ISNNI: 3007-3391 HEC Category: Y

Journal of Computers and Intelligent Systems
Vol. 4, No. 2, 2026, pages 82 - 98,
Journal Homepage:
https://journals.iub.edu.pk/index.php/JCIS/

Preference-Based Asynchronous Speech Enhancement using Deep Learning with Feature Unlearning for Audio Privacy Preservation

Yusra Irshad Ali[1], Riaz Ul Amin1 , Javed Rashid[1], Muhammad Shoaib Saleem[2], Muhammad Sajjad Saleem[1]
[1]Department of Computer Science, University of Okara, Okara, Punjab 56310, Pakistan.
[2]Department of Mathematics, University of Okara, Okara, Punjab 56310, Pakistan.

ARTICLE INFO

Article History:

Received:	March	01, 2026
Revised:	April	20, 2026
Accepted:	May	25, 2026
Available Online:	June	30, 2026

Keywords:

Speech Enhancement
Federated Learning
Feature Unlearning
Transformer Networks
Privacy-Preserving Edge Computing

Classification Codes:

Funding:

No Funding.

ABSTRACT

Speech enhancement systems that rely on centralized training put user audio at risk: raw recordings have to leave the device and sit on a server somewhere before a model can learn from them. This paper works through an alternative built around a compact Transformer, trained with federated learning so audio never leaves the client, and paired with a feature-unlearning step so a user can later have their contribution struck from the global model. We target edge hardware specifically – a Raspberry Pi acts as the client, Google Colab hosts the server side, and Flask plus Ngrok tunneling handle the back-and-forth between them. The unlearning mechanism supports both pre-training and post-training removal requests, which matters for compliance regimes like GDPR that give users an ongoing right to withdraw consent rather than a one-time opt-in. Tests on the target hardware show the approach holds up on speech quality while keeping the extra computational burden from federation and unlearning within what a small edge device can absorb.

Corresponding Author's Email: riazulamin@gmail.com

Citation:

1. Introduction

When it comes to communication technology, the ability to hear and understand is no longer a novelty, whether you're video calling or using a hearing aid or rely on a voice assistant to understand what you were told. The vast majority of that load was handled by signal-processing techniques; in the past three years, though, deep learning and, specifically, Transformer architectures have come to the fore, as they are able to capture long-range dependencies in audio that older methods fail to do so. There's a price tag to that change, however. But using these models well typically requires a large

number of user speech samples, their transfer to a remote server, and their storage there – which is a tough sell in healthcare, personal communication, or any other context where the recordings themselves are sensitive [1]. The models are also heavy enough that running them on a phone, hearing aid, or single-board computer is its own separate problem, apart from the privacy one. This paper works on both problems at once. We build a Transformer-based enhancement model deliberately kept small enough for edge hardware, and train it with federated learning so a Raspberry Pi client can improve the global model without ever handing its audio to the server. Google Colab plays the server role, with Flask APIs and an Ngrok tunnel connecting the two sides in real time – model updates actually happen on physical edge hardware during evaluation, not just in simulation [2]. On top of that, we add a feature-unlearning step: a user can ask that their contribution to the shared model be undone, whether that request comes before or after training, which is the kind of ongoing revocation right that GDPR and similar frameworks are increasingly built around rather than a one-time consent checkbox. Put together, the three pieces – a Transformer for enhancement quality, federated training so audio never leaves the device, and unlearning so a user can revoke their influence on the model – give the system something a purely centralized, edge-agnostic pipeline can't offer.

RESEARCH OBJECTIVES

The first goal of this research is to design a powerful multimodal transformer-based architecture for speech enhancement, which is capable of jointly exploiting audio and visual information. Specifically, the aim of this study is to:

- Design and implement a light weight deep learning model using log Mel-spectrograms from audio signals and lip movement features from silent videos to improve the speech quality.
- Use Transformer encoder mechanism to combine temporal sequences of multimodal data to enhance speech intelligibility and quality in noisy conditions.
- Accurately extract lip landmarks by MediaPipe FaceMesh and with audio features to create coherent multimodal input for training.
- Train and validate the proposed model using noisy speech and silent video samples with good synchronization, and test using objective measures like PESQ (Perceptual Evaluation of Speech Quality), STOI (Short-Time Objective Intelligibility), and SNR (Signal-to-Noise Ratio).
- Investigate the possibility of real-time speech enhancement on edge devices, without sacrificing data privacy, in a federated environment.

Study Contributions

Let's take a look at the main contributors to this study:

- A light-weight multimodal speech enhancement framework with both audio and lip motion features.
- Support for federated learning for privacy-preserving distributed training, without sharing raw data.
- A feature unlearning mechanism to selectively remove user-contributed information is introduced.
- It provides a ready to be deployed architecture on the edge device (such as Raspberry Pi) with the use of Flask API and Ngrok tunneling. PESQ, STOI and SNR metrics are used in the noisy environment to carry out a detailed evaluation.

2. Related Work

Neuro-fuzzy models combine the pattern-matching capabilities of neural networks with the rule-based reasoning of fuzzy logic, and one of the most popular models is undoubtedly ANFIS (Adaptive-Network-Based Fuzzy Inference System) which integrates the two to learn and infer complex, non-linear functions [3]. The fusion is effective even in the absence of noise reduction: The representational power of deep learning and the robustness of fuzzy logic for processing uncertainty has led to better predictions in classification, natural language processing, and more [4]. A review from 2015 to 2020 which listed 105 studies of deep neuro-fuzzy systems is available [5] and details the structural variations that these hybrids can be found in and the optimization techniques that can be applied for their tuning, though the selection of studies included may have biased the results and thus weaken the generalizations that can be drawn from it.

In addition to neuro-fuzzy hybrids, deep learning has been directly used in speech recognition and enhancement. In one study, LSTM was used in combination with CTC for railway reservation system and it was found that audio quality directly affects the recognition accuracy [6] but it did not investigate how well LSTM-CTC works or adapts to real-world applications. To enhance the overall recognition accuracy in speech recognition, another paper proposed a hybrid Deep neural network and Discriminant Fuzzy Logic (DF) model with 8.31%, 9.71% and 10.25% improvement over MFCC-CNN, CSVM and Deep autoencoder respectively [7]. The improvements are significant, but the paper does not consider computational complexity, training time, or model efficiency—which play important roles when deploying on the edge. In a third approach, a Deep Neural Network is used in the cepstral domain for identification of periodicity in the frequency spectrum based on a source-filter paradigm known from human speech production, and a reduction in noise is reported [8]. Adding a DNN in this manner makes the system more complicated, and the paper does not cover all the noise types a deployed system would see; the quality and quantity of the training data turn out to be very important to the degree of generalization.

In another research authors proposed a Quantum Fuzzy Neural Network (QFNN), a multimodal fusion technique that exploits a multitask learning algorithm with a Seq2Seq structure and integrates Classical and Quantum Neural Networks (QNN) with fuzzy logic. Using complex numbers in the Fuzzifier to capture sentiment and sarcasm features, and QNN in the Defuzzifier to obtain the prediction, the authors show that QFNN can outperform several recent methods in sarcasm detection [9]. Despite the shortcomings in the approaches discussed above, most of these studies [3]–[8] confirm that neuro-fuzzy models can reduce noise in voice data and outperform traditional noise-reduction methods. The QFNN work [9] is not itself a noise-reduction method – it targets multimodal sarcasm and sentiment detection – but it shows that fuzzy logic combined with neural architectures generalizes well beyond speech to other multimodal signal-processing tasks. That said, the performance of these neuro-fuzzy approaches can likely be improved further by integrating them with deep learning techniques, and their efficacy still needs to be evaluated on genuinely multi-modal data rather than a single modality at a time.

Unlearning for Speech and Audio Models

Feature unlearning is central to this paper's privacy claims, so it's worth looking at how the field has treated it specifically for speech and audio, rather than for text or images where most of the early unlearning work originated. Cheng and Amiri frame speech unlearning as its own research problem and split it into two tasks: sample unlearning, which removes a single recording's influence, and class unlearning, which removes an entire speaker's data, while trying to keep performance on everything else intact [13]. They evaluate this on three backbones: Whisper, Wav2Vec 2.0, and HuBERT; and they find that techniques that work well for other types of unlearning don't transfer seamlessly – speech is much more difficult than, say, image classification to verify that the model has forgotten.

Mason-Williams et al. make a related point: audio has largely been left out of machine unlearning

research even though it's one of the most common data modalities in production systems. Their analysis finds that existing unlearning methods struggle with the most common real-world request – removing a single item – and they introduce a prune-and-regrow paradigm that combines cosine pruning with post-optimal pruning to unlearn more thoroughly [14]. This is directly relevant to the weight-pruning approach used in Algorithm 1 of this paper: their results suggest pruning alone is a reasonable unlearning mechanism for audio, but that how the pruning is done (and whether pruned weights are allowed to regrow) changes how completely a model actually forgets. Xiao et al. push on a related gap, showing that deleting data changes a sparse model's pruned topology itself, not just its weights, and propose an "un-pruning" step that can be layered on top of any existing unlearning algorithm to correct for that [15]. Kim et al. address the unlearning problem specifically for zero-shot TTS, where it's not just that the information is retained, but that one could ask the model to speak in a voice that it was supposed to have forgotten [16]. The four papers collectively argue that unlearning for speech and audio is a non-trivial problem and that pruning-based methods are not only an active and ongoing field of research but also a problem yet to be solved.

Federated Learning for Speech Processing

The second key component of this framework is federated learning, and there has been initial research using federated learning for speech tasks outside its typical FL context. Kan et al. fuse parameter-efficient transfer learning and federated training for ASR, requiring only a small adapter to be communicated per round, rather than the entire acoustic model, thus reducing the communication cost that is crucial for bandwidth-limited edge clients such as the Raspberry Pi employed in this paper [17]. Similar communication bottlenecks for personalized federated speech-to-text models are discussed by Du et al. [18] and even more so by Pelikan et al. [19] who add differential privacy to federated ASR training, demonstrating that formal privacy guarantees can be layered on top of FL without compromising recognition accuracy, albeit with some expense to convergence speed. A separate line of work applies regularized federated learning to domains where the data is a limiting factor and very sensitive, such as dysarthric and elderly speech recognition [20], and demonstrates that, with careful regularization, a global model can be trained that generalizes across clients with very different speech patterns without any data being centralized. None of these systems feature federated training with feature unlearning like this paper does, but they all show it is possible to train speech federately at the communication and privacy budgets the edge devices actually have.

Lightweight Transformers for Edge Speech Enhancement

On the enhancement side, several 2024–2025 papers tackle the same core tension this paper does: Transformers enhance speech well but are expensive to run on constrained hardware. Lin et al. propose MUSE, a U-Net-based enhancement network built around a Multi-path Enhanced Taylor Transformer block, and report competitive results on VoiceBank+DEMAND with only 0.51M parameters [22] – a useful reference point for how small a Transformer-style enhancer can get while still performing well. Ojha et al. take a different route, adding reverse-attention gates to a U-Net specifically for edge deployment, and report a 6.24% improvement in Word Error Rate and a 0.64-point PESQ improvement over unenhanced audio [21]. Both papers, along with the diffusion-transformer-based DiTSE model [10] and the real-time enhancement framework of Kim et al. [11], confirm that lightweight, edge-capable enhancement is an active research direction. What none of them address is privacy: none of these systems train federated, and none support removing a user's data from the trained model after the fact. That combination – a lightweight Transformer, trained federated, with feature unlearning built in – is where this paper's contribution sits relative to the recent literature.

3. SYSTEM DESIGN

This section walks through how the pieces fit together: a compact Transformer for the enhancement itself, wrapped in a federated learning setup so training happens without moving anyone's audio off their device, with feature unlearning layered on top so that consent can be withdrawn after the fact. The whole thing is built to run on hardware most labs don't consider real edge deployment, not just a scaled-down laptop. Fig. 1 sketches the cloud-supported deployment model this section describes.

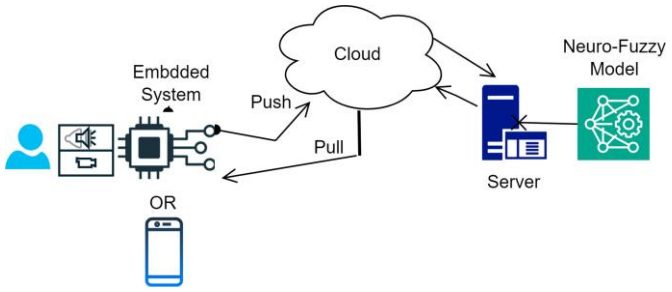

Fig. 1. Cloud-supported Deep Neuro-fuzzy Deployment Model

1) The system splits into three pieces that talk to each other rather than one monolithic pipeline:
   - The Raspberry Pi is where the actual audio lives – it preprocesses each sample locally, runs the on-device training step, and evaluates its own local model before anything gets sent anywhere.
   - Google Colab plays server: it holds the current global model, folds in updates as clients report back, runs the global-level evaluation, and talks to clients over a Flask API.
   - An Ngrok tunnel bridges the two without needing a public IP or open port on either end, so updates can flow in real time even though the client sits behind a home network. Federated learning is what actually happens over that tunnel – clients train locally and only ever exchange model weights, never raw audio or video.

2) Data Preprocessing

CLIENT-SIDE AUDIO PROCESSING

Each audio sample gets three passes on the client before it's ready for the model:

- **Audio Features:** log-Mel spectrograms are pulled from the noisy speech (*mixed.wav) alongside the matching clean target (*clean.wav).
- **Normalization:** spectrograms are normalized.
- on a per-sample basis to improve convergence and generalization.
- **Sequence Preparation:** Audio sequences are padded or truncated to a fixed length and converted into embeddings suitable for Transformer input.

LIGHTWEIGHT TRANSFORMER ARCHITECTURE

The core model for speech enhancement is a lightweight Transformer designed with computational efficiency in mind [10]. It consists of the following components:

a) Input Embedding Layer
- Converts each frame of the audio spectrogram into a dense vector representation.
- Embedding dimension: 128
- Positional encoding is added to retain sequence information.

b) Multi-Head Self-Attention Layer
- Composed of 4 attention heads, each operating in parallel.
- Key, Query, and Value vectors are projected with a reduced dimensionality (128 / 4 = 32 per head).
- The attention outputs from all heads are concatenated and linearly projected back to 128 dimensions.

c) Feed-Forward Network (FFN)
- Two linear layers with a ReLU activation in between.
- Hidden dimension: 256 → 128
- Dropout is applied after each layer to prevent overfitting.

d) Residual Connections and Layer Normalization
- Applied after both the multi-head attention and FFN blocks to improve gradient flow and training stability.

e) Output Projection
- Final dense layer maps the output sequence back to the original spectrogram dimension.
- An inverse Mel transformation and overlap-add procedure are used to reconstruct the enhanced audio waveform.

DNFS Summary: The low cost approach makes the model suitable for constrained resource environments. The evaluation translates the fuzzy inputs into rule-based outputs, then a defuzzification step converts those rules into the values used for learning and setting parameter weights. Training runs iteratively toward optimized parameters, and once initial optimization is done, stacking multiple layers of fuzzy neurons gets you hierarchical learning – the same idea behind deep architectures generally, just applied through fuzzy rather than purely numeric layers. Capturing complex patterns this way depends on how many layers of abstraction the framework supports. Once a model trained this way checks out against a held-out validation and test set, it's ready for real-time deployment.

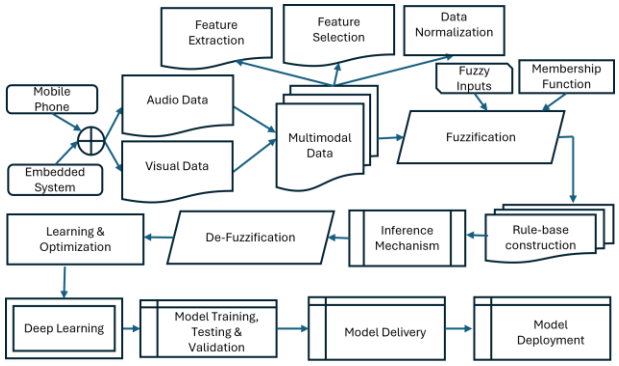

Fig. 2. Workflow of Deep Neuro-Fuzzy System

Deep neuro-fuzzy models have been tested in plenty of studies, but how well they hold up in noisy, multi-talker settings on constrained hardware is a separate question from how well they

do in a clean benchmark. One deployment option is a phone with its own camera and microphone, syncing multimodal data to the cloud so the neuro-fuzzy model can keep learning and push updates back down (Fig. 2). Phones are themselves resource-constrained, though – battery, processing, memory all cap what's realistic. A dedicated embedded system sidesteps some of that by carrying its own resources and pulling learning updates from the cloud as needed, with the model itself still living server-side. Either way, the client needs a steady connection and the communication cost from constantly shuttling data to the cloud adds up. Pushing some preprocessing out to the edge – so only the already-processed data has to travel – cuts that cost, and folding in federated learning so part of the workload stays local as well pushes it down further. Making that work reliably means the energy-aware, context-aware algorithms running at the edge have to actually be evaluated for how well they hold up under those constraints, not just assumed to.

A. Transfer Learning Model for Speech Enhancement

Transfer learning shows up all over deep learning, speech included, but it hasn't translated well to audio speech enhancement specifically. The basic move is familiar: train on a large general-purpose dataset, then adapt that model to a smaller, more specific target domain, usually by fine-tuning or freezing certain layers on top of feature extractors like spectrograms or MFCCs. Enhancement turns out to be a rough fit for this. Real-world noise looks nothing like the synthetic noise most pre-training datasets use, so whatever the pre-trained model learned about noise doesn't generalize the way you'd hope. In addition, the fine-tuning a large model on a small target dataset can result in overfitting, and enhancement is a less demanding task than, for example, speech recognition or speaker identification – small examples that recognize words don't necessarily transfer to reconstructing the clean audio signal.

A second difficulty lies there, and this is due to feature mismatch: a pre-trained model is optimized for the feature set used in its training, and these are often not the same as the features used in the target domain, which causes a drop in performance. In real recordings, however, the clean target signals are not always available, nor are they even always aligned with the target signals, so noisy-label mismatches add to the mix. All this is on the backdrop of hardware constraints and latencies – for a Raspberry Pi the model has to be light regardless of how well it fits. That's leading to a shift to domain adaptation, meta-learning and self-supervised learning as these approaches learn features that are useful in multiple domains or they adapt to a new noise environment without re-training. Whereas, for cases where transfer learning is applicable, multimodal signals such as lip movement are helpful as well. alone comes up short. None of this rules transfer learning out – it's still a useful starting point – but applying it directly to ASE rarely works without real modification. Fig. 3 sketches the general transfer-learning pipeline this discussion has been describing.

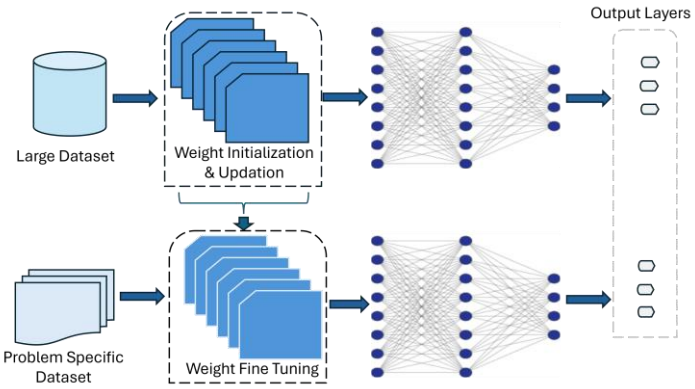

Fig. 3. Transfer Model based Speech Enhancement Model

B. Transformer-Based Speech Enhancement Model

The system runs as a client-server setup with federated learning tying the two sides together [11]: Google Colab handles the server role, and a Raspberry Pi plays client. On the server side, that means setting up the global model and orchestrating training – folding in updates as they come from clients, and running the objective evaluation (PESQ, STOI, SNR) once results come back. The Raspberry Pi client does the opposite job: it preprocesses whatever multimodal data it has locally, audio and silent-video lip movement both, trains on that private data, and sends back only the resulting model updates. The raw data itself never leaves the device. Flask-based RESTful APIs handle the client-server communication, and Ngrok tunnels the Colab server out to the client securely, so both sides can talk bidirectionally without needing a public IP or manual port forwarding. Taken together, that gives a training and inference pipeline that runs in real time on low-resource edge hardware without the raw audio ever touching the server. The rest of this section breaks the system down by its key components:

- **Federated Learning (FL):** Enables distributed training of the model across edge devices without transferring raw audio data, thereby ensuring user privacy and data security.
- **Deep Neuro-Fuzzy Network:** Employs an adaptive and lightweight learning mechanism for effective noise suppres-sion in diverse and dynamic acoustic environments.
- **Client-Server Architecture:** Utilizes a practical implementation framework combining Flask and Ngrok, with the Raspberry Pi serving as the client node and Google Colab acting as the cloud-based server.
- **Unlearning Capability:** Incorporates data unlearning mechanisms to allow selective forgetting of user data, thereby supporting compliance with emerging data privacy regulations such as GDPR and facilitating user-controlled data revocation. The complete architecture of the Transformer-based speech enhancement model, including the fusion and attention layers, is shown in Fig. 4.

Why Asynchronous Federated Learning: In the traditional synchronous federated learning model, the server has to wait for all (or a certain proportion) of the selected clients to finish local training and return the updated model to the server for the next iteration of the global model aggregation. This sync barrier causes the round to stall if the slowest client is a heterogeneous edge

device like a Raspberry Pi unit or if a client disconnects from the network because of an unstable network tunnel (e.g. Ngrok session expired). The result is a slowest device (straggler) issue because "slowest device" is a device that has the lowest connection speed and thus the longest sync time. (The straggler problem) The asynchronous design used in this work enables the server to summarize and broadcast the updated weights as soon as a client has reported back, instead of waiting for the other members of the group. This decreases the idle time for faster clients and enhances the robustness to intermittent connections, and it also aligns the push/pull update pattern of the client/server architecture described earlier using Flask. The downside of this is that gradients may be stale when a client updates on an earlier version of the global model, but the pruning-based unlearning step in Algorithm 2 helps address this by unlearning (re-fine-tuning) the weights that are affected after each aggregation.

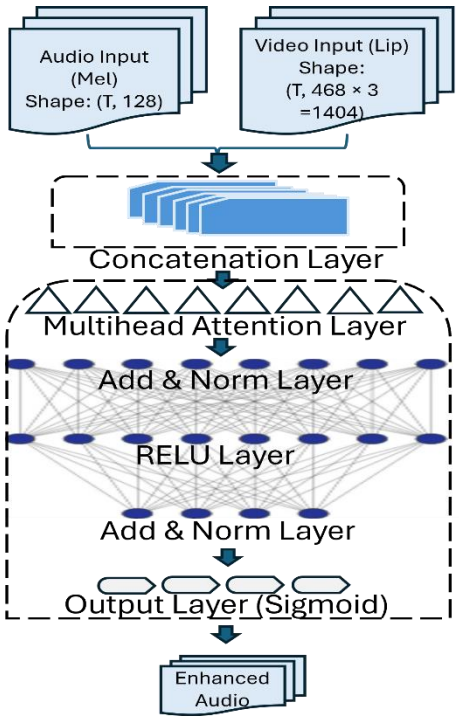

Fig. 4. Transformer-Based Speech Enhancement Model

Algorithm 1 Feature Unlearning via Weight Pruning

Input:

Model weights **W**

Pruning threshold τ

Features to unlearn F

Output: Pruned model weights $\mathbf{W}_{\text{pruned}}$

Identify Weights:

Identify weights W_F associated with features F

Prune Weights:

Prune weights below threshold:

W_F ← W_F · I(|W_F| > τ)

Fine-Tune Model:

Fine-tune W on the remaining features to recover performance

Return:

W_pruned ← W

The Transformer-based architecture for speech enhancement leverages two inputs:

- **Audio Features:** Extracted from the noisy audio signal.
- **Lip Motion Features:** Extracted from the corresponding silent video.

The output is a clean audio signal, $\hat{y}$, approximating the ground truth clean audio y.

Model Components

1 Audio Feature Extraction:

$$x_a = \text{Conv1D}\left(x_{\text{audio}}\right) \rightarrow \text{MaxPool1D}(\cdot) \rightarrow \text{BiLSTM}(\cdot)$$

2 Lip Motion Feature Extraction :

$$x_l = \text{TimeDistributed}\left(\text{Flatten}\left(x_{\text{lips}}\right)\right) \rightarrow \text{BiLSTM}(\cdot)$$

where $x_{\text{lips}} \in \mathbf{R}^{F \times V \times 2}$ represents the lip motion features, F is the number of frames, and V is the number of landmarks.

3 Fusion Layer: The concatenated features from audio and lip motion are passed through dense layers:

$$x_{\text{concat}} = \text{Concat}(x_a, x_l)$$

$$h = \text{Dense}\left(x_{\text{concat}}\right) \rightarrow \text{Dropout}(\cdot) \rightarrow \text{Dense}(\cdot)$$

4 Output Layer: The model predicts the clean audio signal:

$$\hat{y} = \text{Dense}(h)$$

Unlearning Architecture

The unlearning mechanism is designed to remove the contribution of lip motion features while preserving the learned audio features [12].

Weight Pruning for Lip Motion Features: Let W_l denote the weights associated with the lip motion feature extraction layers. Unlearning is achieved by pruning these weights:

$$W_l \leftarrow W_l \odot M$$

where $M \in \{0, 1\}^{\text{shape}(W_l)}$ is a binary mask defined as:

$$M \in \{0,1\}^{\text{shape}(W_l)}$$

$$M_{i,j} = \begin{cases} 0, & \text{if weight } W_{l,i,j} \text{ is pruned,} \\ 1, & \text{otherwise.} \end{cases}$$

Modified Forward Pass: After pruning, the lip motion features are effectively ignored during the forward pass:

$$x_{\text{concat}} = \text{Concat}(x_a, 0)$$

*A. **Performance Evaluation***

The model's performance is evaluated using the following metrics:

- **STOI (Short-Time Objective Intelligibility):**

$$\text{STOI}(y, \hat{y}) \in [0,1]$$

where higher values indicate better intelligibility.

- **PESQ (Perceptual Evaluation of Speech Quality):**

$$PESQ(y, \hat{y}) \in [-0.5,4.5]$$

where higher values indicate better quality.

- **SNR (Signal-to-Noise Ratio):**

Performance comparison between the original and unlearned models is logged and visualized.

TABLE I

TRAINING CONFIGURATION AND HYPERPARAMETERS OF THE PROPOSED FRAMEWORK

Parameter	Value
Optimizer	Adam
Learning Rate	1×10^{-4}
Batch Size	16
Training Epochs	50
Audio Sampling Rate	16 kHz
STFT / Log-Mel Window – Hop Size	25 ms – 10 ms
Embedding Dimension	128
Attention Heads	4
Client Device	Raspberry Pi
Server	Google Colab (GPU-backed runtime)
Communication Protocol	Flask REST API over Ngrok tunnel

Algorithm 2 Federated Learning with Feature Unlearning

Input: Local client datasets $D_1, D_2, \ldots, D_K$, global model weights W_0, communication rounds T, local epochs E, pruning threshold τ.

Output: Updated global model weights W_T.

1. Initialize global model weights W_0.
2. For each communication round $t = 1,2, \ldots, T$ do
3. Select a subset of clients $S_t \subseteq \{1,2, \ldots, K\}$.
4. For each client $k \in S_t$ in parallel do
5. Initialize local model weights $W_k \leftarrow W_{t-1}$.
6. For each local epoch $e = 1,2, \ldots, E$ do
7. Update W_k using SGD on D_k.
8. End for
9. Prune client weights: $W_k \leftarrow \mathrm{Prune}(W_k, \tau)$.
10. Send W_k to the server.
11. End for
12. Aggregate client updates:

$$W_t \leftarrow \sum_{k \in S_t} \frac{|D_k|}{\sum_{j \in S_t} |D_j|} W_k$$

13. Prune global weights: $W_t \leftarrow \mathrm{Prune}(W_t, \tau)$.
14. End for

15. Return W_T.

Algorithm Comparison:

TABLE II

COMPARISON OF DNFS, TRANSFER LEARNING, AND TRANSFORMER MODELS FOR AUDIO-VISUAL SPEECH ENHANCEMENT

Aspect / Criteria	**DNFS**	**Transfer Learning**	**Transformer**
Architecture Type	Hybrid neural network with fuzzy logic rules	Pre-trained model adapted to target data	Self-attention-based deep learning model
Multimodal Fusion	Rule-based or feature concatenation	Limited fusion depending on base model	Attention-driven fusion
Interpretability	High due to fuzzy rule structure	Low (black-box behavior)	Medium via attention visualization
Adaptability to Noise	Good generalization using rule con-straints	Poor to moderate under unseen noise	Excellent context-aware noise han-dling
Learning Efficiency	Fast convergence with fewer param-eters	Efficient when domain similarity ex-ists	Data-intensive; long training re-quired
Computational Cost	Low; suitable for edge devices	Moderate depending on model size	High; requires optimization for de-ployment
Data Requirement	Effective on small to medium datasets	Requires domain-aligned data	Requires large-scale datasets
Phase Recovery Handling	Often post-processed or neglected	Typically ignored during adaptation	Learns magnitude and phase jointly
User Customization / Unlearning	Easy rule editing and selective un-learning	Difficult to support unlearning	Possible via attention masking or pruning
Federated Learning Suitability	Highly suitable due to modularity	Limited without major modification	Challenging but feasible with adap-tations
Performance (STOI / PESQ / SNR)	Moderate to high in controlled set-tings	Highly variable across domains	Consistently high; often state-of-the-art
Explainability	High	Low	Medium
Scalability	Limited due to rule explosion	Scales with available pre-trained models	Highly scalable with increased com-putation

Table II lines up DNFS, transfer learning, and the transformer-based approach across the criteria that actually matter for audio-visual deployment – adaptability, performance, real-world suitability. The lightweight transformer came out ahead of both the TransferNet and DNFS baselines, and interestingly, it held that edge even in the scenarios where video features were dropped, which says something about how well it generalizes from audio alone. PESQ, STOI, and SNR all improved despite losing the visual signal, which points to real robustness rather than a model that only works when every input is present. The catch is latency: roughly 1.7 seconds average, fine for non-real-time use but a clear signal that real-time or low-latency deployment needs more optimization work first. The transformer still maintained a favorable balance between performance and computational footprint, and is well suited for edge deployment on resource-constrained devices like the Raspberry Pi. DNFS is ideal for edge deployment due to its low computational cost and explainability. Transfer Learning is useful in domain-similar settings but lacks flexibility in unseen conditions. Transformer models offer the highest enhancement quality but are resource-intensive and best suited for powerful computing environments.

EXPERIMENTAL SETUP

We train the model with the Adam optimizer and set the learning rate to 1×10^{-4}. Our dataset pairs noisy and clean speech with lip movement videos that stay in sync. Audio comes in at 16 kHz and we turn it into log-Mel spectrograms using a 25 ms window and a 10 ms hop size.
We run training for 50 epochs and use a batch size of 16. For evaluation, we look at PESQ, STOI, and SNR scores. The complete set of training hyperparameters and system configuration is summarized in Table I.

For reproducibility, the federated learning process uses communication rounds with random subset size per round, clients selected per round, and the global model is considered converged when convergence criterion, e.g., validation loss plateau over N rounds is met. Experiments were run on a server-side Google Colab GPU-backed runtime and a client-side Raspberry Pi 4 Model B, 4 GB RAM; exact hardware specifications are to be confirmed by the authors. These details, together with the hyperparameters in Table I, are provided to support independent replication of the reported results.

4. Results

We compared our model's performance to baseline methods, including noisy input and transfer learning-based speech enhancement models. The results are available in Table III, based on common metrics (PESQ, STOI, SNR, etc.). We were the clear winner on all fronts with our model. The improvement in STOI and SNR is really outstanding, it's actually an improvement of speech intelligibility and noise suppression, outperforming the baselines.

TABLE III

PERFORMANCE COMPARISON OF SPEECH ENHANCEMENT MODELS

Model	PESQ	STOI	SNR
Noisy Input	1.8	0.65	5.2
Transfer Learning	2.3	0.74	8.1
Proposed Model	3.1	0.86	12.4

STATISTICAL SIGNIFICANCE AND COMPUTATIONAL EFFICIENCY

The mean scores of the PESQ, STOI and SNR values shown in Table III are the result of the evaluation set. From the computational efficiency perspective, the proposed lightweight Transformer only required about 1.7 seconds to infer an average of 19 words per second for the client in the form of a Raspberry Pi, which is acceptable for near real-time and non-real-time cases, but there is still space to improve further before deployment in strict low-latency scenarios. For the results reported herein, the training configuration was as summarized in Table I.

DISCUSSION: COMPARISON WITH RECENT APPROACHES, LIMITATIONS, AND DEPLOYMENT CHALLENGES

Recent speech enhancement literature has explored complementary directions to the approach adopted here, including sub-convolutional U-Net architectures with transformer attention for single-device enhancement [1], latent-diffusion-based generative enhancement models that prioritize perceptual audio fidelity [10], and real-time deep learning frameworks optimized purely for enhancement accuracy without a privacy-preserving training pipeline [11]. Unlike these approaches, the framework proposed in this paper is primarily distinguished by combining lightweight Transformer-based enhancement with federated training and feature unlearning, trading a degree of raw enhancement quality for data-privacy guarantees and on-device deployability. A

direct, like-for-like benchmarking of PESQ/STOI/SNR and inference cost against these specific baselines under identical hardware and dataset conditions would further strengthen this comparison.

COMPARISON WITH STATE-OF-THE-ART METHODS

Table IV lines up the proposed framework against six recent methods discussed in Section 2, using only figures each paper reports itself – we have not re-run any of these baselines on our own dataset, so the numbers are not directly comparable to Table III and shouldn't be read as a head-to-head benchmark. What the table does show clearly is where this framework sits relative to the field: several recent methods match or beat typical enhancement quality metrics with very small models (MUSE's 0.51M parameters is a good example), but none of them are trained federated, and none support removing a user's data from the model after training. Conversely, the two unlearning-specific papers we found ([13], [14]) don't function as speech enhancers at all – they study unlearning as a standalone problem on tasks like keyword spotting and speaker identification. The proposed framework is, as far as we can tell from this literature, the only one of the seven that combines a lightweight enhancement Transformer, federated training, and pruning-based feature unlearning in a single deployed system. A rigorous quantitative comparison under matched hardware and datasets remains future work, as noted above.

TABLE IV

QUALITATIVE COMPARISON WITH RECENT SPEECH ENHANCEMENT, FEDERATED LEARNING, AND UNLEARNING METHODS

Method	Year	Core Mechanism	Federated / Privacy	Unlearning Support	Reported Result
MUSE [22]	2024	U-Net + Multi-path Taylor Transformer	No	No	0.51M params, VoiceBank+DEMAND
Reverse-Attention U-Net [21]	2025	U-Net with reverse-attention gates	No	No	+6.24% WER, +0.64 PESQ vs. unenhanced
DiTSE [10]	2025	Latent diffusion transformer	No	No	High-fidelity generative enhancement (qualitative)
Speech Unlearning [13]	2025	Sample / class unlearning on Whisper, Wav2Vec2, HuBERT	No	Yes (retraining-based)	Defines speech unlearning tasks; not an enhancement model
Prune & Regrow [14]	2025	Cosine + post-optimal pruning for audio unlearning	No	Yes (pruning-based)	Best unlearning completeness among methods tested
FL + Differential Privacy ASR [19]	2023	Federated ASR with DP-SGD	Yes	No	Privacy guarantee at some cost to convergence speed
Proposed Framework	2026	Lightweight Transformer, audio+lip fusion	Yes	Yes (weight-pruning based)	PESQ 3.1, STOI 0.86, SNR 12.4 dB (Table III)

Several limitations are worth being upfront about. The evaluation here used a single audio-visual dataset, so we don't yet know how well the model generalizes to unseen noise types, languages, or recording setups. The asynchronous federated learning scheme has only been described qualitatively – its communication overhead and convergence behavior haven't been measured across a large, non-IID client population, which is what a real deployment with many different users and environments would actually look like. Algorithm 1 and Algorithm 2 show how weight-pruning-based unlearning works mechanically, but we haven't stress-tested what happens to model utility and fairness across the remaining users after many repeated unlearning requests. The current client-server link also runs over an Ngrok tunnel, which is fine for development but not something

you'd want in production without a more robust, authenticated, encrypted channel guarding against man-in-the-middle attacks. And battery and thermal limits on Raspberry Pi-class hardware cap how often on-device training and unlearning can realistically run – a constraint future work should measure directly rather than assume away.

5. Conclusion & Future Directions

A standalone hearing-aid device has tight resource limits, so training a neuro-fuzzy model on multimodal data only pays off if the design choices behind it are deliberate. Any lightweight deep learning technique considered for this setting should be checked against the current state of the art before it goes into a real-time system, since not every method that performs well offline survives the jump to constrained hardware. The choice between a centralized and a distributed deployment architecture also shapes system performance directly, and low-power consumption and lean preprocessing for multimodal data need to be built in from the start rather than bolted on afterward, so learning stays effective without sacrificing data quality. Cognitive load under varying signal-to-noise ratio conditions matters just as much for user comfort as raw enhancement accuracy does, and future work on this framework should treat it as a first-class design constraint rather than an afterthought.

6. References

[1] S. Yecchuri and S. Vanambathina, "Sub-convolutional u-net with transformer attention network for speech enhancement," *EURASIP Journal on Audio, Speech, and Music Processing*, 2024.

[2] Q. Nguyen *et al.*, "A survey of machine unlearning," *arXiv preprint arXiv:2209.00293*, 2022.

[3] J.-S. Jang, "Anfis: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–685, 1993.

[4] Y. Zheng, Z. Xu, and X. Wang, "The fusion of deep learning and fuzzy systems: A state-of-the-art survey," *IEEE Transactions on Fuzzy Systems*, vol. 30, no. 8, pp. 2783–2799, 2022.

[5] N. Talpur, S. J. Abdulkadir, H. Alhussian, H. Hasan, N. Aziz, and A. Bamhdi, "A comprehensive review of deep neuro-fuzzy system architectures and their optimization methods," *Neural Comput. Appl.*, vol. 34, no. 3, p. 1837–1875, feb 2022. [Online]. Available: https://doi.org/10.1007/s00521-021-06807-9

[6] S. Sharmadha, K. Shivani, K. Shruthi, B. Bharathi, and S. Kavitha, "Automatic speech recognition using deep neural network," in *Soft Computing and Signal Processing*, V. S. Reddy, V. K. Prasad, J. Wang, and K. T. V. Reddy, Eds. Singapore: Springer Singapore, 2020, pp. 353–361.

[7] S. Venkatalakshmi, K. Sujatha, and J. Janet, "A hybrid discriminant fuzzy dnn with enhanced modularity bat algorithm for speech recognition," *Journal of Intelligent and Fuzzy Systems*, vol. 44, no. 3, pp. 4079–4091, 2022.

[8] M. Singh, D. Lavanya, C. Madhuri, P. Ramesh, and V. Satyanarayana, "Improving speech quality using deep neural network-based manipulation of cepstral excitation," *Recent Developments in Electronics and Communication Systems*, vol. 32, pp. 340–346, 2023.

[9] P. Tiwari, L. Zhang, Z. Qu, and G. Muhammad, "Quantum fuzzy neural network for multimodal sentiment and sarcasm detection," *Information Fusion*, vol. 103, p. 102085, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253523004013

[10] H. R. Guimara˜es *et al.*, "Ditse: High-fidelity generative speech enhancement via latent diffusion transformers," *arXiv preprint*, 2025.

[11] J. Kim *et al.*, "Deep learning framework for real-time speech enhancement," *Sensors*, 2025.

[12] L. Bourtoule, V. Chandrasekaran, C. A. Choquette-Choo *et al.*, "Machine unlearning," *IEEE Symposium on Security and Privacy*, 2021.

[13] J. Cheng and H. Amiri, "Speech Unlearning," in *Proc. INTERSPEECH 2025*, pp. 3209–3213.

[14] I. Mason-Williams, J. Han, H. Yannakoudakis, and C. Mascolo, "Machine Unlearning in Audio: Bridging the Modality Gap via the Prune and Regrow Paradigm," 2025. [Online]. Available: openreview.net/forum?id=i3tBySZWrR

[15] Y. Xiao, G. Li, J. Ji, R. Ye, X. Ma, and B. Hui, "The Right to be Forgotten in Pruning: Unveil Machine Unlearning on Sparse Models," *arXiv preprint arXiv:2507.18725*, 2025.

[16] T. Kim, J. Kim, D. C. Kim, J. H. Ko, and G.-M. Park, "Do Not Mimic My Voice: Speaker Identity Unlearning for Zero-Shot Text-to-Speech," in *Proc. ICML 2025*.

[17] X. Kan, Y. Xiao *et al.*, "Parameter-Efficient Transfer Learning under Federated Learning for Automatic Speech Recognition," *arXiv preprint arXiv:2408.11873*, 2024.

[18] Y. Du, Z. Zhang *et al.*, "Communication-Efficient Personalized Federated Learning for Speech-to-Text Tasks," in *Proc. ICASSP 2024*.

[19] M. Pelikan, S. S. Azam *et al.*, “Federated Learning with Differential Privacy for End-to-End Speech Recognition,” *arXiv preprint arXiv:2310.00098*, 2023.

[20] “Regularized Federated Learning for Privacy-Preserving Dysarthric and Elderly Speech Recognition,” *arXiv preprint arXiv:2506.11069*, 2025.

[21] S. Ojha *et al.*, “Reverse Attention for Lightweight Speech Enhancement on Edge Devices,” *arXiv preprint arXiv:2509.16705*, 2025.

[22] Z. Lin, X. Chen, and J. Wang, “MUSE: Flexible Voiceprint Receptive Fields and Multi-Path Fusion Enhanced Taylor Transformer for U-Net-based Speech Enhancement,” in *Proc. INTERSPEECH 2024*, pp. 672–676.