

Algorithmic Framework for Speech Command Recognition Using Hybrid Deep Learning Models

Bhagya Shree

Department of Computer Science and Applications,
Panjab University, Chandigarh, India.
Corresponding author: shreebhagya91@gmail.com
ORCID: <https://orcid.org/0000-0002-5342-5315>

Sonal Chawla

Department of Computer Science and Applications,
Panjab University, Chandigarh, India.
E-mail: sonal_chawla@pu.ac.in
ORCID: <https://orcid.org/0000-0002-1161-4732>

(Received on March 28, 2026; Revised on June 10, 2026 & July 22, 2026; Accepted on August 10, 2026)

Abstract

Speech command recognition is an essential subsystem for Speech to Sign Language translation systems. It facilitates communication between the hearing disabled community and society. Researchers have been exploring speech command recognition techniques to improve the accuracy and efficiency so that these can be beneficial for hearing disabled persons. This research introduces a hybrid deep learning framework named Convolutional Residual Dense Network (CResD-Net) for efficient speech command recognition. This research pursues four key objectives. First, it reviews and compares the existing speech command recognition frameworks. Second, it presents an algorithmic framework leveraging CResD-Net to enhance speech command recognition efficiency. Third, the framework is evaluated through two research studies. In Research Study 1, an ablation analysis was carried out to understand the role of each component of CResD-Net. In Research Study 2, the proposed framework was compared with existing studies from literature to evaluate its performance. Various metrics like accuracy, WER, Weighted Precision, Weighted Recall and Weighted F1-score were reported in this experiment. Finally, the outcomes of the study were analyzed to identify their implications for improving speech command recognition systems.

Keywords- Residual network, Dense neural network, Speech command recognition, Convolutional neural network.

1. Introduction

Effective communication plays an important role in social inclusion and human interaction. However, communication challenges can make it difficult for the hearing disabled to fully engage in society. Speech command recognition technology serves as a bridge by converting spoken language into text. It can then be integrated with a sign language generation module. Speech command recognition has the potential to enhance accessibility for the hearing disabled. With the rapid growth of digital technologies, speech command recognition has become a promising technology for accessibility solutions.

Speech command recognition systems have evolved from conventional to modern techniques. Conventional systems are based on statistical models like Hidden Markov Model (HMM) (Rudrappa et al., 2023) and Gaussian Mixture Models (GMM), whereas modern systems utilize deep learning sequence modeling architectures including Gated Recurrent Units (GRU), Bidirectional Long Short-Term Memory networks (BiLSTM) (Dhakal et al., 2022), Deep Neural Networks (DNN), transformers (Sun et al., 2024), Recurrent Neural Networks (RNN) and Residual Neural Network (ResNet) with different numbers of layers (Oruh et al., 2022; Tang et al., 2020). These models have substantially improved accuracy, adaptability and robustness. These features made speech command recognition more effective in real-world applications.

Speech command recognition systems are designed to accept speech as input, process it and convert it into corresponding text. These systems consist of three fundamental layers- input layer, speech recognition system layer and output layer. Speech is captured in the input layer and converted to digital form. The speech command recognition layer extracts feature from the speech. It further applies classification using machine learning models to generate accurate text representations (Gambhir et al., 2024; Liu et al., 2025). Finally, the output layer decodes predictions to produce the recognized text. This layered framework has been widely adopted in both conventional Hidden Markov Model-based frameworks and modern end-to-end deep learning frameworks.

Recent innovations in signal processing techniques have improved the robustness of the speech recognition systems operating in noisy environments. These include spectrogram-based methods, log-Mel spectrogram (Tang et al., 2020), mel-frequency cepstral coefficients (MFCC) (Dwijayanti et al., 2021), feature extraction and advanced acoustic modeling. These extracted features are classified into phonemes, words, or sentence-level representations using trained models. This classification can be rule-based, probabilistic, or deep learning-driven. These models are typically assessed using metrics such as precision, F1-score, recall, accuracy (Oruh et al., 2022) and Word Error Rate (WER) (Ren et al., 2022). Training these models on large benchmark datasets requires high computational resources for both learning and evaluation. The effective classification of recognized text is essential for enabling real-time captioning and speech-to-sign language translation applications. Although this study is limited to optimizing speech command recognition. Its outcomes are expected to provide a strong basis for future research on extending these systems toward speech-to-sign translation.

The paper is structured in the following manner. Section 2 discusses related studies in the speech command recognition field and identifies the research gaps. The research objectives are presented in Section 3. Section 4 describes the proposed methodology and the CResD-Net framework. The evaluation based on various performance metrics is presented in Section 5. The findings are discussed in Section 6. Section 7 concludes the research and outlines future directions. Section 8 provides data availability.

2. Literature Review

Speech command recognition developed rapidly in recent times. It transitioned from traditional statistical models to advanced deep learning-based techniques. Researchers have worked on improving recognition accuracy and scalability using various neural network techniques. This section presents a review of key research contributions in this domain:

Rudrappa et al. (2023) developed a multilingual speech translation system using open-source software. Their approach combined Hidden Markov and Gaussian Mixture Models with artificial neural networks and deep neural networks to enhance speech-matching accuracy. Dhakal et al. (2022) proposed a hybrid model of 1D Convolutional Neural Network, ResNet and Bidirectional LSTM. It achieved a 17.06%-character error rate (CER). Sun et al. (2024) proposed a swin-transformer-based keyword spotting method designed to reduce the computational overhead of transformer architectures while maintaining high accuracy. Their approach extracts dynamic features from MFCCs with the help of temporal convolutional networks. It was followed by a swin-transformer to capture multi-scale hierarchical features through window-based attention. On the speech commands V1 dataset, it achieved 98.01% accuracy with fewer parameters. Oruh et al. (2022) compared various models, ResNet with 18 and 34 layers, Densely Connected Convolutional Network (DenseNet) with 121 and 169 layers, VGG-16 and LSTM-RNN models for speech recognition. Their study found that LSTM-RNN obtained an accuracy of 99.36%. It reported weighted average precision, F1-score and recall as 0.70, 0.60 and 0.60, respectively on the benchmark spoken English digit dataset. Tang et al. (2020) introduced a system called Howl, incorporating various augmentation

techniques such as time stretching and time shifting. SpecAugment, synthetic noise injection and recorded noise blending were also applied. The speech commands dataset was pre-processed to extract log-Mel spectrogram frames. These were classified by the Res-8 model. This combination achieved an accuracy of 97% and 97.8% on the development dataset and test dataset respectively. Gambhir et al. (2024) proposed an end-to-end Sequential 2-Dimensional Convolutional Layer (Conv2D) architecture designed for low-resource keyword spotting. It was assessed using Hindi words. Their model used 2D convolutions to capture both spectral and temporal features from speech spectrograms. The model achieved 91.60% accuracy using spectrogram inputs on the speech commands dataset. Liu et al. (2025) designed a dynamic convolution mechanism. The model was tested on the speech commands dataset. It achieved 97% accuracy. Dwijayanti et al. (2021) implemented a deep BiLSTM algorithm for Indonesian speech recognition. The results showed that MFCC features outperformed spectrograms. It achieved a WER of 0.2745%. Ren et al. (2022) developed a novel Multi-Scale Parallel Convolution (MSPC) feature extractor utilizing multi-scale convolutional kernels. These features were classified using Visual Geometry Group Network (VGGNet). The proposed approach achieved improvements of 2.42% in Character Error Rate (CER) and 2.96% in WER on the LibriSpeech test-other dataset.

Bittner et al. (2025) presented S-Edge which reduced the computational cost through layer wise temporal down sampling. It enabled adjustments in sequence resolution between layers. It achieved 90–95% accuracy on the speech commands dataset. Ilgaz et al. (2024) evaluated various Convolutional Neural Network (CNN) architectures for automatic speech recognition using spectrograms. Their top-performing model achieved 94.60% accuracy on speech commands dataset V1. This demonstrated the effectiveness of convolutional networks for short-term acoustic tasks. Li et al. (2024) proposed an hourglass-shaped LSTM with skip connections. Two attention mechanisms and ideal ratio mask estimation were also utilized. It was tested on the TIMIT, LibriSpeech, and VoiceBank datasets and showed consistent improvements in Short Time Objective Intelligibility (STOI) and Perceptual Evaluation of Speech Quality (PESQ). Rybakov et al. (2020) implemented a multi-head attention RNN model. It was combined with SpecAugment using Keras and TensorFlow Lite (TFLite). It achieved accuracy rates of 97.2% and 98.0% on Speech Commands V1 and V2, respectively. Soundarya et al. (2023) utilized a recurrent neural network to analyze features generated by a Convolutional Neural Network. Their combined CNN–RNN framework achieved a WER of 12.4% and accuracy of 91.81%. Sardjono and Purwanto (2026) conducted a comparative analysis of 1D, 2D and 3D CNN architectures for robotic voice command recognition using the speech commands dataset. Experimental results showed that the 3D CNN achieved the highest accuracy of 89.61% with the lowest test loss of 0.406. Laaidi et al. (2026) proposed a CNN–LSTM framework with temporal attention for Arabic speech command recognition in industrial environments. Performance was evaluated using accuracy. The mel-spectrograms based framework achieved the highest baseline accuracy of 92.50% and the augmented framework achieved a maximum accuracy of 97.17%. **Table 1** presents a comparative analysis of the studies based on the techniques, datasets used, evaluation metrics and reported results.

Table 1. Comparative analysis of literature review studies of speech-to-text conversion systems

Study	Prevalent techniques	Dataset used	Metrics	Results/Outcomes
Laaidi et al. (2026)	CNN-LSTM with Temporal Attention and Data Augmentation	Arabic Industrial Command	Accuracy	Achieved 97.17% accuracy.
Sardjono and Purwanto (2026)	1D CNN, 2D CNN, 3D CNN	Speech Commands	Accuracy, Test Loss	3D CNN achieved 89.61% accuracy and test loss of 0.406.
Bittner et al. (2025)	S-Edge	Speech Commands	Accuracy	Achieved 90–95% accuracy.
Liu et al. (2025)	Dynamic Convolution + Mutual Learning	Speech Commands	Accuracy	Achieved 97% accuracy.

Table 1 continued...

Li et al. (2024)	Hourglass LSTM with Attention	TIMIT, LibriSpeech, VoiceBank	STOI, PESQ	Improved speech quality and intelligibility.
Sun et al. (2024)	TCN + Swin Transformer	Speech Commands	Accuracy	Achieved 98.01% accuracy.
Gambhir et al. (2024)	Sequential Conv2D	Speech Commands	Accuracy	Achieved 91.60% accuracy.
Ilgaz et al. (2024)	CNN	Speech Commands	Accuracy	Achieved 94.60% accuracy.
Soundarya et al. (2023)	CNN-RNN	Speech Commands	Accuracy, WER	Achieved 91.81% accuracy and a WER of 12.4%.
Rudrappa et al. (2023)	HMM-GMM + ANN + DNN	Multilingual Speech	Accuracy	Improved speech recognition accuracy.
Ren et al. (2022)	MSPC + VGGNet	LibriSpeech	CER, WER	Improved CER 2.42% and WER 2.96% performance.
Dhakal et al. (2022)	1D-CNN + ResNet + BiLSTM	Speech Recognition	CER	Achieved 17.06% CER.
Oruh et al. (2022)	ResNet-18, ResNet-34, DenseNet-121, DenseNet-169, VGG-16, LSTM-RNN	Spoken English Digit	Accuracy, Weighted Precision, Weighted Recall, F1-score	LSTM-RNN achieved 99.36% accuracy and Weighted Average Precision, F1-score and Recall as 0.70, 0.60 and 0.60, respectively.
Dwijayanti et al. (2021)	Deep BiLSTM	Indonesian Speech	WER	Achieved 0.2745% WER.
Tang et al. (2020)	Res-8 with Data Augmentation	Speech Commands	Accuracy	Achieved 97.8% test accuracy.
Rybakov et al. (2020)	Multi-Head Attention RNN + SpecAugment	Speech Commands	Accuracy	Achieved 98.0% accuracy on V2.

While prior research has largely emphasized improving accuracy, there remains a lack of comprehensive comparative studies on hybrid techniques evaluated across multiple performance parameters (Bittner et al., 2025; Liu et al., 2025; Sun et al., 2024). Despite significant advancements in speech command recognition, several research gaps remain that need to be addressed. Most research studies focus on evaluating individual models rather than exploring hybrid techniques that combine convolutional neural network based feature extraction with different classifiers. The lack of comprehensive analysis of such hybrid frameworks is a significant gap. These could improve the effectiveness of speech command recognition systems. Addressing these gaps can lead to improvements in the accuracy of the speech command recognition system. Many research studies primarily evaluate accuracy or WER. Other critical evaluation metrics such as weighted recall, weighted F1-score and weighted precision are neglected, which are essential for measuring the overall effectiveness of such systems.

These research gaps show the need for a detailed study of hybrid deep learning frameworks that combine feature extraction and classification effectively. A systematic comparison of these frameworks with performance evaluation metrics like weighted recall, weighted precision, weighted F1-score, accuracy and WER can help identify their strengths and limitations. There is also a demand for open-source and accessible speech command recognition frameworks that can facilitate the development of assistive technologies. Speech command recognition represents the initial stage of speech-to-sign language translation systems, making accurate recognition of command words essential because errors introduced at this stage propagate to downstream sign generation. Integrating these with text to sign language translation systems can improve communication for the hearing disabled. Motivated by these challenges, this study proposes and evaluates a CNN-based hybrid framework designed to improve command recognition performance on the Google speech commands benchmark dataset. The proposed framework provides a robust foundation for future integration with Text-to-Sign language translation systems.

3. Research Objectives

Taking these gaps into consideration, the research pursues the following objectives:

- To compare, contrast and analyze existing tools and techniques employed for speech recognition, as discussed in the literature.
- To propose and design an optimized algorithmic framework for a speech command recognition system based on hybrid deep learning employing convolutional neural networks for feature extraction.
- To evaluate the proposed framework by comparing it with various hybrid frameworks under identical experimental conditions using key metrics such as accuracy, loss, WER, weighted precision, weighted recall and weighted F1-score. Additionally, the accuracy of the proposed framework is evaluated against previously reported systems from the literature.

4. Research Methodology

Based on the identified gaps in the literature, this research focused on designing the framework for Speech-to-Text conversion in detail. The methodology consisted of three phases. Phase I - data preprocessing and proposed framework design, phase II - algorithmic framework for speech command recognition with CResD-Net and phase III - evaluation of proposed framework.

Phase I - Data preprocessing and proposed framework design focused on identifying and preparing the dataset for the study. During this phase, the dataset was collected and pre-processed to make it consistent and suitable for framework training. After preprocessing, several hybrid frameworks were designed and compared to determine the most suitable architecture for the proposed framework. In phase II - Algorithmic Framework for speech command recognition with CResD-Net, the algorithmic structure of the proposed framework was developed. This phase defined the operational steps required for implementing the CResD-Net framework for speech command recognition. In phase III - evaluation of the proposed framework, the proposed framework was assessed through two research studies. The first study compared the proposed framework with different hybrid frameworks. The second study evaluated its performance against existing state-of-the-art frameworks. **Figure 1** presents the research methodology for the proposed framework.

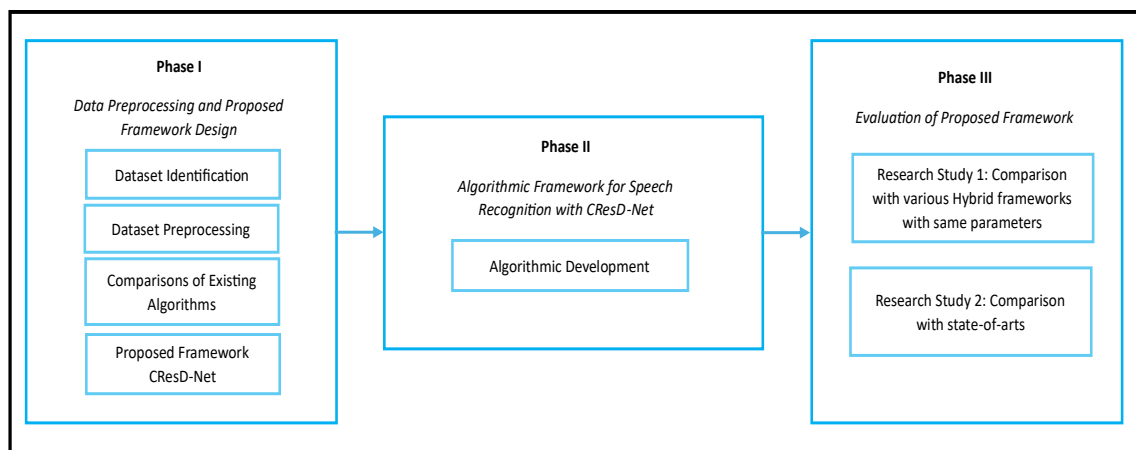


Figure 1. Research methodology of proposed framework.

Subsequent sections discuss the three phases of the proposed framework in detail.

4.1 Phase I: Data Preprocessing and Proposed Framework Design

4.1.1 Data Identification

The first phase involved data identification. This was the primary step in initiating the research process. A speech command recognition system required a dataset containing diverse spoken words and multiple speakers' accents. It could improve the generalization capability of the framework. Speech commands dataset of Google was selected for this reason. Version 1 (V1) comprised 64,727 one-second audio recordings with 30-word classes from numerous speakers. Version 2 (V2) contained 105,829 recordings across 35 classes. Its diversity in accents and pronunciations made the dataset suitable for training and evaluation of the proposed framework. **Table 2** presents the details of the speech commands dataset acquired from Google including the dataset versions, number of recordings, classes, audio duration and key characteristics relevant to speech command recognition tasks.

Table 2. Details of the speech commands' dataset.

Dataset version	Total audio recordings	Number of classes	Audio duration	Source	Key characteristics
Speech Commands V1	64,727	30	1 second	Speech Commands Dataset	Contains recordings from numerous speakers with diverse accents and pronunciations; widely used for speech recognition research.
Speech Commands V2	1,05,829	35	1 second	Speech Commands Dataset	Extended version of V1 with additional recordings and classes; provides greater diversity in speakers, accents and pronunciations for model training and evaluation.

4.1.2 Data Preprocessing

After data identification, the data preprocessing stage was performed. In this stage, data acquisition was carried out, where recorded speech with labelled datasets was used as input. Since spoken words existed in analog form and could not be directly processed by the system, they had to first undergo preprocessing. Feature extraction served as a fundamental step in speech command recognition systems, as the quality of extracted features directly influenced the performance of the learning framework. In feature extraction, spectrogram-based representations had shown strong effectiveness in capturing both temporal and frequency information from speech signals. 2D convolutional neural networks applied to spectrogram representations demonstrated superior performance compared to traditional cepstral features like Gamma-tone Frequency Cepstral Quantities (GFCQ), Mel-Frequency Cepstral Coefficients and Bark Frequency Cepstral Coefficients (BFCC) (Gambhir et al., 2024) Therefore, spectrogram representations were adopted in this study for feature extraction.

The speech signals were initially represented as waveforms sampled at 16 kHz. Each waveform was zero-padded to 16,000 samples, normalized to the float32 range [-1, 1] and transformed into a linear-magnitude short-time fourier transform spectrogram with a Hann window of a frame length of 255 samples and a frame step of 128 samples. The resulting spectrograms had shape (124, 129, 1). Data augmentation was not employed to isolate the contribution of architectural design from augmentation effects, consistent with the goal of evaluating the framework itself. Finally, the dataset was divided into three subsets: training, validation and testing datasets for training and performance evaluation of the framework.

4.1.3 Comparison of Algorithms

Following the preprocessing stage, the next important step was to identify suitable algorithms from the literature review that could enhance the responsiveness of the proposed speech command recognition system. To achieve this several existing deep learning frameworks commonly used for speech command

recognition were explored from the literature. These frameworks were based on long short-term memory, gated recurrent unit (Dhakal et al., 2022), residual network (Oruh et al., 2022; Tang et al., 2020), deep neural network (Sun et al., 2024), and convolutional neural network. The advantages and limitations of these frameworks were analyzed to understand their suitability for speech classification tasks. **Table 3** highlights the key features and limitations of LSTM, GRU, DNN and ResNet models for speech classification.

Table 3. Comparisons of various deep learning neural network models for speech classification.

Model	Key features	Computational complexity	Memory requirement	Training time	Limitations
LSTM (Dhakal et al., 2022; Laaidi et al. 2026)	A recurrent neural network designed for sequential data processing. It can retain long-term contextual information in speech sequences. It is widely used for modeling temporal dependencies in audio signals.	High	High	Slow	Training can be computationally expensive and time-consuming when large speech datasets are used, due to the complex gating mechanisms and large number of parameters.
GRU (Dhakal et al., 2022)	A streamlined variant of LSTM that uses fewer gating mechanisms. It uses reduced parameter size and faster training with minimum impact on sequential relationships in speech data.	Moderate	Moderate	Faster than LSTM	Although efficient, it may not capture highly complex temporal dependencies as effectively as LSTM in some speech recognition applications.
DNN (Sun et al., 2024)	A multilayer neural network capable of learning complex nonlinear mappings between input speech features and output classes, making it suitable for classification tasks.	Moderate	Moderate	Moderate	It lacks an inherent mechanism for modeling temporal or spatial dependencies in sequential speech signals.
Transformer (Sun et al., 2024)	An attention-based architecture that captures long-range dependencies through self-attention mechanisms, enabling parallel processing and effective feature extraction from speech signals.	High	High	Moderate to Fast	Self-attention operations have quadratic complexity with sequence length, resulting in high memory consumption and computational cost for long input sequences.
ResNet (Oruh et al., 2022; Tang et al., 2020)	Utilizes residual connections. It enables the training of deeper neural networks by improving gradient propagation. It prevents vanishing gradient issues. This allows better feature representation.	High	High	Moderate to Slow	Deep residual architectures may increase computational requirements and model complexity, particularly when many layers are incorporated.

Based on the findings of the literature review and the characteristics of the individual deep learning models, a controlled comparison of hybrid deep learning techniques was conducted. This resulted in identifying the most suitable architectural components for the proposed framework. The objective was to investigate how different combinations of feature extraction and classification techniques influence recognition performance. To achieve this, twelve hybrid frameworks, including CNN-LSTM, CNN-GRU, CNN-Transformer, CNN-ResNet, CNN-DNN, CNN-LSTM-Attention, CNN-LSTM-DNN, CNN-GRU-LSTM, CNN-ResNet-LSTM, CNN-ResNet-GRU, CNN-GRU-DNN and CNN-ResNet-DNN were developed. These were compared to evaluate the contribution of their key architectural components. All frameworks were trained using identical preprocessing steps to ensure a fair comparison. The same training configurations such as input and output dimensions, number of epochs, batch size and an 80:20 training-testing split on the speech commands V1 dataset with 30 classes (Ilgaz et al., 2024) were used. The only

variation among the frameworks was their architectural design and trainable parameters. This systematic comparison was performed to identify the most effective combination of techniques for the proposed framework. These frameworks were evaluated using accuracy, weighted recall, weighted precision and weighted F1-score (Lee et al., 2024; Oruh et al., 2022). **Table 4** provides the comparative analysis of speech command recognition frameworks.

Table 4. Comparative analysis of the various speech command recognition neural network frameworks.

Framework	Parameters	Accuracy	Loss	Weighted precision	Weighted recall	Weighted F1-score
CNN-ResNet	4.93 M	98.44	0.0534	98.5	98.44	98.44
CNN-LSTM	1.12 M	95.72	0.1684	95.98	95.72	95.76
CNN-LSTM-Attention	1.14 M	91.86	0.4195	92.98	91.86	91.96
CNN-Transformer	1.13 M	98.11	0.0612	98.17	98.11	98.12
CNN-DNN	6.1 M	97.26	0.0953	97.63	97.26	97.34
CNN-GRU	0.85M	97.52	0.1684	97.72	97.52	97.56
CNN-GRU-DNN	0.905 M	96.96	0.1265	96.9	96.96	97.56
CNN-LSTM-DNN	1.18 M	96.96	0.114	97.46	96.96	97.09
CNN-GRU-LSTM	1 M	96.43	0.1251	96.75	96.43	96.51
CNN-ResNet-LSTM	5.3 M	98.47	0.054	98.56	98.47	98.49
CNN- ResNet-GRU	5.2 M	98.53	0.053	98.61	98.53	98.55
CNN-ResNet-DNN	7.57 M	99.33	0.025	99.34	99.33	99.33

Among all the evaluated models, CNN-ResNet-DNN achieved the highest performance. It achieved test accuracy of 99.33% and lowest loss of 0.025. Other than these, weighted recall of 99.33%, weighted precision of 99.34% and weighted F1-score of 99.33% were reported by this hybrid framework. ResNet-based frameworks consistently outperformed their non-residual counterparts. CNN-ResNet 98.44%, CNN-ResNet-LSTM 98.47% and CNN-ResNet-GRU 98.53% achieved higher accuracy than CNN-LSTM 95.72%, CNN-GRU 97.52% and CNN-GRU-LSTM 96.43%. It demonstrated the effectiveness of residual learning for deep feature extraction. Furthermore, the addition of DNN layers to CNN-LSTM improved accuracy from 95.72% to 96.96%. Similarly, incorporating DNN with CNN-ResNet increased accuracy from 98.44% to 99.33%, indicating the benefit of enhanced classification through dense feature integration. The combination of ResNet and DNN components yielded the highest performance suggesting that residual feature enhancement coupled with dense classification provides a more effective architecture for speech command recognition. **Figure 2** presents a comparative analysis of the evaluated speech command recognition frameworks in terms of training and test accuracy and loss.

Figure 3 presents a comparative analysis of the evaluated hybrid frameworks based on accuracy, weighted precision, weighted recall and weighted F1-score.

Based on this analysis, a hybrid framework was designed by integrating the beneficial features of these models, which could result in developing a more robust and efficient speech command recognition framework. A comprehensive explanation of the proposed hybrid framework and its functioning is provided in the next section.

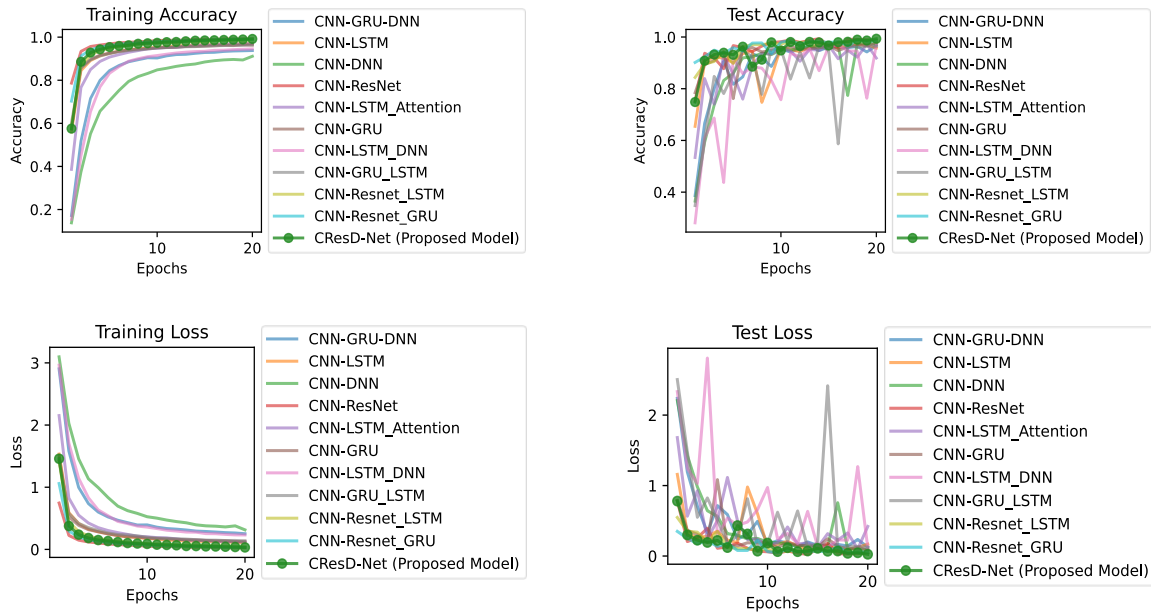


Figure 2. Comparison of various speech command recognition frameworks for accuracy and loss.

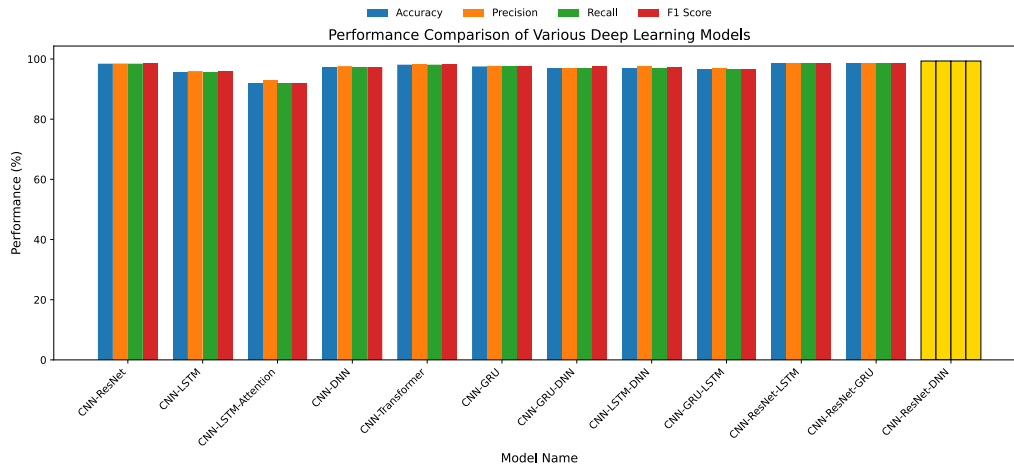


Figure 3. Comparison of frameworks for accuracy, weighted precision, weighted F1-score and weighted recall.

4.1.4 CResD-Net Framework Design

In this stage, the processed dataset from the previous stage was used as input to train and evaluate the CResD-Net (Convolutional-Residual-Dense Network) framework. The CResD-Net framework was inspired by existing hybrid models and integrated multiple deep learning components. The framework leveraged convolutional neural networks for learning spectral-temporal features. ResNet and Deep Neural Networks were integrated for deep hierarchical feature refinement and classification. The dataset was

partitioned into training, validation and testing subsets. After training on the training set, the framework was utilized on the test set to obtain predictions. CResD-Net framework could be conceptually divided into feature extraction and transfer learning and classification stages.

The proposed CResD-Net framework integrated CNN, ResNet, and DNN modules into a single framework. In this framework, the CNN captured important speech patterns, the ResNet improved feature learning and the DNN classified the speech commands. The main contribution came from the architectural design and controlled evaluation rather than modification of individual layers.

Feature Extraction and Transfer Learning

The framework applied a two-dimensional convolutional neural network layer. A larger kernel size of 7×7 and a stride of 2 were used. It extracted low level temporal and spectral features from the spectrograms. These layers captured important speech information such as formants and phonemes. Batch normalization followed by ReLU activation improved training stability.

$$y(i, j) = \sum_{m=-M}^M \sum_{n=-N}^N W(m, n) S(i - m, j - n) \quad (1)$$

where, $y(i, j)$ was the output spectrogram after convolution at position (i, j) .

$W(m, n)$ was the convolution kernel filter applied over the spectrogram.

$S(i, j)$ was the spectrogram input.

Based on the work of He et al. (2016), a ResNet based deep learning module with four residual blocks was used. Increasing filter sizes 64, 128, 256 and 512 were used. It helped in capturing complex hierarchical feature extraction in a better manner. The output of a residual block for an input x was expressed as:

$$y = f(x, \{W_i\}) + x \quad (2)$$

Each residual block was comprised of a pair of convolutional layers along with two batch normalization layers, as proposed by Ioffe and Szegedy (2015). A layer with ReLU activation introduced by Nair and Hinton (2010) was applied to facilitate gradient flow. A final convolutional layer, batch normalization and Rectified Linear Unit (ReLU) activation further refined the learned features. It ensured non-linearity and stability during training. The skip connections in ResNet prevented vanishing gradients and allowed for effective deep feature learning by capturing both local and global patterns in speech signals. A flatten layer reshaped the extracted features for classification.

Classification

The classification phase consisted of a multi-layer Dense network, where the number of neurons progressively decreased through the layers. The network architecture transitioned through Dense layers with 256, 128 and 64 neurons. In all hidden layers, the ReLU activation function was utilized to enable non-linear transformations and enhance feature extraction. These fully connected layers further refined and processed the high-level acoustic features from the convolutional front-end. They transformed these features into representations that helped the framework distinguish between different speech sounds. This was important for tasks like speech classification or keyword spotting. The final layer consisted of a Dense layer with neurons corresponding to the number of target classes, with Softmax activation applied for probability distribution over classes. This enabled the framework to generate class-wise probability scores and predict the most probable keyword or speech category for the given input.

4.1.5 Training Configuration

Model optimization was performed using the Adam optimizer. A learning rate of 0.001 was used. Sparse categorical cross-entropy was utilized as the loss function due to integer-encoded labels. CResD-Net was

trained for 20 epochs using a batch size of 64. Since the number of epochs was limited, early stopping was not applied. The proposed framework included approximately 7.57 million trainable parameters. The dataset was partitioned into 80% and 20% for training and testing respectively and compared with baseline frameworks. Additionally, the dataset was divided into 80% training, 10% validation and 10% testing. TensorFlow tf.data with autotune was used for efficient data loading and preprocessing. This split facilitated comparison with state-of-the-art frameworks.

The framework was evaluated using held-out test data and the reported metrics included accuracy, weighted recall, weighted precision and weighted F1-score. All hyperparameters, preprocessing and pipeline details along with other configuration details were represented in **Table 5**.

In this stage, the framework was applied to a subset of the test split to identify the predicted word. A random input from the test dataset was provided to the framework and then the predicted word was identified and displayed as output to the user.

This framework aligned with existing literature (Gambhir et al., 2024; Ilgaz et al., 2024; Oruh et al., 2022), where CNNs extracted spatial features, but extended further by integrating ResNet's depth efficiency and DNN's abstract learning capacity, which had been less explored directly in combination in prior works.

Table 5. Training configuration and framework details.

Parameter	Value/Description
Dataset	Speech Commands V1, V2
Dataset Split (Research Study 1)	Training - 80%, test - 20%
Dataset Split (Research Study 2)	Standard (Training - 80%, validation - 10%, test - 10%).
Feature Type	2D - Spectrogram
Input Shape	(124, 129, 1)
Output Classes	Dynamic determined at runtime from dataset directory (10, 12, 30, 35)
Data augmentation	None
Sampling rate	16 kHz
Waveform normalization	Float32, -1 to 1
Padding	Zero padding
Window function	Hann
Spectrogram type	2D STFT magnitude
Frame length (window size)	255(~16ms)
Frame step (hop length)	128(~8ms)
Spectrogram Scaling	Linear magnitude
Total Parameters	75,73,153
Trainable Parameters	75,67,393
Non-Trainable Parameters	5,760 (from batch normalization)
Loss Function	Sparse Categorical Cross-entropy loss
Optimizer	Adam optimization Algorithm
Learning Rate	0.001
Epochs	20
Batch Size	64
Dataset pipeline	TensorFlow tf.data with autotune parallel mapping
Metrics Used	Accuracy, WER, Weighted Precision, Weighted Recall, Weighted F1-score
Architecture	Input spectrogram (124×129×1) is passed through a 2D CNN stem: Conv2D (64 filters, 7×7 kernel, stride 2), Batch Normalization, ReLU, MaxPooling (3×3, stride 2). Four residual blocks follow with progressively increasing filter depth (64, 128, 256, 512); blocks 2,3,4 use stride 2 for spatial downsampling and a projection shortcut (1×1 Conv + Batch Normalization) is applied whenever channels or stride change. The flattened output feeds a dense classification head: Dense (256, ReLU), Dense (128, ReLU), Dense (64, ReLU), Dense (n_classes, Softmax)
Hardware Used	32.0 GB CPU, Windows 11 Enterprise, TensorFlow 2.x
Processor	i7-1165G7

4.2 Phase II: Algorithm of Proposed Framework CResD-Net

As proposed in the methodology, this section provides the algorithmic framework of CResD-Net. The algorithm converted raw audio into spectrogram representations from the benchmark dataset. These spectrograms were then used in training and testing of the proposed CResD-Net framework. In the framework, CNN feature extraction with diverse classifiers was used. The evaluation metrics including accuracy, WER, weighted recall, weighted precision and weighted F1-score were evaluated. CResD-Net combined the strengths of convolutional neural networks, residual networks and deep neural networks into a unified framework. The system generated corresponding textual outputs and confusion matrices were generated to evaluate the framework. The following algorithm outlines each step of this process.

Algorithm 1: Proposed Algorithm for Speech to Text Translation using CResD-Net

The algorithm takes benchmark recorded speech as input and produces its corresponding text output.

Begin

Input: Dataset path, epoch size, batch size

Output: Accuracy, Weighted Recall, Weighted Precision, Weighted F1- Score, Predicted_Label

- 1) Import necessary libraries for audio preprocessing, model training and text generation
- 2) Process_Dataset(files, desired_length=16000)
 - Decode_audio(audio_binary)
 - Get_Label(file_path)
 - Pad_waveform(waveform)
 - Get_waveform_and_label(file_path)
 - #Compute spectrograms for all waveforms of dataset audios:
 - Get_Spectrogram()
 - Spectrograms = compute_spectrogram(waveform, 255, 128)
 - Normalize spectrograms: Spectrograms = |Spectrograms|
 - Pre_Process(file_path)
 - Waveform,label= Get_waveform_and_label(file_path)
 - Get_Spectrogram()
 - #Convert text labels to numeric:
 - LabelsId = convert_labels(Labels)
 - Return Spectrogram, LabelsId
 - Dataset = Dataset(files)
 - #Preprocess and Autotune Dataset
 - Dataset = Dataset.map(preprocess,Autotune)
 - Return dataset
- 3) Load files: ds_files = load_files(dataset_location)
 - a. Training_files = ds_files[:len(ds_files)*0.8)
 - b. Validation_files = ds_files(len(ds_files)*0.8: len(ds_files)*0.9)
 - c. Testing_files = ds_files(len(ds_files)*0.9:)
- 4) #Set training parameters:
 - a. Epoch = epoch_size, Batch = batch_size
 - b. Train_s = preprocess_batch_autotune(Process Dataset(Training_files, desired_length=16000))
 - c. Test_s = preprocess_batch_autotune(Process Dataset(Testing_files, desired_length=16000))
 - d. Val_s = preprocess_batch_autotune(Process Dataset(Validation_files, desired_length=16000))
- 5) Initialize recognition timer:

```

a. Start_Time = current_time()
b. Train the CResD-Net:
c. model.fit(Train_s, validation_data=Val_s, epochs=Epoch)
d. End recognition timer: End_Time = current_time()
e. Compute recognition time: Time = End_Time - Start_Time
6) Evaluate CResD-Net:
a. Model_Loss, Model_Accuracy = model.evaluate(Test_s)
7) Generate confusion matrix and classification report (wghtd_avg_precision, wghtd_avg_recall,
wghtd_avg_F1-score)
a. Print (Accuracy, wghtd_avg_precision, wghtd_avg_recall, wghtd_avg_F1-score)
8) file_path = get_random_file(file_path)
9) Label = predict_single_command(file_path, model, commands, desired_length=16000)
10) Print (Label)
End

```

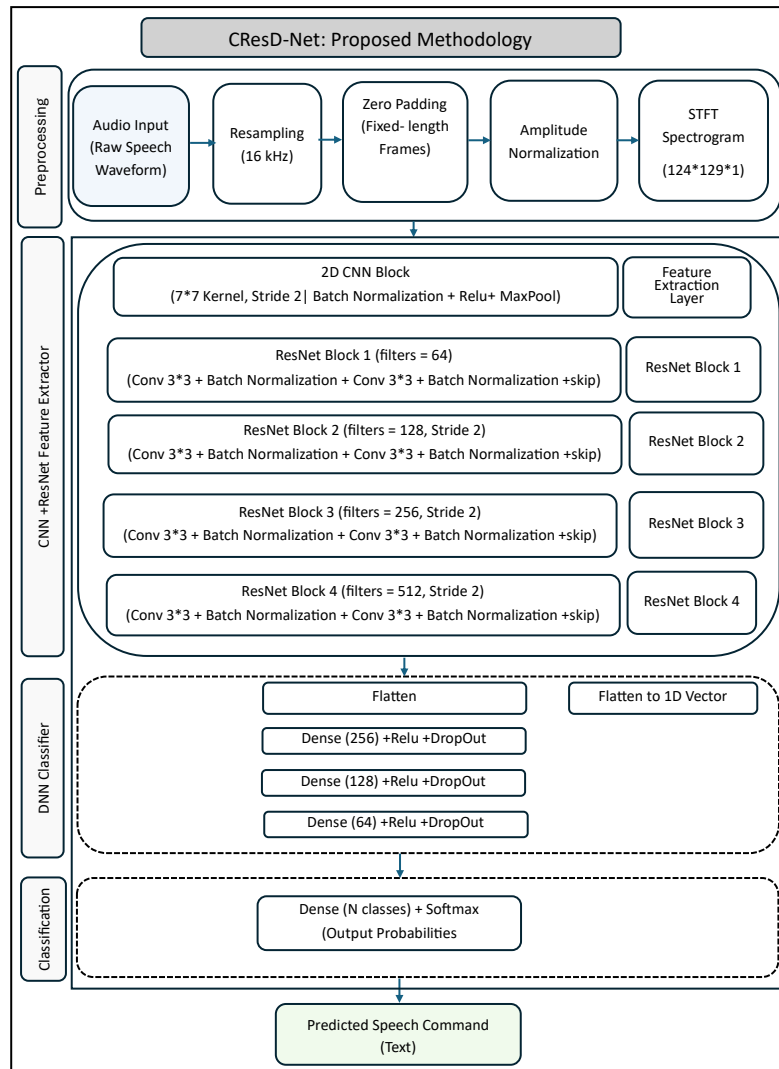


Figure 4. Flow design for proposed speech command recognition system using CResD-net framework.

Figure 4 illustrates the overall flow of the proposed CResD-Net framework for speech command recognition. Dataset preprocessing, feature extraction and classification were performed. Initially, raw speech waveforms were preprocessed through resampling to 16 kHz. Fixed length zero padding was applied and amplitude normalization was performed. Short-Time Fourier Transform (STFT) based spectrogram was generated to obtain a standardized time frequency representation. The feature extraction stage combined a 2D CNN block with four ResNet residual blocks of increasing filter sizes 64, 128, 256 and 512 to capture both low level and high-level speech characteristics. Skip connections were applied to preserve information. The extracted features were then flattened and passed to a DNN classifier. It was composed of fully connected layers with ReLU activation. Dropout was applied for improved feature discrimination and reduced overfitting. Finally, a Softmax output layer predicted the probability distribution across the speech command classes. The hierarchical integration of CNN-based feature extraction, ResNet-based deep feature enhancement and DNN-based classification enabled effective learning of discriminative speech patterns for robust speech command recognition.

4.3 Phase III: Evaluation of Proposed Framework

In this research, various frameworks were trained using the same input and output shapes, number of epochs and batch size on speech commands V1 and V2 datasets (Warden 2017). The TensorFlow (Developers, 2022) library was utilized for preprocessing and training, while the scikit-learn (sklearn) library (Pedregosa et al., 2011) was employed for evaluating the metrics. Evaluation was conducted through two research studies. In research study 1, the ablation results of the proposed framework were reported for accuracy, WER, weighted precision, weighted recall and weighted F1-score. The proposed framework was compared with existing systems in terms of accuracy in research study 2.

5. Evaluation Results

Speech commands datasets V1 and V2 were used in this work. V1 contained 64K+ audio files of 30 words recorded by multiple speakers, whereas V2 contained over 105K audio files. The proposed CResD-Net framework was trained on these versions of the dataset using different numbers of classes. This speech data was converted into spectrograms, these spectrograms were normalized and mapped with corresponding labels. The waveform of the "down" word spoken by one speaker (00176480_nohash_0.wav) is shown and its corresponding spectrogram is illustrated in **Figure 5**.

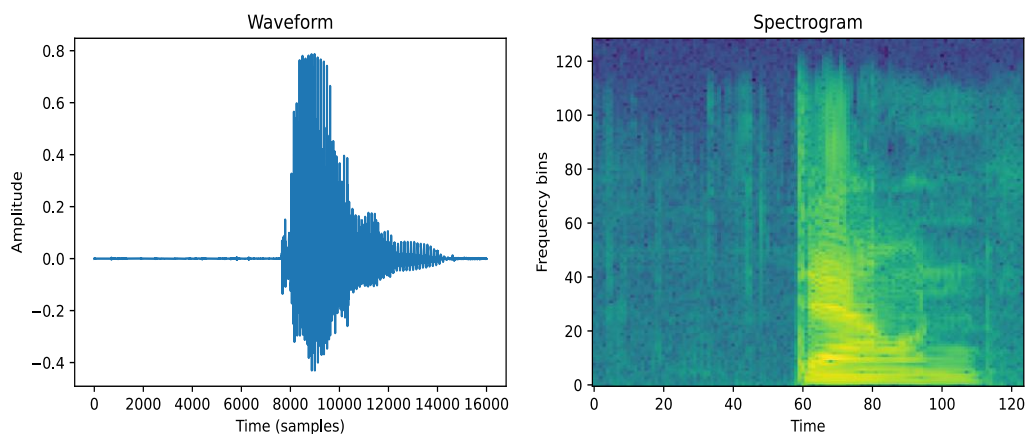


Figure 5. Waveform and spectrogram of down word.

5.1 Research Study 1: Ablation Study of CResD-Net

The ablation results in **Table 6** demonstrated that both ResNet residual blocks and DNN classification layers contributed meaningfully to CResD-Net's performance. Removing the ResNet component using CNN-DNN only resulted in a 2.07 percentage point drop in accuracy from 99.33% to 97.26%. It confirmed that deep hierarchical feature refinement using residual learning had a major impact on performance. Removing the DNN classification layers using CNN-ResNet caused a 0.89 percentage decrease from 99.33% to 98.44%. It indicated that dense classification layers provided additional discriminative capacity as well. Similar trends were observed for the Word Error Rate (WER), with the proposed framework achieving the lowest WER of 0.67%, compared with 1.56% for CNN-ResNet and 2.74% for CNN-DNN, demonstrating that the combined use of residual learning and DNN-based classification reduced command recognition errors. Notably, the full CResD-Net framework with 7.57M parameters outperformed both ablated variants despite their lower parameter count CNN-ResNet: 4.93M, CNN-DNN: 6.1M. It demonstrated that the performance gain came from the combined use of residual learning and dense classification. The improvement was not just due to a larger model capacity.

Table 6. Experiment 1 ablation study results on speech commands V1 (80:20 Split).

Framework	Parameters	Accuracy	Delta accuracy	WER %	Delta WER %	Weighted precision	Delta weighted precision	Weighted recall	Delta weighted recall	Weighted F1-score	Delta weighted F1-score
CNN-ResNet	4.93 M	98.44	-0.89	1.56	0.89	98.5	-0.84	98.44	-0.89	98.44	-0.89
CNN-DNN	6.1 M	97.26	-2.07	2.74	2.07	97.63	-1.71	97.26	-2.07	97.34	-1.99
CResD-Net (Proposed framework)	7.57 M	99.33	–	0.67	–	99.34	–	99.33	–	99.33	–

Overall, the results showed improved performance. This improvement came from combining residual learning, convolutional feature extraction and dense classification. It was not due to any single component. These findings support the contribution of the CResD-Net framework. As multiple frameworks achieved strong performance under identical experimental conditions, it indicated that the dataset was learnable. CResD-Net consistently achieved the best results. It demonstrated more effective feature utilization and improved generalization performance.

5.2 Research Study 2: Comparison of CResD-Net Framework with Existing Systems

The effectiveness of the proposed CResD-Net framework was assessed by comparing it with existing systems from reputed journals and conferences. All referenced works were evaluated on the speech commands dataset. The dataset versions, class subsets, training and test splits varied across studies. The purpose of this comparison was to present an overall performance benchmark rather than a strictly controlled experimental comparison. Due to variations in experimental configurations across studies like preprocessing strategies, model configurations, class distributions and dataset splits, the reported results are presented as a contextual performance comparison under non-uniform evaluation settings. To maintain evaluation consistency within this study, only comparable configurations reported in the literature were selected for alignment. To ensure consistency, we followed the speech commands V1 dataset with 10 (GSC V1 10), 12 (GSC V1 12) classes, V2 with 12 (GSC V2 12) and 35 classes (GSC V2-35) with standard split (80:10:10) reported in (Liu et al., 2025; Seo et al., 2021). The class-specific results are reported in **Table 7**. Although there are differences in preprocessing procedures, data augmentation techniques and experimental protocols across studies. Aligning the dataset version, class composition and data partitioning provides a more consistent basis for benchmarking. Therefore, the results presented in **Table 7** should be interpreted as a contextual performance comparison. However, these results were not obtained under fully identical evaluation conditions. The comparative analysis demonstrated the robustness and effectiveness of the

proposed CResD-Net framework across multiple speech commands benchmark configurations. The performance metrics are presented in **Table 7**.

An accuracy of 99.45% and WER of 0.55% were achieved by the proposed framework on the speech commands V2 dataset with 12 classes. It contained keywords “yes,” “no,” “up,” “down,” “left,” “right,” “on,” “off,” “stop,” “go,” unknown, or silence. For the 35-class dataset of Version V2, the framework obtained an accuracy of 99.02%, with WER of 0.98%, weighted recall, weighted precision and weighted F1-score of 99.02%, 99.05% and 99.02%, respectively. The proposed framework obtained 98.84% accuracy, along with 97.47% weighted precision, 97.43% weighted recall and 97.43% weighted F1-score on the V1 10-class dataset. WER of 1.16% was achieved in this evaluation. The inclusion of additional classes improved framework performance due to more diverse sets of speech patterns, speaker variations and phonetic structures. This increased variability enhanced the framework’s generalization capability and robustness.

Table 7. Accuracy comparison of proposed framework with existing systems from literature.

Research	Index	Purpose	Techniques	GSC V1 (10 Classes)	GSC V1 (12 Classes)	GSC V2 (12 Classes)	GSC V2 (35 Classes)	Dataset Split (Train: Validation: Test)
Sun et al. (2024)	Scopus, ESCI	Keyword Spotting	TCN + Swin- Transformer	98.01	-	-	-	Not specified
Tang et al. (2020)	-	Word Detection	Howl	-	97.8	-	-	80:10:10
Gambhir et al. (2024)	Scopus	Speech Keyword Recognition	CNN with 3 Conv2D layers	91.60	-	-	-	80:10:10
Liu et al. (2025)	SCI, Scopus	Keyword spotting	Dynamic convolution + mutual learning	-	~97%	-	-	80:10:10
Bittner et al. (2025)	SCI, Scopus	Audio Classification	State Space Models with down sampling	-	-	-	95.19	Not specified
Rybakov et al. (2020)	Scopus	Keyword Spotting	MHAtt RNN	-	97.2	98	-	80:10:10
Seo et. al. (2021)	SCIE	Keyword Spotting	Wav2KWS	-	97.9	98.5	-	80:10:10
CResD-Net (Proposed Framework)	-	Speech Command Recognition	2D CNN+ ResNet 4 layers (4 Residual Blocks: 64, 128, 256, 512 Filters) + DNN Classifier (256, 128, 64 Neurons) with Softmax Output	98.84	99.21	99.45	99.02	80:10:10

Figure 6 illustrates the accuracy comparisons of the proposed framework and state-of-the-art as mentioned in **Table 7**.

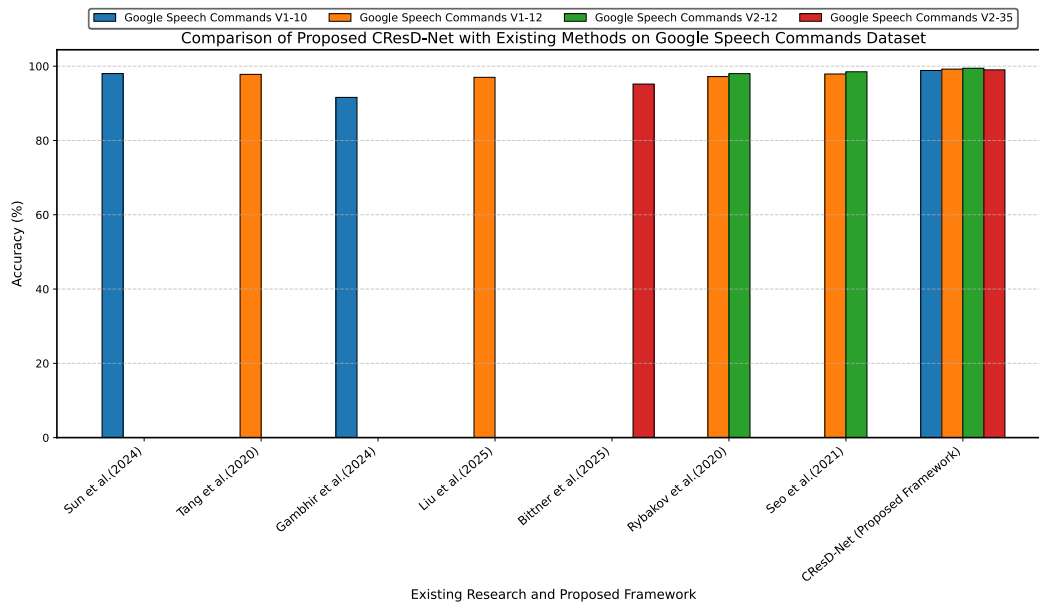


Figure 6. Comparison of various Speech command recognition systems and proposed framework for accuracy.

6. Discussion

The proposed framework leverages ResNet-inspired convolutional blocks to learn long-term dependencies in spectrograms. It combines them with dense layers to classify subtle acoustic differences. This design provides a balanced trade-off between accuracy and other parameters. This design achieves high recognition performance while enabling real-time inference on smartphone, GPU, and edge server platforms. It achieves competitive performance relative to existing state-of-the-art frameworks, as reported in **Table 7**. The framework enhances generalization compared to shallow CNNs, achieves competitive accuracy with recent state-of-the-art frameworks (Soundarya et al., 2023; Tang et al., 2020) and maintains efficiency relative to large transformer-based models (Sun et al., 2024). Extensive experiments demonstrated that the proposed framework attained state-of-the-art performance on the speech commands benchmark dataset.

Earlier CNN-ResNet hybrid frameworks used recurrent classifiers such as LSTM or GRU for classification (Dhakal et al., 2022; Sardjono & Purwanto, 2026). This design is motivated by the observation that one-second command utterances contain limited long-range temporal structure. Recurrent classifiers are largely underutilized for this task. CResD-Net uses a 4-layer ResNet with gradually increasing filter sizes of 64, 128, 256 and 512 extracts both low-level and high-level spectral-temporal features. ReLU activation in each residual block introduces non-linearity for effective feature learning. The extracted features are then passed to a dense classification network with progressively decreasing neurons of 256, 128 and 64, which progressively transforms the extracted features into more discriminative representations for classification. This proposed framework eliminates the need for recurrent modules and reduces the computational overhead associated with sequential processing. All frameworks in the ablation study were trained and evaluated under identical preprocessing, dataset splits and optimization settings. This ensures that performance differences were caused by architectural changes. This configuration provides better recognition performance than the evaluated CNN-ResNet-LSTM and CNN-ResNet-GRU frameworks under identical experimental conditions.

The scientific contribution of this work is not the combination of existing deep learning modules, but the investigation of how CNN-based feature extraction, residual feature refinement and dense classification interact within a unified framework. The study highlights the experimental validation of each framework component and provides insight into how hybrid deep learning components collectively influence performance.

The proposed CResD-Net contains 7.57 million parameters and requires 0.586 GFLOPs per inference. It achieves an average CPU inference latency of 11.14 ms for one-second utterance and enables real-time speech command recognition. The model has a larger parameter count than lightweight microcontroller-oriented models, it is intended for deployment on smartphones, GPUs, and edge-server platforms rather than memory-constrained microcontrollers. The framework is particularly suitable for speech command recognition tasks. It can further be used in keyword spotting and continuous speech recognition after evaluating it on various datasets. As speech command recognition represents the initial stage of speech-to-sign language translation, the proposed framework provides a foundation for future integration with speech-to-sign language translation systems. The proposed framework can support open and customizable speech to sign language translation systems. It can promote accessibility for the hearing disabled. The strong performance achieved across different class configurations indicates that the framework can also operate reliably under varying speech patterns and speaker characteristics. The findings indicate that the proposed integration strategy provides a more balanced trade-off between recognition accuracy and computational efficiency. The present study evaluates the proposed framework using clean speech. The robustness of the proposed framework under noisy background conditions was not investigated and remains a limitation of this study.

7. Conclusion and Future Scope

This paper reviews existing research to analyze the commonly used datasets and techniques. A key observation from the review is that many studies rely on accuracy and WER evaluation metrics. There is limited evaluation on other metrics like weighted recall, weighted precision and weighted F1-score to ensure the overall effectiveness of the frameworks. To address this limitation, a hybrid algorithmic framework for speech command recognition is proposed and implemented. It proposes CResD-Net for speech command recognition. This employs 2D-CNN layers for feature extraction, ResNet for deep feature refinement and DNN based classification into a unified framework. Several hybrid deep learning frameworks are developed and evaluated under consistent experimental conditions. The speech commands benchmark dataset is used for performance evaluation. In Research Study 1, the contribution of individual components was examined through ablation analysis. The ResNet blocks improved accuracy by 2.07%, while the DNN layers added 0.89%. In Research Study 2, CResD-Net achieved competitive performance across multiple dataset configurations. It attained accuracy of 98.84% and 99.21% on V1 with 10 and 12 classes, corresponding to WER of 1.16% and 0.79% respectively. On V2, the framework achieved 99.45% accuracy and WER of 0.55% with 12 classes and 99.02% accuracy and WER of 0.98% with 35 classes. For the V2 with 35 classes configuration, the corresponding weighted precision, weighted recall and weighted F1-score were 99.05%, 99.02% and 99.02%, respectively using an 80:10:10 split. The next step is to integrate this speech command recognition system with a text-to-sign language translation module. This will enable the system to convert recognized speech into sign language. It will enhance accessibility for individuals with hearing disabilities. The proposed system will be evaluated against existing systems to assess its effectiveness as an assistive tool to promote inclusivity in society. It will also be evaluated using noisy datasets and continuous speech datasets such as LibriSpeech to assess its practical applicability for real-time continuous speech recognition.

Conflicts of Interest

No conflicts of interest are declared by the authors for this study.

Acknowledgments

This research did not receive any specific grant from funding agencies in the public, commercial, or other sectors.

AI Disclosure

The authors confirm that generative AI tools were not used in this article preparation.

Data Availability

The benchmark speech commands dataset by Google was analyzed during experiments in the current study. The V1 and V2 datasets can be downloaded from http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz and http://download.tensorflow.org/data/speech_commands_v0.02.tar.gz respectively.

References

- Bittner, M., Schnöll, D., Wess, M., & Jantsch, A. (2025). Efficient and interpretable raw audio classification with diagonal state space models. *Machine Learning*, 114(8), 175.
- Developers, T. (2022). Tensor flow. *Zenodo*. <https://doi.org/10.5281/zenodo.4758419>
- Dhakal, M., Chhetri, A., Gupta, A.K., Lamichhane, P., Pandey, S., & Shakya, S. (2022). Automatic speech recognition for the Nepali language using CNN, bidirectional LSTM and ResNet. In *2022 International Conference on Inventive Computation Technologies* (pp. 515-521). IEEE. Nepal.
- Dwijayanti, S., Tami, M.A., & Suprpto, B.Y. (2021). Speech-to-text conversion in indonesian language using a deep bidirectional long short-term memory algorithm. *International Journal of Advanced Computer Science and Applications*, 12(3), 225-230.
- Gambhir, P., Dev, A., Bansal, P., & Sharma, D.K. (2024). End-to-end multi-modal low-resourced speech keywords recognition using sequential Conv2D nets. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 23(1), 1-21.
- He, K., Zhang, X., Ren, S., Sun, J. (2016). Identity mappings in deep residual networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) *Computer Vision – ECCV 2016. Lecture Notes in Computer Science (vol 9908)*. Springer, Cham. https://doi.org/10.1007/978-3-319-46493-0_38
- Lee, M.C.H., Braet, J., & Springael, J. (2024). Performance metrics for multilabel emotion classification: comparing micro, macro, and weighted f1-scores. *Applied Sciences*, 14(21), 9863. <https://doi.org/10.3390/app14219863>
- Ilgaz, H., Akkoyun, B., Alpay, Ö., & Akcayol, M.A. (2024). CNN based automatic speech recognition: a comparative study. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, 13, e29191-e29191.
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning* (Vol. 37, pp. 448-456). PMLR.
- Laaidi, N., Telmem, M., Lamrini, M., & Satori, H. (2026). Arabic speech command recognition using an enhanced CNN-LSTM model with attention and data augmentation. *International Journal of Speech Technology*, 29(1), 18. <https://doi.org/10.1007/s10772-025-10227-8>
- Li, Z., Basit, A., Daraz, A., & Jan, A. (2024). Deep causal speech enhancement and recognition using efficient long-short term memory recurrent neural network. *Plos One*, 19(1), e0291240.
- Liu, R., Chen, W., & Zhang, X. (2025). Dynamic convolution models for cross-frontend keyword spotting. *Scientific Reports*, 15(1), 16745. <https://doi.org/10.1038/s41598-025-00304-y>
- Nair, V., & Hinton, G.E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning* (pp. 807-814). Haifa, Israel.

- Oruh, J., Viriri, S., & Adegun, A. (2022). Long short-term memory recurrent neural network for automatic speech recognition. *IEEE Access*, 10, 30069-30079.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., & Cournapeau, D. (2011). Scikit-learn: machine learning in Python. *The Journal of Machine Learning Research*, 12, 2825-2830.
- Ren, Z., Yolwas, N., Slamu, W., Cao, R., & Wang, H. (2022). Improving hybrid ctc/attention architecture for agglutinative language speech recognition. *Sensors*, 22(19), 7319.
- Rudrappa, N.T., Reddy, M.V., & Hanumanthappa, M. (2023). KHiTE: multilingual speech acquisition to monolingual text translation. *Indian Journal of Science and Technology*, 16(21), 1572-1579.
- Rybakov, O., Kononenko, N., Subrahmanya, N., Visontai, M., & Lorenzo, S. (2020). Streaming keyword spotting on mobile devices. *arXiv preprint arXiv:2005.06720*.
- Sardjono, T.A., & Purwanto, D. (2026). A performance comparison of 1D, 2D and 3D CNN architectures for robot voice command classification. *Engineering, Technology & Applied Science Research*, 16(1), 32110-32119.
- Seo, D., Oh, H.-S., & Jung, Y. (2021). Wav2kws: transfer learning from speech representations for keyword spotting. *IEEE Access*, 9, 80682-80691.
- Soundarya, M., Karthikeyan, P.R., & Thangarasu, G. (2023). Automatic speech recognition trained with convolutional neural network and predicted with recurrent neural network. In *2023 9th International Conference on Electrical Energy Systems* (pp. 41-45). IEEE. Chennai, India.
- Sun, C., Chen, B., Chen, F., Leng, Y., & Guo, Q. (2024). Speech keyword spotting method based on SWIN-transformer model. *International Journal of Computational Intelligence Systems*, 17(1), 61. <https://doi.org/10.1007/s44196-024-00448-1>
- Tang, R., Lee, J., Razi, A., Cambre, J., Bicking, I., Kaye, J., & Lin, J. (2020). Howl: a deployed, open-source wake word detection system. In *Proceedings of Second Workshop for NLP Open Source Software* (pp. 61-65). Association for Computational Linguistics.
- Warden, P. (2017). Speech commands: a public dataset for single-word speech recognition. *Dataset Available from Http://Download. Tensorflow. Org/Data/Speech_commands_v0, 1*