecture, achieving an FOM of 0.171. While its absolute EER of 21.4% lags behind the most accurate constrained models by 11.2%, it compensates with exceptional efficiency consuming only 52.4 KB of RAM and maintaining an RTF of 0.01.

*2) **Accuracy Focused** ($w_1 = 0.70, w_2 = 0.15, w_3 = 0.15$):* By heavily weighting biometric performance ($w_1 = 0.70$), the most computationally dense of the surviving architectures surface as the top performers. In this regime, the (MFCC, ZCR) + CNN-1D and UBM-GMM pipelines achieve the lowest FOMs, with the CNN-1D pipeline achieving the lowest constrained EER of 10.2%. However, this accuracy incurs a resource penalty. As seen in the stacked charts, the normalized RTF and RAM costs inflate their total FOM to 0.263–0.301. Compared to the balanced optimal model, this configuration demands a $7.72\times$ increase in peak memory (404.6 KB) and a $53.4\times$ increase in computational latency. Consequently, these models are viable for higher-tier edge devices.

*3) **Efficiency Focused** ($w_1 = 0.20, w_2 = 0.35, w_3 = 0.45$):* For ultra-low-power, battery-operated systems where memory and latency are critical bottlenecks, the (MFCC, RMS) + SVM-RBF pipeline remains the definitive choice. By shifting the weights away from accuracy, its combined cost drops even further to an FOM of 0.104. The fact that this specific architecture dominates both the balanced and efficiency-focused scenarios highlights its exceptional robustness as a Pareto-optimal solution.

C. Constrained vs. Unconstrained Architectures

The FOM analysis delineates the optimal models *within* strict hardware boundaries ($RAM \leq 512$ KB); however, to provide an equitable analysis of the current SV landscape, we must contextualize these constrained results against state-of-the-art DL pipelines. During initial profiling, DL embedding architectures, notably ECAPA variants [5], [8] and standard x-vectors [4], were evaluated but disqualified due to constraints violations. These unconstrained models achieved exceptional biometric performance, recording an EER of 1.28%—a staggering $7.96\times$ relative improvement over our most accurate constrained pipeline (the 10.2% EER of the CNN-1D).

To bridge the 20.12% absolute gap in EER between the optimal edge SVM pipeline and the ECAPA-2 model, the memory footprint must expand from 52.4 KB to approximately 80–150 MB. This equates to an increase of roughly $2,500\times$ in the memory requirement for a reduction of $15.7\times$ in the error rate. This extreme asymmetry confirms that while deep embeddings dominate cloud-based verification, their parameter density renders them fundamentally incompatible with TinyML paradigms without aggressive—and potentially accuracy-destroying—quantization or pruning [6].

D. Feature Dynamics

We note that while MFCCs drove the optimal baseline, alternative frequency-domain representations demonstrated significant viability. Low-power systems could strategically leverage LPCCs or PLPs [23] to further minimize extraction overhead based on the specific presence or absence of floating-point units (FPUs).

IV. Conclusion

This work presented a constraint-driven framework for systematically evaluating biometric pipelines against rigid hardware limitations. By formalizing model selection as a constraint satisfaction problem and applying a multi-objective FOM, this framework empowers system designers to objectively eliminate prohibitive architectures and precisely balance EER, RTF and peak RAM.

Extensive evaluation across 225 pipelines established that while deep learning models such as ECAPA-2 provide unmatched accuracy, their exponential resource costs disqualify them for edge use. Instead, our analysis identifies an RBF-kernel SVM trained on functionally pooled MFCC and ZCR/RMS energy features as a Pareto-optimal solution. Ultimately, this work proves that highly optimized classical pipelines, paired with intelligent feature compression, remain indispensable for the next generation of embedded audio intelligence.

REFERENCES

[1] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 2014, pp. 10–14.

[2] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digit. Signal Process.*, vol. 10, no. 1, p. 19–41, Jan. 2000. [Online]. Available: https://doi.org/10.1006/dspr.1999.0361

[3] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[4] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5329–5333.

[5] B. Desplanques, J. Thienpondt, and K. Demuynck, "Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification," in *Interspeech 2020*. ISCA, Oct. 2020, p. 3830–3834. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-2650

[6] J. Lin, W.-M. Chen, Y. Lin, J. Cohn, C. Gan, and S. Han, "Mcunet: Tiny deep learning on iot devices," 2020. [Online]. Available: https://arxiv.org/abs/2007.10319

[7] R. David, J. Duke, A. Jain, V. J. Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, S. Regev, R. Rhodes, T. Wang, and P. Warden, "Tensorflow lite micro: Embedded machine learning on tinyml systems," *CoRR*, vol. abs/2010.08678, 2020. [Online]. Available: https://arxiv.org/abs/2010.08678

[8] J. Thienpondt and K. Demuynck, "Ecapa2: A hybrid neural network architecture and training strategy for robust speaker embeddings," 2024. [Online]. Available: https://arxiv.org/abs/2401.08342

[9] T. Iatariene, A. Guérin, and R. Serizel, "Towards low-latency tracking of multiple speakers with short-context speaker embeddings," 2025. [Online]. Available: https://arxiv.org/abs/2508.14115

[10] Y. Li, Z. Jiang, Q. Huang, W. Cao, and J. Li, "Lightweight speaker verification using transformation module with feature partition and fusion," 2023. [Online]. Available: https://arxiv.org/abs/2312.03324

[11] J. Balian, R. Tavarone, M. Poumeyrol, and A. Coucke, "Small footprint text-independent speaker verification for embedded systems," 2021. [Online]. Available: https://arxiv.org/abs/2011.01709

[12] A. Kumar, S. Goyal, and M. Varma, "Resource-efficient machine learning in 2 KB RAM for the internet of things," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 06–11 Aug 2017, pp. 1935–1944. [Online]. Available: https://proceedings.mlr.press/v70/kumar17a.html

[13] A. Nassif, I. Shahin, M. Lataifeh, A. Elnagar, and N. Nemmour, "Empirical comparison between deep and classical classifiers for speaker verification in emotional talking environments," *Information*, vol. 13, p. 456, 09 2022.

[14] J. S. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep Speaker Recognition," in *Interspeech 2018*, 2018, pp. 1086–1090.

[15] J. Ramírez, J. C. Segura, C. Benítez, Á. de la Torre, and A. Rubio, "Efficient voice activity detection algorithms using long-term speech information," *Speech Communication*, vol. 42, no. 3, pp. 271–287, 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167639303001201

[16] S. Team, "Silero VAD: pre-trained enterprise-grade Voice Activity Detector (VAD), Number Detector and Language Classifier," 2024.

[17] L. R. Rabiner and R. W. Schafer, *Theory and Applications of Digital Speech Processing*. Pearson, 2011.

[18] D. T. Toledano, M. P. Fernández-Gallego, and A. Lozano-Diez, "Multi-resolution speech analysis for automatic speech recognition using deep neural networks: Experiments on timit," *PLOS ONE*, vol. 13, no. 10, pp. 1–24, 10 2018. [Online]. Available: https://doi.org/10.1371/journal.pone.0205355

[19] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, 1980.

[20] B. Schuller, S. Steidl, A. Batliner, A. Vinciarelli, K. Scherer, F. Ringeval, M. Chetouani, F. Weninger, F. Eyben, E. Marchi, M. Mortillaro, H. Salamin, A. Polychroniou, F. Valente, and S. Kim, "The INTERSPEECH 2013 computational paralinguistics challenge: social signals, conflict, emotion, autism," in *Interspeech 2013*, 2013, pp. 148–152.

[21] A. F. Martin, G. R. Doddington, T. M. Kamm, M. Ordowski, and M. A. Przybocki, "The det curve in assessment of detection task performance," in *EUROSPEECH*, 1997. [Online]. Available: https://api.semanticscholar.org/CorpusID:9497630

[22] R. Marler and J. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and Multidisciplinary Optimization*, vol. 26, pp. 369–395, 04 2004.

[23] P. Suba and B. Bharathi, "Analysing the performance of speaker identification task using different short term and long term features," in *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies*, 2014, pp. 1451–1456.