Reservoir Computing for Time-Domain Audio Processing

Rinku Sebastian

PhD

University of York

School of Physics, Engineering and Technology

February 2026

Abstract

Reservoir Computing (RC) presents a promising pathway for efficient temporal signal processing, yet its application in audio has largely been confined to the role of a classifier. This perpetuates reliance on the conventional, multi-stage audio processing pipeline, where computationally expensive time-frequency transformations like Mel-Frequency Cepstral Coefficients (MFCCs) remain a bottleneck. This inefficiency limits the deployment of audio systems in real-time, low-power scenarios. In this work, we fundamentally redefine the use of RC for audio by introducing a unified framework that collapses the entire traditional pipeline. We demonstrate that a single reservoir can be configured to function not only as a classifier but also as a powerful feature extractor, operating directly on raw audio waveforms in the time domain. Specifically, we show the reservoir's high-dimensional dynamics can be trained to mimic conventional MFCC extraction, and generate equivalent features without the need for domain transformations. Our results confirm that this integrated approach successfully enables end-to-end time-domain processing from raw audio to classification, significantly reducing computational complexity and latency. This work establishes RC as a versatile and efficient alternative for audio processing, paving the way for its integration into next-generation technologies for embedded systems, and real-time speech interfaces.

Acknowledgements

Completing this thesis has been one of the most challenging yet profoundly rewarding experiences of my life. First and foremost, I express my deepest gratitude to God Almighty for granting me the strength, wisdom, and opportunity to undertake this journey.

I would like to start by sincerely thanking my supervisors, Prof. Martin Trefzer and Prof. Simon O'Keefe. Their knowledge, direction, tolerance, and mentoring have greatly influenced the direction of my research. Their enthusiasm and commitment to academic success never cease to impress me. Their feedback and encouragement have pushed me to heights I never believed imaginable.

My deepest appreciation goes to my beloved husband, Tins Jose, for being my constant source of strength, love, and support. His unwavering belief in me and his patience during the most demanding phases made this journey possible.

I am eternally grateful to my parents, Sebastian Joseph and Molly Mathew, and my parents-in-law, Jose Antony and Mercy Jose, for instilling in me the values of perseverance and hard work, and for their endless encouragement and steadfast support.

I would also like to express my deepest gratitude to our siblings and their families, as well as all our beloved relatives and friends, for their unwavering love and encouragement.

My sincere gratitude extends to the University of York's academic and administrative staff for their support over the years. Special thanks to my colleagues in the BIST group for their insightful discussions, camaraderie,

and motivation.

Looking back, I see now that this journey was never mine alone, it was woven together by the love, wisdom, and unwavering support of those who walked beside me. This achievement stands as a testament to their strength, and it is to them that I dedicate this milestone, with deepest gratitude.

Declaration and Related publications

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References.

Chapter 5 'Feature Extraction by Mimicking MFCC Function' is based on the following conference paper.

> Rinku Sebastian, Simon O'Keefe, and Martin Trefzer. 'Enhancing MFCC Feature Extraction Through Reservoir Computing.' *UCNC*, 22nd International Conference on Unconventional Computing and Natural Computing (UCNC 2025).
> *https://link.springer.com/chapter/10.1007/978-3-032-15641-920*

Chapter 6 'Feature Extraction by Convolution' is based on the following journal paper (submitted-under review).

> Rinku Sebastian, Simon O'Keefe, and Martin Trefzer. 'Bridging Biological Hearing and Neuromorphic Computing: End-to-End Time-Domain Audio Signal Processing with Reservoir Computing.' *Neural Computing and Applications*, 2025. Springer Nature.
> *https://link.springer.com/journal/521*

Appendix A 'Combining MFCC and Wavelet Features in Time-Domain' is based on the following paper.

Rinku Sebastian, Simon O'Keefe, and Martin Trefzer. 'Audio Signal Processing Using Time domain Mel Frequency wavelet coefficient' *License: CC BY 4.0arXiv:2510.24519v2 [cs.SD] 30 Oct 2025. https://arxiv.org/abs/2510.24519*

ABSTRACTS

- Efficient Real-time Audio Signal Processing Using Reservoir Computing, Workshop on unconventional Computing, Erice, Italy, 2022.
- End to End Speech Classification using Reservoir Computing, In International Workshop on Reliable and Sustainable Neuromorphic Hardware, York, UK, 2025.
- End to End Speech Classification using Reservoir Computing, In Proceedings of Dynamics Days Europe 2025 (DDE2025), Thessaloniki, Greece, 2025 (pp.277).

Contents

List of Figures

1 Introduction

The relentless progress of conventional computing architectures, while revolutionary, is approaching fundamental physical and theoretical limitations that necessitate exploration of alternative computational paradigms. As we reach the atomic limits of semiconductor technology, the scientific community faces three critical challenges that motivate the transition towards unconventional computing approaches.

First, Moore's Law reveals inherent constraints in transistor scaling, where quantum effects and thermal management issues prevent further miniaturization while maintaining reliable performance (10). Second, the Von-Neumann bottleneck, i.e., the inherent latency and energy inefficiency caused by separating memory and processing units, becomes increasingly problematic for data-intensive applications (11). Third, there exists a growing performance mismatch between conventional architectures and emerging computational needs in artificial intelligence, real-time processing, and complex system modelling (12).

Biological systems demonstrate remarkable capabilities that highlight these limitations. The human brain, for instance, performs complex pattern recognition and adaptive learning with an energy efficiency orders of

magnitude superior to digital processors. In contrast to natural systems, which routinely solve optimization problems and process noisy, uncertain information with ease, such tasks push conventional computing approaches beyond their limits.

These observations have spurred interest in alternative computing paradigms that leverage: novel physical substrates (quantum, optical, or molecular systems), Brain-inspired architectures (neuromorphic and Reservoir Computing) (13), Complex dynamical systems (chaotic and emergent computation) (12). Such unconventional approaches offer potential solutions to current limitations through inherent parallelism, energy-efficient computation, and natural tolerance to noise and variability (14).

Reservoir Computing (RC) is an emerging paradigm in unconventional computing that offers a simplified yet powerful approach to processing temporal data. Inspired by recurrent neural networks (RNNs), RC utilizes a fixed, randomly interconnected "Reservoir" of nodes, where only the output layer needs to be trained (15). This architecture eliminates the need for computationally expensive Back-Propagation Through Time (BPTT), making RC highly efficient for real-time applications. The inherent dynamics of a reservoir allow it to naturally capture temporal dependencies, which is particularly beneficial for audio and speech processing. Given its low training cost and compatibility with hardware implementations, RC presents a promising alternative to traditional deep learning methods, especially in scenarios requiring energy efficiency and rapid inference (12).

1.1. Motivation

Modern computing systems built on Von Neumann architecture and digital logic face significant challenges when dealing with real-world cognitive tasks (11). While they perform exceptionally well at executing precise mathematical operations, they struggle with processing sensory data, pattern recognition, and adaptive learning tasks that biological systems handle effortlessly (13). This efficiency gap has motivated the exploration of bio-inspired computing paradigms that move beyond traditional digital logic.

The human brain serves as a powerful inspiration, demonstrating remarkable computational efficiency through massively parallel networks of neurons that process continuous analogue signals (16). This biological architecture enables real-time perception and learning with minimal energy, capabilities that remain challenging for conventional computers to replicate (17). Rather than executing programmed algorithms sequentially, the brain's operation suggests that computation can be viewed as an emergent property of a physical system's dynamics (18).

This insight underpins the field of neuromorphic (brain-inspired) computing, which seeks to replicate the brain's information processing principles in hardware and software. Neuromorphic systems are a leading example of unconventional computing—a paradigm that exploits the natural physics of a system to perform computations— often maintaining information in its native, non-binary form (19). This thesis explores the principles of neural networks, tracing their inspiration from biological computation to their implementation as a powerful tool for overcoming the limitations of conventional

computing architectures (12).

In recent years, the landscape of temporal signal and audio processing has been radically transformed by the emergence of deep attention-based architectures, most notably Transformers (20) and Large Audio-Language Models (LLMs) (21). Frameworks like Whisper (22), AudioLM (23), and Wav2Vec 2.0 (24) leverage massive multi-head self-attention mechanisms to learn high-quality contextual representations directly from audio data, achieving state-of-the-art performance across speech recognition, synthesis, and acoustic scene classification tasks. However, these modern AI architectures come with severe computational bottlenecks. They require massive parallel hardware infrastructures, huge parameter footprints, and intense memory bandwidth during inference. This reliance makes the deployment of transformers highly impractical for low-power, edge-based, and real-time streaming scenarios. This technological constraint underscores the critical necessity for efficient alternative paradigms like Reservoir Computing (RC), which can capture complex high-dimensional temporal dynamics without the prohibitive training overhead and parameter scale of transformer-based systems (25).

One of the key advantages of RC in audio processing is its ability to process temporal sequences with minimal training overhead. Unlike traditional RNNs, which require extensive parameter tuning, RC leverages a fixed reservoir that projects input signals into a high-dimensional space, where simple linear regression can be used to train the readout layer (15). This approach significantly reduces computational complexity while maintaining competitive performance in tasks like speech recognition and sound classifi-

cation (26). Moreover, the reservoir's dynamic properties enable it to separate useful signal features from noise, enhancing robustness in real-world environments (27). These characteristics make RC particularly suitable for applications such as hearing aids, voice-activated devices, and real-time audio analysis systems (28).

A critical area of exploration is the implementation of RC in physical hardware (29), (30), which could revolutionize energy-efficient audio processing. Delay-based reservoirs (31), for example, use a single non-linear node with delayed feedback to emulate the behaviour of a large recurrent network, drastically reducing hardware complexity (6). Memristor networks (32), which rely on nanoscale resistive memories, offer another promising avenue by enabling analogue computation with minimal power consumption (33). Photonic reservoirs (34), leveraging light-based signal processing, could further accelerate real-time audio applications due to their inherent parallelism and speed (35), (36). By simplifying audio signal processing for these hardware-compatible reservoirs, researchers can contribute to the development of next-generation, low-power devices capable of human-like auditory performance.

Ongoing investigations into RC's potential in audio signal processing aim to address the limitations of current technologies while paving the way for innovative solutions. By combining the efficiency of Reservoir Computing with advancements in neuromorphic engineering, future systems could achieve real-time, energy-efficient audio processing that rivals biological systems (13). This research not only advances the field of unconventional computing but also holds significant implications for healthcare, consumer electronics, and industrial applications where reliable, low-power audio processing is essen-

tial (17). As the field progresses, the integration of hardware reservoirs with machine learning techniques may unlock new possibilities in audio signal processing.

1.2. The Reservoir Computing Paradigm

Reservoir Computing takes its conceptual roots from recurrent neural networks (RNNs), which are known for their ability to process temporal data by maintaining an internal state that evolves over time (37). However, training traditional RNNs is computationally expensive due to the challenges of back-propagation through time and vanishing gradients (38). Reservoir Computing circumvents these difficulties by adopting a unique architecture where only the output layer is trained, while the reservoir—a fixed, high-dimensional dynamical system—remains untrained (39). This reservoir serves as a complex, non-linear transformation of the input data, projecting it into a higher-dimensional space where linear separation becomes feasible (40). The elegance of RC lies in its simplicity: rather than optimizing every component of the network, it leverages the natural computational properties of the reservoir itself, whether implemented in software, hardware, or even biological substrates (14).

The versatility of Reservoir Computing is evident in its wide range of implementations. In its classical form, echo state networks (ESNs) and liquid state machines (LSMs) use randomly connected artificial neurons to create the reservoir (39), (40). More recently, researchers have demonstrated that physical systems, such as optical cavities (35), memristive devices (33), or even water droplets (41), can function as reservoirs when exploiting their

intrinsic dynamics for computation. This shift toward physical Reservoir Computing opens new possibilities for ultra-fast, low-power computing in edge devices and embedded systems (14). Furthermore, RC has proven highly effective in tasks requiring temporal memory and prediction, including speech recognition (26), robotic control (25), financial forecasting (42), and even modelling chaotic systems (15).

As we delve deeper into the principles and applications of Reservoir Computing, it becomes clear that this paradigm offers a unique combination of efficiency, scalability, and biological plausibility (43), (18). While RC is traditionally celebrated for its theoretical efficiency, its most compelling application lies in the real-time processing of high-dimensional temporal signals (44). In the context of end-to-end audio processing, RC offers a path toward low-latency feature extraction and classification without the prohibitive training costs of standard recurrent architectures. However, the transition from symbolic audio representations to raw signal processing necessitates a deeper understanding of the Reservoir. This work explores how RC can be optimized for audio signal processing, aiming to develop a framework that maintains high accuracy within an end-to-end pipeline.

1.3. Research Hypothesis and Objectives

Effective audio processing is increasingly essential for many modern technologies, including communications, computerized speech transcription and translation, speaker verification etc. (45).

The majority of contemporary audio processing entails translating audio signals to the frequency-domain. Most of these translations eliminate the

time information in the signal and introduce limitations like the irreversible loss of precise time-localized features during Fourier transformations, which significantly hinders tasks that require precise temporal alignment (46), and significant computational overhead from repeated domain conversions. These drawbacks limit neural networks' capacity to extract the best representations straight from unprocessed waveforms, and they also require significant resources for pre-processing instead of core model optimization.

A prevalent approach in audio analysis is the extraction of Mel Frequency Cepstral Coefficients (MFCC). The core strength of MFCCs lies in their biological relevance, and the filtering process is specifically designed to mimic the human auditory system, which perceives frequency non-linearly. The standard MFCC extraction process involves multiple computationally demanding stages: pre-emphasis and framing of the input signal, followed by Fourier Transformation through FFT, application of Mel-scale Filter-Banks, logarithmic compression, and finally Discrete Cosine Transform to produce the cepstral coefficients (47). Research indicates this conventional approach requires significantly more computational resources (48).

More than half of the processing time is spent on the FFT and Mel-Filter-Bank procedures alone. The repeated domain conversions not only increase latency but also create memory bottlenecks, particularly for real-time applications (49). This complexity has motivated us to explore time-domain alternatives that could potentially replicate MFCC-like features, so that the process is simplified and the burden of calculating extra time dependent coefficients is avoided.

The utility of neural networks in the audio signal processing domain has

been explored for a long time, since it is a complex task. Reservoir frameworks present an opportunity here, because the burden of training is reduced as it is limited to the readout. It is possible to solve several tasks with a single input by adding multiple readouts to a single reservoir (50). Hence, multitasking can be efficiently or effectively employed using reservoirs. The echo state property of a reservoir gives the system memory so that it can process time series (51). The fading memory property of a reservoir allows the system not to saturate. Furthermore, the reservoir has the ability to perform non-linear transformations (52). All these qualities of a reservoir show that it is a suitable fit for temporal signal processing (53).

Based on the aforementioned RC properties, this work aims to integrate reservoir dynamics directly into the audio processing chain to minimize preprocessing overhead, leading to the formulation of the following hypotheses:

1.3.1. Hypothesis

Hypothesis An end-to-end Reservoir Computing approach is capable of performing all signal processing functions within an audio processing system, including feature extraction and classification.

Sub-Hypothesis 1 (Ch4): A Reservoir based classifier achieves comparable performance to conventional classifiers using MFCC features.

Sub-Hypothesis 2 (Ch5): Reservoir Computing can successfully replicate key MFCC features, simplifying the feature extraction process exploiting time-domain operation.

Sub-Hypothesis 3 (Ch6): The effectiveness of Reservoir-based feature

extraction can be improved by training it on MFCC-inspired convolutions.

Sub-Hypothesis 4 (Ch7): A Reservoir-based architecture, operating directly on the raw audio waveform, can capture sufficient discriminative acoustic information, eliminating the need for separate feature extraction stage.

1.3.2. Research Objectives

Objective 1: To implement the RC-based practical applications (e.g., speech recognition, speaker identification) utilizing standard MFCC inputs and evaluate the performance metrics (e.g., accuracy, NRMSE), establishing a performance baseline for reservoir-based audio processing.

Objective 2: To engineer the Reservoir Computing system parameters, so that it can extract audio features (MFCC) in the time domain from the raw waveform input, demonstrating the reservoir's capacity to encode complex acoustic characteristics without explicit frequency-domain transformations.

Objective 3: To develop a reservoir-based MFCC inspired convolution feature, compare its' performance against the conventional MFCC pipeline and to validate its ability in capturing transient acoustic information.

Objective 4: To evaluate the feasibility of a unified reservoir architecture that operates on raw audio that autonomously extract salient information from raw waveforms and successfully execute classification tasks without the intervention of traditional feature extraction techniques.

Objective 5: To utilize Echo State Networks (ESNs) as a scalable framework for neuromorphic audio processing. By leveraging their inherent time-domain dynamics, this research provides a direct pathway for translating

energy-efficient, real-time speech recognition onto neuromorphic hardware, moving closer to the goal of neuromorphic hearing.

1.4. Contributions

Throughout the process of this work, several key contributions have been made to the academic discipline of audio signal processing with Reservoir Computing:

- **Evaluating the reservoir's Efficacy as a Novel Audio Feature Extractor:** This work rigorously proves the ability of the Reservoir Computing system (specifically the ESN) to function as a high-quality, non-linear audio feature extractor. This validation establishes a new paradigm for feature generation outside of conventional deep neural networks and signal processing toolkits.

- **Development of Pure Time-Domain Audio Features:** We successfully developed and validated a set of novel Audio Features in the Time Domain by leveraging Reservoir Computing. This methodology fundamentally bypasses all intermediate domain conversions (e.g., FFT, DCT, Mel-Filter Banks), thus solving the primary challenge of computational overhead and latency associated with traditional frequency-domain feature pipelines.

- **Demonstration of End-to-End, Low-Complexity Audio Processing:** This research proves the efficacy of using a simple Echo State Network (ESN) as a complete feature extraction pipeline, thereby achieving end-to-end audio processing. This significantly simplifies the

overall audio processing architecture, eliminating the need for expensive pre-processing.

- **Validation as a Computationally Lean Alternative to MFCCs:** We validated the ability of the reservoir based features to achieve competitive performance in downstream tasks. This demonstrates the potential of time-domain Reservoir Computing as a lightweight, low-complexity alternative to resource-intensive frequency-domain methods, simplifying the audio processing pipeline.

- **Development of a Parameter-Lean and Streamlined Architecture:** The proposed model utilizes a fixed-weight reservoir with a minimal number of neurons, significantly reducing the volume of trainable parameters and the associated optimization overhead. This provides a functional alternative to conventional deep learning models, which typically rely on massive datasets and extensive gradient-based updates across high-dimensional parameter spaces.

- **Simultaneous Mapping of multiple Acoustic Coefficients within a Single Reservoir Framework** By leveraging the rich dynamical states of Reservoir Computing, we established a single, fixed reservoir capable of driving 14 independent, parallel readout layers. This architecture enables the simultaneous extraction of distinct Mel-Frequency Cepstral (MFC) coefficients from a single raw audio input, demonstrating the framework's efficiency in multiplexing diverse signal characteristics through a shared computational core.

- **Validation of Direct Waveform-to-Task Mapping:** We demon-

strate the capacity of the reservoir framework to inherently capture and represent salient acoustic patterns directly from raw waveforms. This proves that the reservoir's internal dynamical states can process audio signal directly, bypassing the need for traditional signal-to-frequency transformations or hand-engineered descriptors.

- **Advancing Neuromorphic Solutions for Audio Signal Processing:** By utilizing the principles of biologically inspired Echo State Networks (ESNs), this research bridges the gap between neuromorphic computing and current audio processing challenges, offering a scalable and energy-efficient foundation for next-generation, real-time speech and auditory recognition systems.

1.5. Thesis Outline

This thesis is structured to transition systematically from reservoir-based emulations of traditional features to fully autonomous, end-to-end raw audio processing. The following chapters detail this progression:

Chapter 1 and 2- Introduction and Background: These chapters establish the research motivation and hypothesis, define the core objectives, and outline the original contributions of the work. Additionally, they provide the necessary theoretical background and reviewing literature relevant to Reservoir Computing and audio processing.

Chapter 3- Dataset and Experiments: The chapter provides the empirical foundation of the research, detailing the datasets used and the experiment methodology used.

Chapter 4- Reservoir Computing for Audio Classification: This chapter evaluates the efficacy of Reservoir Computing (RC) as a framework for audio signal classification. It details the architectural configuration of the reservoir and provides a comprehensive analysis of its performance across various acoustic benchmarks, emphasizing its capacity for high-accuracy temporal pattern recognition.

Chapter 5- Feature Extraction by Mimicking MFCC Function: This chapter investigates the capacity of Reservoir Computing (RC) to generate features that mimic standard Mel Frequency Cepstral Coefficients (MFCCs). It sets as an initial experiments for end-to-end performance validation. The limitations in mimicking MFCC led to the development of extracting MFCC in time domain as explained in next chapter.

Chapter 6- Feature Extraction by Convolution: Building upon the mimicking approach, this chapter explores the synthesis of Mel Filter-Banks using RC-based convolution. It simplifies the entire MFCC pipeline using Reservoir Computing, achieving comparable result. The findings position Reservoir Computing as a viable, end-to-end alternative to traditional frequency-domain feature engineering in neuromorphic systems.

Chapter 7- "Feature-Free" Audio Processing: This chapter explores the transition from shallow RC architectures to Deep Reservoir Computing (DRC) for direct raw audio signals. Specifically, it analyses the performance disparities between series-connected and parallel-connected deep architectures when processing waveforms without prior feature extraction.

Chapter 8- Discussion: This chapter offers a comparative analysis of the reservoir's dual roles as a classifier and a feature extractor, addressing the

specific challenges of raw signal processing and the necessity of multi-scale temporal architectures.

Chapter 9- Conclusion and Future Work: This chapter provides a definitive summary of the study's contributions, specifically validating the use of Reservoir Computing for raw time-domain audio analysis. Furthermore, it identifies critical areas for future development.

Appendix A: Combining MFCC and Wavelet Features in Time-Domain: This appendix introduces the time domain Mel Frequency Wavelet Coefficient (MFWC). It describes the methodology for designing a Time-Domain Mel Filter-Bank (TDMFB) and evaluates the computational efficiency of extracting and combining MFCC and Wavelet Coefficients via convolution. This method is proposed as a specialized alternative to traditional frequency-domain extraction for real-time applications.

Appendix B: Supplementary Experiments: This appendix provide supplementary documentation on the signal transformation pipeline and experimental cross-validation between traditional Matlab-based methods and reservoir-based feature extraction. They also detail the mathematical frameworks for data reduction and the implementation of acoustic coefficient extraction via time-domain convolution.

2 Background

This chapter provides a comprehensive overview of the foundational theories and methodologies underlying real-time, low-power neuromorphic audio processing. It begins by tracking the progression from general Artificial Intelligence (AI) and Machine Learning (ML) to the architectural layout of Artificial Neural Networks (ANNs). This computational timeline establishes the conceptual framework for Reservoir Computing (RC), evaluating their temporal dynamics as low-overhead alternatives to fully trained recurrent structures. Following this foundational computing review, the discussion transitions to signal processing fundamentals, examining human speech production along with conventional acoustic extraction methodologies—specifically Mel-Frequency Cepstral Coefficients (MFCCs) and Wavelet Transforms. Finally, the chapter identifies critical literature gaps in current audio feature extraction.

2.1. Artificial Intelligence

Artificial Intelligence (AI) represents one of the most transformative technological advancements of the modern era, enabling machines to perform tasks that traditionally required human intelligence (54). AI systems are

capable of analysing enormous volumes of data, identifying patterns, and accurately adapting to new information by mimicking cognitive processes like learning, reasoning, problem-solving, and decision-making. From virtual assistants and recommendation systems to autonomous vehicles and medical diagnostics, AI is reshaping industries and redefining how humans interact with technology (55). Its rapid evolution continues to push the boundaries of what machines can achieve, blurring the line between human and artificial capabilities.

The foundation of AI lies in its ability to process and interpret complex data through sophisticated algorithms and computational models. A fundamental component of artificial intelligence, machine learning enables systems to learn from experience and gradually advance their performance without the need for explicit programming. Deep learning, a more advanced technique, leverages neural networks with multiple layers to model intricate patterns in unstructured data such as images, speech, and text (56). These advancements have led to breakthroughs in natural language processing, computer vision, and robotics, making AI a vital tool for resolving problems in the real world.

As AI continues to integrate into everyday life, fostering responsible innovation and governance will be crucial in maximizing its potential while mitigating risks (57). Looking ahead, AI has the potential to transform industries like healthcare, education, and sustainability, offering unprecedented opportunities to address global challenges and enhance human progress (58).

With new developments like explainable AI, quantum computing, and human-AI cooperation influencing the next wave of innovation, the future of

AI is expected to be even more dynamic. AI's revolutionary influence will only increase as researchers and developers dive into new areas, strengthening its position as a pillar of technological progress. (54).

2.2. Machine Learning

Machine Learning (ML) has emerged as a revolutionary branch of artificial intelligence that enables systems to automatically learn and improve from experience without being explicitly programmed (57). By developing algorithms that can analyse data, identify patterns, and make intelligent decisions, ML has become a driving force behind many modern technological innovations. From personalized recommendations on streaming platforms to fraud detection in banking and predictive diagnostics in healthcare, ML applications are transforming industries and enhancing decision-making processes across numerous domains (59).

At its core, machine learning relies on statistical techniques and computational models that allow computers to learn from data (60). The field encompasses several distinct approaches, including supervised learning where models are trained on labelled datasets, unsupervised learning which discovers hidden patterns in unlabelled data, and reinforcement learning where algorithms learn through trial-and-error interactions with environments (61). These methodologies power everything from speech recognition systems and image classification to autonomous vehicles and financial forecasting tools, demonstrating ML's versatility in handling diverse problem types (56).

Machine Learning strives to solve complex non-linear problems—such as recognizing intricate speech patterns or processing the temporal signals found

in delay-feedback masking—by utilizing Neural Networks as its primary computational engine (62). By mimicking biological structures, these networks allow ML to move beyond simple linear logic through layers of interconnected nodes that deconstruct high-dimensional data into manageable features (56). This architectural depth enables the system to iteratively tune internal weights, transforming raw inputs into meaningful predictions.

2.3. Artificial Neural Networks

Artificial Neural Networks (ANNs) are computational models inspired by the biological brain, designed to learn complex mappings from data (56). At their core, ANNs consist of interconnected processing units (neurons) that collectively can approximate a wide variety of non-linear functions. The fundamental operation of a single neuron involves computing a weighted sum of its inputs and applying a non-linear activation function. For an input vector $\mathbf{x} = [x_1, x_2, ..., x_N]^T$, the output y of a neuron is given by:

$$y = \phi(\mathbf{w}^T\mathbf{x} + b) \tag{2.1}$$

where $\mathbf{w}$ are the connection weights, b is a bias term, and $\phi(\cdot)$ is the activation function (e.g., tanh, ReLU).

ANNs are broadly categorized by their connectivity patterns. The two primary classes most relevant to this work are Feed-forward Neural Networks (FNNs) and Recurrent Neural Networks (RNNs).

2.3.1. Feed-forward Neural Networks

In Feed-forward Neural Networks (FNNs), information flows strictly from the input layer, through one or more hidden layers, to the output layer without any cycles. This architecture implements a static, memoryless mapping from input to output.

For a network with L layers, the forward pass is described by:

$$\mathbf{h}^{(l)} = \phi^{(l)}(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}) \tag{2.2}$$

where $\mathbf{h}^{(l)}$ is the activation vector of layer l, with $\mathbf{h}^{(0)} = \mathbf{x}$ (the input) and $\mathbf{h}^{(L)} = \mathbf{y}$ (the output). $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the weight matrix and bias vector for layer l, respectively.

FNNs are typically trained using back-propagation and gradient descent to minimize a loss function $\mathcal{L}$ (63). While powerful for spatial pattern recognition (e.g., image classification (64)), their lack of internal state makes them unsuitable for processing temporal sequences directly.

2.3.2. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) extend FNNs by introducing directed cycles in their connection graph, allowing them to maintain an internal state or memory of previous inputs (2). This makes them inherently suited for processing time-series data, such as audio or text.

The dynamics of a simple RNN at discrete time step t are governed by:

$$\mathbf{h}^{(t)} = \phi(\mathbf{W}_{in}\mathbf{x}^{(t)} + \mathbf{W}_{rec}\mathbf{h}^{(t-1)} + \mathbf{b}) \tag{2.3}$$

where $\mathbf{h}^{(t)}$ is the hidden state (or context) at time t, $\mathbf{x}^{(t)}$ is the input vector, $\mathbf{W}_{in}$ is the input weight matrix, $\mathbf{W}_{rec}$ is the recurrent weight matrix, and $\mathbf{b}$ is the bias vector. The output $\mathbf{y}^{(t)}$ is typically computed from the hidden state as $\mathbf{y}^{(t)} = \mathbf{W}_{out}\mathbf{h}^{(t)}$.

While theoretically powerful, training RNNs using back-propagation through time (BPTT) is notoriously difficult due to the problems of vanishing and exploding gradients, which hinder the learning of long-range temporal dependencies (65).

2.4. Fundamentals of Reservoir Computing

Reservoir Computing is a recently developed, bio-inspired, paradigm in machine-learning. Reservoir Computing is a framework for computation derived from the theory of recurrent neural networks that maps input signals into higher dimensional computational spaces via the dynamics of a fixed, non-linear system known as a reservoir. After the input signal is fed into the reservoir, which is treated as a 'black box' (66), a straightforward readout mechanism is trained to read the state of the reservoir and map it to the desired output (67). An RNN is created at random and just a readout is trained in Reservoir Computing.

Since RNN development is sluggish and challenging, in 2001 Wolfgang Maass and Herbert Jaeger independently suggested Liquid State Machines (4) and Echo State Networks (68) as fundamentally new approaches to RNN design and training. Reservoir Computing is a term that is being used more and more frequently to refer to this method, which has roots in computational neuroscience (69) and later consequences in machine learning as

the Backpropagation-Decorrelation (70) learning rule. The drawbacks of gradient-descent RNN training are avoided by the RC paradigm. This made it much easier to use RNNs in real-world applications and outperformed traditional fully trained RNNs in many tasks (71).

2.4.1. Advantages of Reservoir Computing

Reservoir Computing is a unique system design and has various benefits. The simplicity of the training process, which makes learning swift and stable, is the first benefit. The weights in the network are not always selected for training in the Reservoir Computing framework (72). Instead, the training is mostly for the readout portion, which reduces the number of parameters that need to be set and speeds up training considerably. The training can be carried out with a straightforward linear regression or ridge regression, and the ideal set of weights can be induced all at once through a batch learning procedure, making the entire learning process simple and stable. This is especially true if the readout part is set as linear and static weights. Reservoir Computing is simple, this does not imply that RC is less powerful than conventional machine learning techniques (73).

The second benefit is its simplicity in sequential learning or multitasking (73). Multitasking can be securely done in the Reservoir Computing framework because the training is essentially limited to the readout portion and there is no interaction between the tasks.

The third benefit is the freedom and variety in reservoir selection. The fundamental idea behind RC is to utilise the reservoir's inherent dynamics by outsourcing learning (which calls for some parameter tuning) to the readout

component. This particular configuration allows reservoirs to be any dynamical system rather than just an RNN. This idea naturally leads us to exploit the physical dynamics as a reservoir instead of using the simulated dynamics inside the PC. This framework is called Physical Reservoir Computing (PRC) (30).

Reservoirs are high-dimensional (74). Many tasks cannot be accurately solved by a simple linear relation between the u and y^{target}. In such situations one has to resort to non-linear models. A number of generic and widely used approaches to non-linear modelling are based on the idea of non-linearly expanding the input u(n) into a high-dimensional feature vector $x(n)\varepsilon R^{N_x}$, and then utilizing those features using linear methods, for instance by linear regression or computing for a linear separation hyperplane, to get a reasonable y. This transformation from input space into higher dimensional space is an advantage of reservoir (1).

2.4.2. General Model

ESNs are used in supervised temporal ML problems when the desired target output signal $y^{target}(n)\varepsilon R^{N_y}$ is known for a given training input signal $u(n)\varepsilon R^{N_u}$. Here, T is the number of data points in the training dataset, and n = 1,..., T is the discrete time. Figure 2.1 shows an example of a generic architecture of the Reservoir. The state equations used in training with a forced teacher are:

$$x(n+1) = f(W^{res}_{res}x(n) + W^{res}_{inp}u(n) + W^{res}_{out}y(n) + W^{res}_{bias}) \tag{2.4}$$

$$y^{target}(n+1) = W^{out}_{res}x(n+1) + W^{out}_{inp}u(n) + W^{out}_{out}y(n) + W^{out}_{bias} \tag{2.5}$$

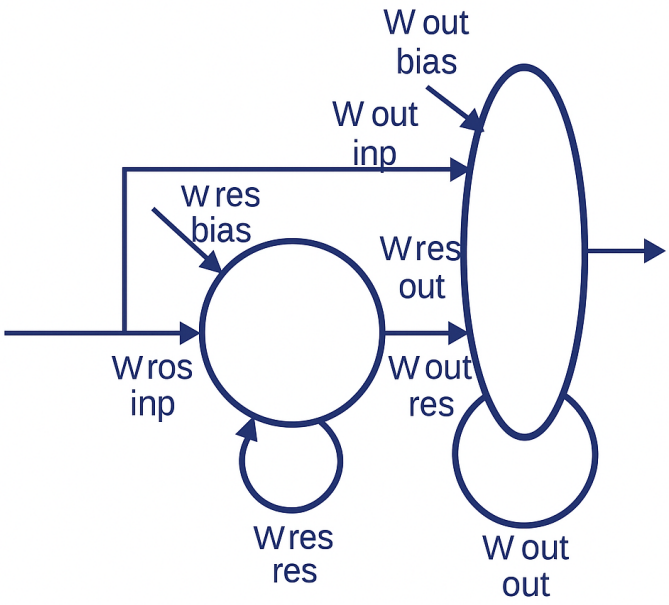


Figure 2.1: Reservoir general model (1)

Here, all weights matrices to the reservoir (W_{res}) are initialized at random, while all connections to the output (W_{out}) are trained (1).

The task is to learn a model with output $y(n)\varepsilon R^{N_y}$,where y(n) matches $y^{target}(n)$ as well as possible, minimizing an error measure $E(y, y^{target})$, and, more importantly, generalizes well to unseen data. The error measure $\in$ is typically a Mean-Square Error (MSE). For example Root-Mean-Square Error (RMSE) is given by:

$$E(y, y^{target}) = 1/N_y \sum_{i=1}^{N_y} \sqrt{1/T \sum_{n=1}^{T} (y_i(n) - y_i^{target}(n))^2}, \tag{2.6}$$

The key metric driving the evaluation and parameter optimization process is the multi-dimensional Normalized Root Mean Square Error (NRMSE). Unlike absolute error metrics, this formulation computes the tracking residuals across all target feature channels, normalized by their combined average standard deviation, and is formally defined as:

$$\text{NRMSE} = \frac{\frac{1}{N_y} \sum_{i=1}^{N_y} \sqrt{\frac{1}{T} \sum_{n=1}^{T} \left(y_i(n) - y_i^{\text{target}}(n)\right)^2}}{\frac{1}{N_y} \sum_{i=1}^{N_y} \sqrt{\frac{1}{T} \sum_{n=1}^{T} \left(y_i^{\text{target}}(n) - \overline{y}_i^{\text{target}}\right)^2}} \tag{2.7}$$

where N_y represents the total number of target feature channels, T is the discrete time horizon, $y_i(n)$ is the predicted output, $y_i^{\text{target}}(n)$ istarget output and $\overline{y}_i^{\text{target}}$ is the mean of the target. This provides a standardized, bounded measure of prediction accuracy where an NRMSE approaching 0 represents near-perfect waveform replication, while a value of 1 indicates performance no better than a primitive baseline mean predictor. Crucially, this metric is highly sensitive to the amplitude and variance of the signal. Because the

denominator depend on target volatility, higher fluctuations in signal can produce disproportionately high NRMSE spikes—even when the Echo State Network has successfully captured the core structural trends.

2.4.3. Global Parameters of the Reservoir

The tuple (W^{in}, W, α) determines the reservoir. The leakage rate is chosen as a free parameter itself, and the input and recurrent connection matrices W_{in} and W are generated at random based on the parameters specified subsequently (75).

In analogy to other machine learning, and especially neural networks, approaches, what we call parameters here could as well be called meta-parameters or hyper-parameters, as they are not concrete connection weights but parameters governing their distributions. To more accurately describe them, we shall refer to them as global parameters, or just parameters for convenience. The defining global parameters of the reservoir are: the size (N_x), sparsity, distribution of non-zero elements, and spectral radius(W) scaling(-s) of W^{in}, and the leaking rate(α) (76).

Size of Reservoir:

The number of units in the reservoir, denoted by N_x, is a clearly important parameter. According to conventional wisdom, larger reservoirs produce greater performance that can be attained, provided that overfitting is properly regularised. Reservoir sizes of order 10^4 are not unusual because an ESN is computationally inexpensive to train and maintain compared to other RNN techniques. The larger the space of reservoir signals x(n), the

easier it is to identify a linear combination of the signals to approximate the y^{target}. The reservoir can only be overly large when the task is trivial and there is a dearth of data.

Each trial should not take too long because the adjustment of global parameters frequently requires numerous iterations. Even though good parameters are typically transferable to larger reservoirs, this can be verified through several trials with larger reservoirs. Considering the number of independent real values that the reservoir must remember from the input in order to complete the task allows one to roughly estimate a bottom bound for the reservoir size N_x. The maximum number of values that may be saved, or the memory capacity, in ESN is limited to N_x. N_x must at least equal the estimated number of independent real values that the reservoir must be able to recall from the input in order to complete its duty.

Sparcity of the Reservoir:

In the early ESN publications, sparse reservoir connections that is, setting most of the elements in W to zero were advised. The reservoir's sparsity generally has little impact on performance and is not given high attention for optimization. But the performance is marginally improved by sparse connections. Also, if sparse matrix representations are applied, sparsity permits quick reservoir updates. If a fixed fanout number is specified regardless of reservoir size, the computational cost of network state updates increases only linearly rather than quadratic with the network size. This significantly lowers the cost of maintaining large reservoirs.

Distribution of Non-zero Elements:

The matrix W is normally produced sparse, and non-zero members typically have one of three distributions: a discrete bi-valued, symmetrical uniform, or normal distribution with zero at the centre. Popular distributions include Gaussian and uniform. Both distributions perform nearly identically. Although there is a non-zero chance of similar neurons in the discrete bi-valued distribution, this may make it simpler to analyse what is happening in the reservoir. Typically dense but with the same type of distribution as W, the input matrix W_{in} is created.

Spectral Radius ($\rho(W)$):

The spectral radius of the reservoir connection matrix W, or the greatest absolute eigenvalue of this matrix, is one of the most important global characteristics of an Echo state network(ESN). The matrix W is scaled, or, another way to look at it, the width of the distribution of its non-zero components is scaled.

A random sparse W is often created, its spectral radius $\rho(W)$ is calculated, W is divided by the calculated spectral radius to produce a matrix with a unit spectral radius, which is then simply scaled with the final spectral radius to be determined in a tuning operation.

The reservoir must fulfil the echo state property, according to which the state of the reservoir x(n) must be solely determined by the fading history of the input u(n) in order for the ESN technique to function. In other words, the reservoir state x(n) should not be dependent on the initial circumstances that existed before the input for a long enough input u(n).

When the reservoir's nonlinearity is high enough, large $\rho(W)$ values might cause many fixed point, periodic, or even chaotic spontaneous attractor modes, which violates the echo state property. Even though it is possible to violate the echo state property even with $\rho(W) < 1$, this is unlikely to happen in practice.

Input Scaling:

Another crucial parameter in an ESN to be optimised is the scaling of the input weight matrix (W_{in}). When input weights are normally distributed, one may use the standard deviation as a scaling measure. For uniformly distributed input weights, the input scaling is often defined as the range of the interval [a; a] from which values of W_{in} are sampled. A single scaling value is frequently used to scale all the columns of W_{in} in order to have a small number of freely changeable parameters. However, it is possible to separate the optimization of the scaling of the first column of W_{in} from the other columns, which corresponds to the bias input to the reservoir units.

It may be useful to have a bound for the range of input data values. For instance, apply the $tanh()$ squashing to u(n) if its distribution is unbounded. Otherwise, outliers might cause the reservoir state x(n) to move into regions that are unfamiliar to it and are not well covered by its typical working trajectories, for which the global parameters have been tuned or the outputs have been learned. This can lead to a virtual loss of useful memory (due to saturation in the activation non-linearities) or unpredictable outputs at these points, respectively.

The input scaling regulates the amount of nonlinearity of the reservoir

representation x(n) (also increasing with $\rho(W)$ and the relative effect of the current input on x(n) as opposed to the history (in proportion to $\rho(W)$).

Leaking Rate:

The leaking rate,is a crucial parameter that controls the speed of the reservoir's dynamics, effectively determining how quickly the reservoir state updates in response to new inputs. The reservoir update dynamics in continuous time can be described as an Ordinary Differential Equation (ODE)

$$x^{\cdot} = -x + tanh(W^{in}[1; u] + Wx). \tag{2.8}$$

Euler's discretization of this ODE in time, taking

$$\Delta x/\Delta t = (x(n+1) - x(n))/\Delta t = x^{\cdot}, \tag{2.9}$$

we arrive at exactly the discrete time equations with α taking the place of the sampling interval Δt. Thus, might be thought of as the space of time in the discrete realization of continuous universe between two successive time steps. Additionally, when the signals are slow, the effect of setting is empirically comparable to that of re-sampling u(n) and $y^{target}(n)$. In some cases setting a small α, and thus inducing slow dynamics of x(n), can dramatically increase the duration of the short-term memory in ESN (76).

While there are guiding intuitions for establishing each ESN reservoir parameter, some of them are easier to adjust than others. In ESN reservoir, the following three factors should be optimised

- input scaling,
- spectral radius,
- leaking rate.

These three factors are particularly task-specific and crucial for a good performance. While the rest of the parameters can be set to suitable default levels, the reservoir size N_x virtually acts as an external restriction.

2.4.4. Requirements

A physical reservoir must meet a number of characteristics in order to effectively complete computational activities (77), (76).

High Dimensionality:

To map inputs into a high-dimensional domain, high dimensionality is required. In classification tasks, this trait makes it possible to separate inputs that were previously inseparable, and in prediction tasks, it makes it possible to read out the spatiotemporal dependencies of inputs. The quantity of distinct signals extracted from the reservoir is correlated with the dimensionality.

Non-linearity:

In order for a reservoir to function as a non-linear mapping, nonlinearity is required. In classification tasks, this attribute enables the transformation of nonlinearly separable data into linearly separable inputs. It can also be effectively used to extract non-linear input relationships in prediction tasks.

Fading Memory:

To make sure that the reservoir state depends on recent past inputs but is independent of distant past inputs, fading memory (or short-term memory) is required. The influence of earlier inputs on the current reservoir states and outputs asymptotically fades off, which is also known as the "echo state property." A characteristic like this is crucial when describing sequential data with transient dependencies.

Separation Property:

To categorize the responses of a reservoir to various signals into discrete groups, separation property is necessary. In contrast, a reservoir ought to be impervious to unimportant minor changes like noise so that similar inputs can be grouped together. Therefore, it is frequently advised that the parameter be set close to the transition point where the transformation by a reservoir is neither extremely expanding nor very contracting when a system parameter adjustment results in a transition between non-chaotic and chaotic regimes (78).

2.4.5. Types of Reservoir Computer Systems

With the advent of Echo State Networks (ESNs) (68) and Liquid State Machines(LSMs) (4) about two decades ago, a new movement toward understanding, training, and employing Recurrent Neural Networks (RNNs) was initiated. Both approaches have the same fundamental concept, even though the former originated in the field of Machine Learning (ML) and the latter in computational neuroscience. There are many types of Reservoir Com-

puting systems developed from this basic concept, however, let us evaluate three of particular interest; Echo State Networks, Liquid State Machines, and Delayed Feedback Reservoirs.

Echo State Networks

The echo state network is based on the finding that training simply a linear readout from a random RNN with specific algebraic features can frequently lead to outstanding performance in real-world applications. Recent years have seen the emergence of echo state networks (ESNs), which are novel recurrent neural networks (RNNs) capable of processing time series with high non-linear mapping capacity and dynamic memory. The hidden layer (reservoir) of an ESN contains sparse random connections, and its only parameters are the output weights, which may be modified using straightforward linear regression. As a result, ESNs and their variants have been successfully used to predict time series with accuracy many orders of magnitude higher than that of earlier methods (79).

Reservoir Computing gets its name from the untrained RNN portion of an ESN, which is referred to as a dynamical reservoir, and the states that are produced as a result, x(n), which are known as echoes of its input history.

Figure 2.2 shows the generic structure of an Echo state network. The reservoir states of ESNs are often computed using

$$x(n) = f(W_{in}u(n) + W_x(n-1)), n = 1, ..., T, \tag{2.10}$$

where the non-linear function f () is a sigmoid, typically the $tanh()$ function. When the greatest singular value of W_{in} is less than 1, it is discovered that

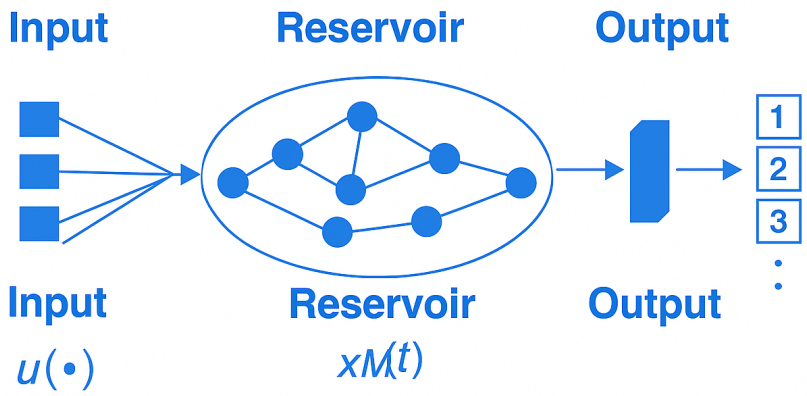


Figure 2.2: Echo state network (2)

the linearized system is asymptotically stable. In actuality, one prefers to consider the system's spectral radius since it provides a more accurate picture of its dynamics. The linearized system's hidden state will begin to increase exponentially if is bigger than one. Another popular choice for ESNs is leaky integrator neuron models. The reservoir readout is typically linear, where u(n) is expressed explicitly as being a part of x (n).

$$y(n) = f_{out}(W_{out}[u(n)|x(n)]), \tag{2.11}$$

$f_{out}()$ is the output neuron activation function (usually the identity) applied component-wise. By linearizing equation 19 around the origin, it is simple to describe the dynamics.

The spectral radius should be kept below or equal to one when initializing

reservoir weights. The network states will depend on a somewhat long history of the input if it is near to one since they will only decay to the fixed point slowly. The states will only be dependent on a brief history of the input if the spectral radius is much smaller than one. The states could theoretically depend on the complete history of the input when it is more than one, which is typically viewed as undesirable.

The ability of the network to be influenced by the recent history of the input signals is known as the "Echo State Property" and is essential to the success of RC. By adjusting the connection matrix's spectral radius, one can adjust the system's memory depth, where often a trade-off between precision and memory depth must be made.

The scaling of input weights and the scaling of an optional bias term are the other two significant factors found as being crucial for ESNs. The input scaling, or overall nonlinearity of the reservoir, will be determined by how far the hidden states are pushed away from the linear portion of the activation given function by the input. An increase in non-linearity often has a negative impact on the system's memory depth because the quenching components of the activation function will reduce the effective spectral radius (80).

Figure 2.3 shows the structure of an echo state network. The reservoir layer, W, is surrounded by the input signal, u(n), which is multiplied by the input weight matrix, $W_{in}(n)$. The output weight matrix, W_{out}, is multiplied by the output states of the reservoir, y(n). Once the error between the goal output, $y^{target(n)}$, and the actual output, y(n), has been minimised, training is completed. The output weight matrix, W_{out}, is then updated in accordance with this error. The reservoir layer's input is given an additional 1 and

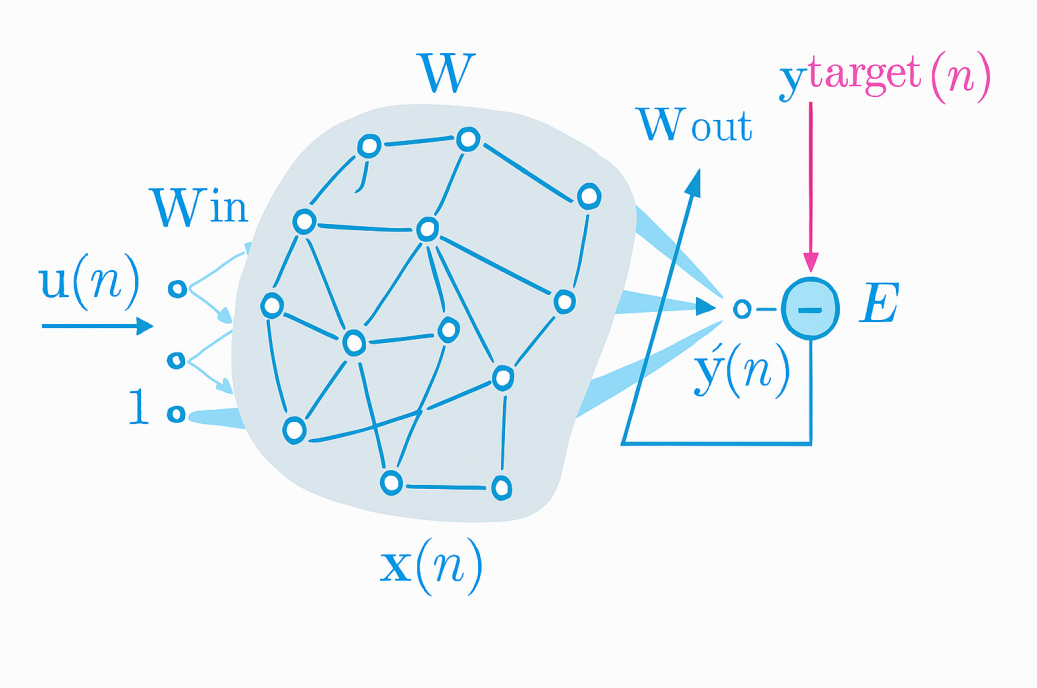


Figure 2.3: Structure of ESN (3)

utilized to bias the network (3). The system can then be simply trained by adjusting the output weights, W_{out}, until the error between y(n) and the target output is minimised to zero.

Due to its attractor dynamics, ESNs have demonstrated outstanding performance in time-series prediction and temporal pattern detection with little training. When the idea of an ESN was initially put forth, Jaeger's original publication demonstrated this. These systems have two major advantages: they are remarkably simple to train with off-line supervised learning, and because the reservoir dynamics are fixed and the only thing that changes during training are the output weights, the inputs to the reservoir shape the dynamics of the reservoir to produce a solution. However, they need some serious thought.

Three main qualities set an ESN different from a conventional RNN:

- The reservoir is driven by input signals, which result in an echo response in a high-dimensional space, allowing the reservoir to perform the same role as the kernel in kernel-based learning techniques.
- The vanishing gradient and expanding gradient issues that are present in conventional RNNs are avoided by the big reservoir with constant weights.
- The reservoir's output signal is a linear combination of linear read-out layer weights, and can be determined using straightforward linear regression procedures.

An ESN has strong computational capabilities for modelling temporal information because training it is easy and quick and it won't get stuck in local minima (79).

Liquid State Machine(LSM)

A liquid state machine (LSM) is a type of reservoir computer that uses a spiking neural network (SNN) (81). Spiking Neural Network is a third generation artificial neuron that is most biologically inspired (40). Since spiking neurons function in the temporal domain and base their computation on time resources, they are favoured over earlier generations of artificial neurons. SNNs are quickly overtaking other agents in brain-inspired neuromorphic computing, which emulates the brain using hardware (82). SNNs were chosen because of their innate effectiveness and accuracy on a variety of cognitive tasks, including speech recognition and image categorization, etc. A

high number of units make up an LSM (called nodes, or neurons). Each node receives time-varying input from both neighbouring nodes and external sources (the inputs). Nodes are joined to one another at random. The time-varying input is transformed into a spatio-temporal pattern of activations in the network nodes by the connection's recurrent nature (5). By using linear discriminant units, the spatio-temporal patterns of activation are read out.

The term liquid in the name refers to the metaphor of dropping a stone into a motionless body of water or other liquid. There will be waves in the liquid caused by the falling stone. A spatiotemporal pattern of liquid displacement has been created from the input (the motion of the falling stone) (ripples). LSMs have been proposed as a technique to describe how brains function. There is a claim that LSMs outperform the theory of artificial neural networks. (67)

Maass claims that the LSM model was created from a computational neuroscience perspective. The capacity of LSM to execute real-time calculations by upscaling the temporally variable input stream into a higher dimensional space is one of its fundamental principles (83). LSM is made up of three essential parts, namely:

- An input layer
- A reservoir or liquid
- A memoryless readout circuit.

Figure 2.4 is the architecture of an LSM. A function of time (time series) u(·) is injected as input into the liquid filter LM, creating at time t the liquid state $x^{M(t)}$, which is transformed by a memory-less readout map f^M

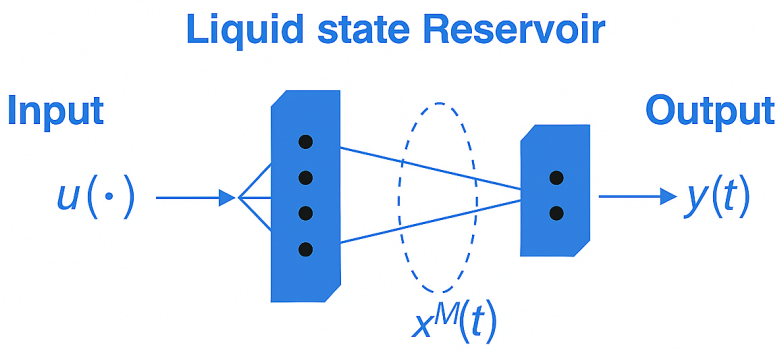


Figure 2.4: Architecture of LSM (4)

to generate an output y(t) (4).

The main distinction between the construction of an LSM and an ESN is the requirement for additional encoding/decoding hardware in the input/output layers to convert between a spike train and a real valued number (28). The same factors, such as the reservoir size and spectral radius, that must be taken into account when employing an ESN also apply when utilizing an LSM.

Being highly adaptive is one of the LSM's major advantages; in fact, the Stone-Weierstrass theorem has demonstrated that the LSM is a universal function approximator, provided that the reservoir network displays input separability and fading memory. (84) The LSM computes information in a flexible manner by mapping input spike trains onto output spike trains rather

than merely executing an algorithm. As a result, the output, which is considered to contain multi-dimensional inputs, is greatly reliant on the sort of readout and training carried out. (85) In order to enable parallel processing, numerous readout layers can be constructed, each of which extracts a distinct piece of information from the input data as shown in figure 2.5.

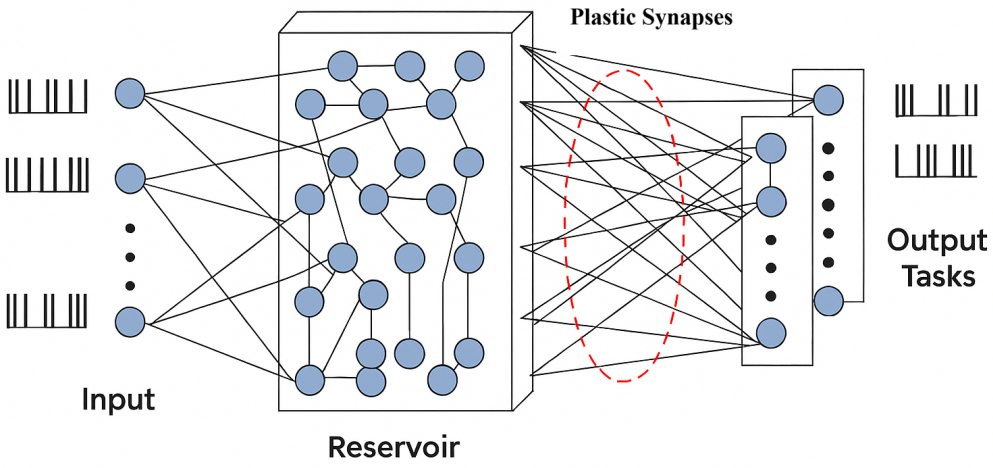


Figure 2.5: LSM supporting multitasking (5)

Delay Feedback Reservoirs

The class of dynamical systems known as delay systems, which includes nonlinear systems with delayed feedback and/or delayed coupling, has received a lot of attention. Reservoir Computing implementations based on dynamical systems with delay are known as delay-based reservoir computers. A delay Line reservoir is a special type of RC that shares resources in time to reduce

routing overhead (86). Different network nodes, often referred to as virtual nodes, are given different time slots of the non linear node response by delay-based reservoir computers. A dispersed group of virtual nodes throughout the delayed feedback loop is comparable to the temporal multiplexing of a single non-linear node's response. The usefulness of delay-based reservoir computers depends on the fact that their hardware implementations are substantially simplified (2).

There is therefore just one (hardware) node from the standpoint of the network. Delay differential equations (DDE), which mathematically describe delay systems, differ significantly from ordinary differential equations in that a DDE's time-dependent solution is not solely controlled by the state of the system at a particular instant (87). To accurately characterise the initial conditions for a DDE, the continuous solution on an interval of one delay time must be given. As a result, the high-dimensional phase space that serves as the foundation for RC is provided by a low-dimensional system with delayed feedback. Therefore, even for very high reservoir sizes, the delay-based technique enables a much simpler system layout (86). When compared to systems that require more hardware, delay-based RC has a huge benefit because of its low hardware requirements. Delay line RC at its core is a concept that involves the exchange of space and time.

Figure 2.6 shows the general delayed feedback reservoir. In contrast to the neural network, which feeds inputs in parallel to several nodes, the delayed feedback system requires that all inputs be injected into a single non-linear node. Pre-processing of the input makes up for the loss of parallelism. The masking procedure will now be used to refer to this pre-processing. It ensures

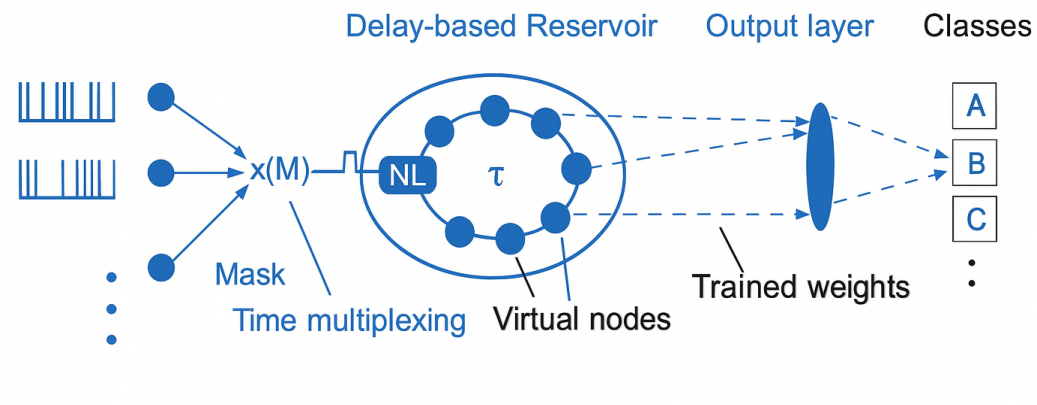


Figure 2.6: Delay line Reservoir Computer schematic (2)

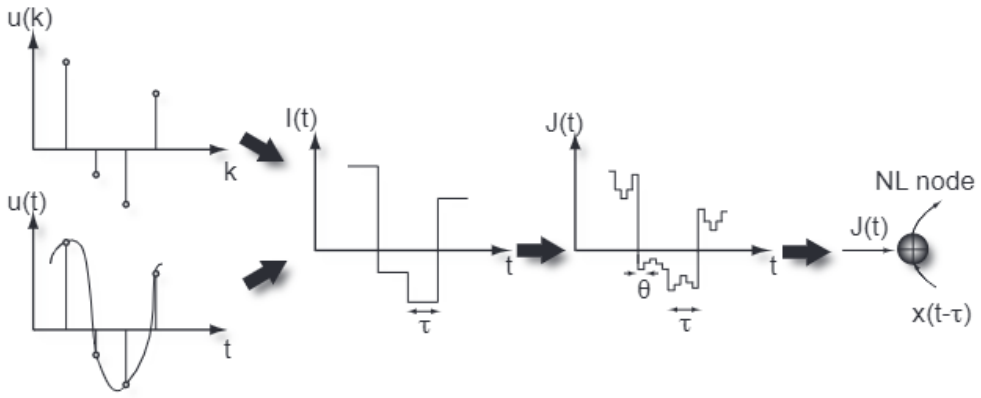


Figure 2.7: Masking Procedure (6)

that the system always exists in the transient regime by time-multiplexing the input and imprinting various scaling factors on the input. It can be thought of as the result of a convolution between a masking function and the intended system input.

Figure 2.7 shows masking procedure. A time-continuous input stream u(t) or time-discrete input u(k) undergoes a sample-and-hold operation, resulting in a stream I (t) that is constant during one delay interval τ before it is updated. The temporal input sequence, feeding the input stream to the

virtual nodes, is then given by J (t) = M ·I (t). The modified signal stays in the delay line for a while after the input is injected into the node before being re-injected into the non-linear node.

Different states present in the system are considered as the neurons or nodes of the system because the high-dimensionality of the system may be identified along the delay line. We call them virtual nodes because they don't reflect actual physical nodes. They actually represent a nonlinearly changed version of the input in the states they contain, but the transformation took place earlier in the actual non-linear node. The performance of the system is significantly influenced by the temporal separation of the various virtual nodes, which is actually the interval with which we read out states of the delay line. An output layer reads out the transient dynamical response of the node along the delay line and linearly adds them up in a weighted sum (2). Time-Delay Reservoirs (TDR) provide two significant advantages over ESNs and LSMs in addition to a decrease in routing overhead. First, adding more neurons to the reservoir is unnecessary and simply lengthens the feedback loop's delay. Second, TDRs may readily be described using delay differential equations and can employ any dynamical system to create their activation function (88).

2.5. Signals and Speech Production

In signal processing, signals including music, pictures, and scientific measurements are analysed, modified, and synthesized. Techniques for signal processing are used to enhance transmissions, increase the effectiveness of digital storage, repair distorted signals, improve the subjective quality of

video, and find or locate specific components of interest in a measured signal (89). Digital signal processing and analogue signal processing are subfields of signal processing.

2.5.1. Categories of Signal Processing

Analogue Signal Processing: Analogue signal processing is used for transmissions that haven't been digitized, like the majority of radio, telephone, and television systems from the 20th century. Both linear and non-linear electrical circuits are involved in this. Examples of the former include delay lines, additive mixers, passive and active filters, and integrators. Companders, multipliers (frequency mixers, voltage-controlled amplifiers), voltage-controlled filters, voltage-controlled oscillators, and phase-locked loops all fall under the category of non-linear circuits.

Continuous Time Signal Processing: Signals that fluctuate with the change of the continuous domain are processed in continuous time (without considering some individual interrupted points). Time domain, frequency domain, and complex frequency domain are all types of signal processing techniques. This technique focuses on modelling linear time-invariant continuous systems, calculating the zero-state response integral, configuring systems, and continuously filtering deterministic signals.

Digital Signal Processing: The processing of discrete-time sampled signals by digital means. General-purpose computers or digital circuits like ASICs, FPGAs, or specialised digital signal processors are used for processing (DSP chips). Fixed-point and floating-point, real-valued and complex-valued, multiplication and addition are common arithmetic operations (90).

Discrete Time Signal Processing: Discrete-time signal processing applies to sampled signals that are only specified at discrete points in time. As a result, they are quantized in time but not in magnitude. Electronic components like sample-and-hold circuits, analogue time-division multiplexers, analogue delay lines, and analogue feedback shift registers are the foundation of the technology known as analogue discrete-time signal processing. The advanced processing of gigahertz transmissions still uses this technology, which was a forerunner to digital signal processing. A theoretical field that creates a mathematical foundation for digital signal processing without taking quantization error into account is known as discrete-time signal processing.

2.5.2. Human Speech Production System

The larynx, which is found in the throat, is primarily responsible for creating the human voice. The vocal cords, which are two muscles stretched across the larynx, function as rubber bands. When someone talks, air rushes out of the lungs and causes the vocal cords to vibrate and thus creating the voice (91). Figure 2.8 demonstrates human speech production system. The degree to which the vocal cord muscles constrict as air from the lungs strikes them determines the pitch of the sound that is produced (92).

Loudness

The perception of loudness is a measure of how powerful a sound wave is in a specific location. It is a dimensionless quantity that is always used in a relative sense. The amplitude of the vibration determines how loud

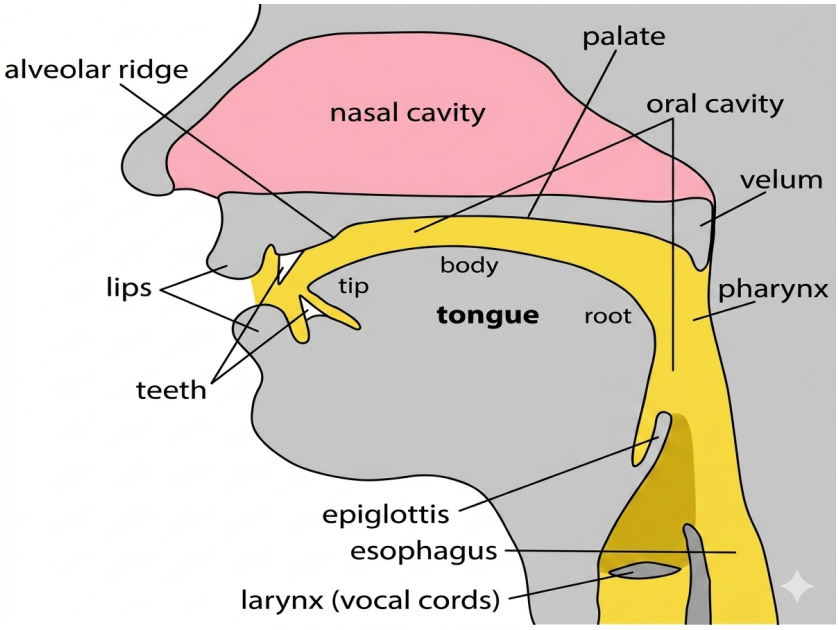


Figure 2.8: Human speech production system (7)

something is. If the amplitude is high, it will be louder. Loudness is measured in decibel (dB). It is given as: $L = log(I)$, here 'I' is the intensity.

Pitch

A subjective psycho-acoustic characteristic of sound is pitch. A perceptual characteristic called pitch enables the ranking of sounds on a frequency-related scale. The pitch of a sound is the term used to describe the perception of a frequency. Low pitch sounds are associated with low frequency sound waves, and high pitch sounds are associated with high frequency sound waves. Voiced speech signals are a type of quasi-periodic signal. The basic period is called the pitch period. Pitch is simply the rate at which vibrations are produced. Typically, this is stated as the number of Hz (hertz, or cycles per second). A full vibration back and forth is one cycle. The frequency of the tone is indicated by the number of Hz. The pitch of a tone increases with

increasing frequency.

A deeper voice results from longer vocal folds. During the mutational period, the mean fundamental frequency falls. This is associated with physiological maturation, laryngeal expansion, and an ensuing drop in mean fundamental frequency. Pitch is a characteristic that describes the shape and dimensions of the larynx and vocal folds (93).

Formants

Formants are the distinctive or significant frequency elements of human speech that are necessary for humans to discriminate between vowels. The spectral peaks of the sound spectrum are known as formants. The first formant is referred to as f1, the second as f2, and the third as f3. The first two formants, f1 and f2, are typically sufficient to distinguish vowels. The quality of vowels in terms of the open/close and front/back dimensions is determined by these two formants. It is frequently measured as a peak in the amplitude of the sound's frequency spectrum. A mathematical model of a filter is frequently used to simulate the voice tract's acoustics. The pole frequencies of this filter model are very similar to the formant frequencies. The frequencies of the poles are now referred to as formants by some voice researchers as a result. Formants are defined as the spectral peaks of the sound spectrum. The formant with the lowest frequency is called f1, the second f2, and the third f3. Most often the first two formants, f1 and f2 are enough to disambiguate vowels. These two formants determine the quality of vowels in terms of the open/ close and front/back dimensions. Formants are defined as the spectral peaks of the sound spectrum of the voice. Formant is also used to

mean an acoustic resonance of the human vocal tract. It is often measured as an amplitude peak in the frequency spectrum of the sound. The acoustics of the vocal tract are often modeled using a mathematical model of a filter. The frequencies of the poles of this filter model fall close to those of the formants. As a result, some voice researchers now refer to the frequencies of the poles as formants. So, to some voice researchers, the formant refers to a peak in the spectrum, to others it refers to a resonance of the vocal tract while to a third group it refers to the pole in a mathematical filter model (93).

2.6. Audio Signal Processing

Analysing an audio signal entails extracting its qualities, forecasting its behaviour, identifying any patterns it may include, and determining how one signal relates to other signals of a similar nature. Music, conversation, and environmental noises are all included in audio signals. In terms of signal analysis and classification, audio signal processing has developed tremendously during the previous few decades. Additionally, it has been demonstrated that many current problems can be resolved by combining advanced machine learning (ML) algorithms with audio signal processing methods. Any ML algorithm's performance is based on the features used for training and testing. Consequently, one of the most crucial steps in a machine learning process is feature extraction (94).

2.6.1. Audio Signal Feature Extraction

Feature extraction is a method of emphasizing the dominant and distinctive qualities of a signal. The process of feature extraction involves converting the speech waveform into a parametric representation at a data rate that is relatively low for further processing and analysis. The goal of feature extraction is to represent a spoken signal using a fixed amount of signal components. This is due to the fact that processing all of the information in the acoustic signal would be difficult, and some of it would be useless for the purpose (95). An appropriate feature mimics a signal's characteristics in a much more condensed manner. The evolution of audio features can be sub-categorized into time-domain, frequency-domain, joint time-frequency domain, and deep features. The time domain is used to retrieve the earliest and most basic properties. By the late 1950s, the time domain features had advanced. Around the 1960s, frequency domain features began to evolve. Pitch, formants, and other features that were developed from the frequency domain and used in a variety of applications to evaluate the spectrum of audio signals. Joint time-frequency feature extraction methods were created in the latter 1960s. Since that time, audio signal processing algorithms have leveraged these qualities. Deep features are widely employed in many applications since the development of deep learning. In audio signal processing, deep features have been used since 2010 in the domain of acoustic scene classification speaker recognition and audio video analysis.

The following section describes the audio features we have considered for our study.

MFCC

Mel Frequency Cepstral Coefficients are referred to as MFCC. The Mel-scale used is to map between linear frequency scale of speech signal to logarithmic scale for frequencies higher than 1 kHz. This makes the spectral frequency characteristics of signal closely corresponding to the human auditory perception and hence Mel Frequency Cepstral Coefficients (MFCCs) are a feature that is frequently used in automatic speech and speaker recognition. The Mel Frequency Cepstrum (MFC), which is based on a linear cosine transform of a log power spectrum on a non-linear Mel scale of frequency, is a representation of the short-term power spectrum of a sound. An MFC is made up of a number of coefficients known as Mel Frequency Cepstral Coefficients (MFCCs). They come from a particular cepstral interpretation of the audio clip. The frequency bands of the MFC are evenly spaced on the Mel scale, which simulates the response of the human auditory system. This frequency warping may make it possible to depict sound more accurately.

MFCCs are commonly derived as follows:

- Take the Fourier Transform of (a windowed excerpt of) a signal.
- Map the powers of the spectrum obtained above onto the Mel scale, using triangular overlapping windows or alternatively, cosine overlapping windows.
- Take the logs of the powers at each of the Mel frequencies.
- the Discrete Cosine Transform of the list of Mel log powers, as if it were a signal.

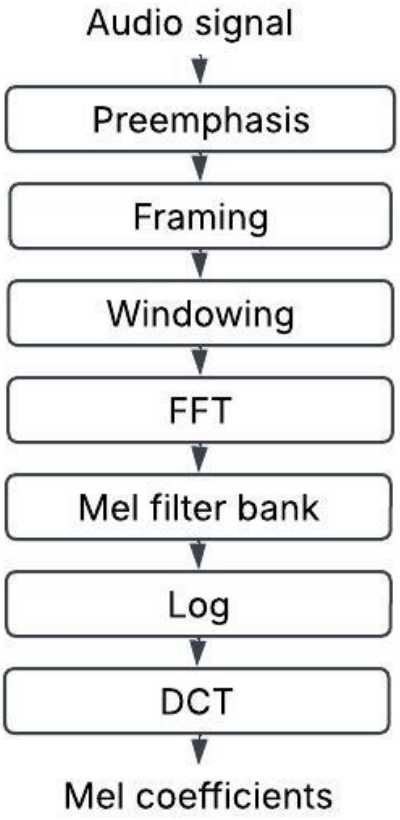


Figure 2.9: MFC extraction (8)

- The MFCCs are the amplitudes of the resulting spectrum

Framing and Windowing: The MFCCs algorithm needs to be transformed from the time domain to the frequency domain because it is based on spectral analysis. The acoustic signal is essentially stationary. The signal is believed not to be periodic for sound samples that are longer than 200 milliseconds. It is impossible to identify whether a sample that lasts between 30 and 200 milliseconds is periodic or not. It is safe to presume that a sound is periodic for samples that are shorter than 30 ms. Signal framing is required

due to this functionality (96). There should be between 20 to 30 milliseconds between each frame. Individual speech sounds' temporal properties can be followed by moving the time window forward by 10 ms at a time, and a 20 ms analysis window is typically long enough to resolve major temporal characteristics while still giving these sounds acceptable spectral resolution. The goal of the overlapping analysis is to ensure that each speech sound in the input sequence is roughly centred at a given frame. The signal is tapered towards the frame borders on each frame by applying a window. Hanning or Hamming windows are typically used. While applying the DFT to the signal, this is done to improve the harmonics, soften the edges, and lessen the edge effect.

DFT Spectrum: Each windowed frame is converted into frequency spectrum by applying DFT.

$$X(k) = \sum_{n=0}^{N-1} x(n) * e^{-j2\pi nk/N} \tag{2.12}$$

Mel Spectrum: Mel spectrum is computed by passing the Fourier transformed signal through a set of band-pass filters known as Mel Filter-Bank. A Mel is a unit of measurement based on how loudness is perceived by the human ear. Since the human auditory system reportedly does not detect pitch linearly, it does not correspond linearly to the tonal frequency physically present in the sound. The frequency spacing for the Mel scale is roughly linear below 1 kHz and logarithmic above 1 kHz. Mel can be approximated by physical frequency using the formula

$$f_{mel} = 2595 log_{10}(1 + f/700) \tag{2.13}$$

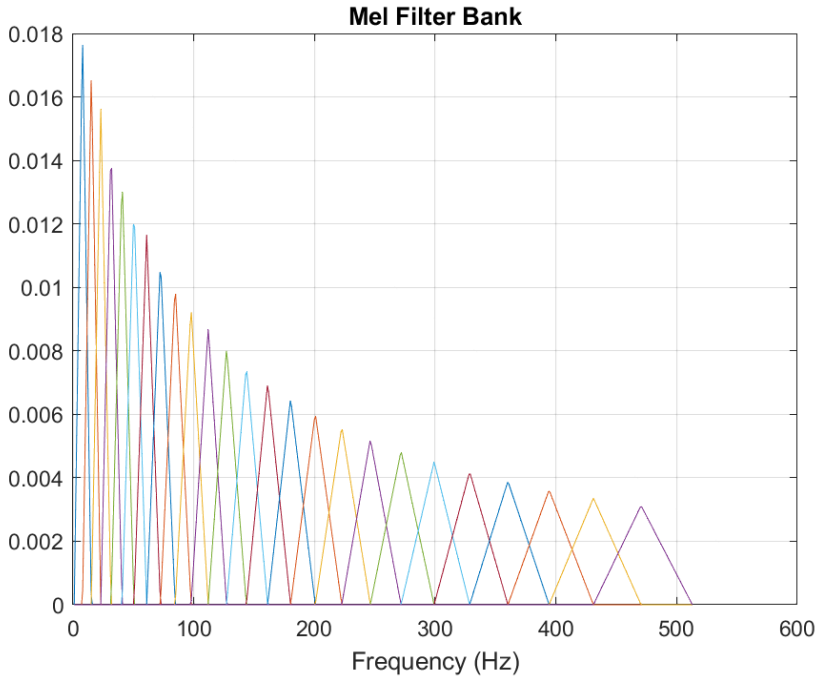


Figure 2.10: Mel Filter-Bank

Where f denotes the physical frequency in Hz, and f_{Mel} denotes the perceived frequency Both the frequency domain and the time domain are capable of supporting filter banks. Filter banks are typically built in the frequency domain for MFCC calculations. On the frequency axis, the centre frequencies of the filters are typically uniformly spaced. However, the warped axis, in accordance with the non-linear function provided in equation (5), is implemented in order to match the human ear's perception. (97) The filter bank typically consists of overlapping triangular filters.(98) Figure 2.10 shows the generated Mel filterbank for 1024 point FFT transform, where number of filters is 25, minimum frequency is 0 Hz, maximum frequency is 4000 Hz and sampling frequency is 8 kHz. The algorithm of MFCCs generation creates the filterbank before all processing is done because filterbank parameters are

constant. The frequency spectrum of the signal(i,e, X(k) from equation (4)) is multiplied with the filter bank to obtain Mel frequency spectrum. Thus mapping the power-spectrum of the signal on to the Mel scale.

Discrete Cosine Transform (DCT): The vocal tract is smooth and hence there is a tendency for adjacent bands' energy levels to correlate. The DCT is used to create a set of cepstral coefficients from the transformed Mel frequency coefficients. The Mel spectrum is typically displayed on a log scale before being subjected to DCT. In the cepstral domain, this produces a signal with a quefrency peak that corresponds to the signal's pitch and a number of formants that represent low quefrency peaks. Since the first few MFCC coefficients constitute the majority of the signal information, the system can be made robust by extracting only those coefficients while ignoring or truncating higher-order DCT components.

Finally MFCC is calculated as

$$c(n) = \sum_{m=0}^{M-1} log_{10}(s(m))cos(\pi n(m - 0.5)/M) \tag{2.14}$$

n=0, 1,2....C-1. where c(n) are the cepstral coefficients, and C is the number of MFCCs. MFCC systems use only 8–13 cepstral coefficients. The zeroth coefficient is often excluded since it represents the average log-energy of the input signal, which only carries small amount of speaker-specific information. (97)

The log Mel spectrum is changed back to time in this last phase. For the specified frame analysis, the cepstral representation of the speech spectrum gives a good representation of the local spectral features of the signal. The Discrete Cosine Transform (DCT) can be used to translate the Mel spectrum

coefficients into the time domain because they are real numbers, as is their logarithm. The log Mel spectrum is transformed back to time in this final stage. The Mel Frequency Cepstrum Coefficients (MFCCs) are the outcome. The Mel Coefficients are transformed back into the time domain using the Discrete Cosine Transform (99).

Deltas and Delta-Deltas: Deltas and Delta-Deltas are also known as differential and acceleration coefficients. Only the power spectral envelope of a single frame is described by the MFCC feature vector, but it would seem that speech would also contain information about dynamics, i.e., the trajectory of the MFCC coefficients over time. It turns out that adding the MFCC trajectories to the original feature vector after computing them significantly improves ASR performance. The benefit of Delta features is that they are used to represent the temporal information. To calculate the delta coefficients, the following formula is used.

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2}, \tag{2.15}$$

where d_t is a delta coefficient from frame t computed in terms of the static coefficients c_{t-n} to c_{t+n}. n is usually taken to be 2. By taking the derivative of Delta features, Delta-Delta features are extracted. (100)

Wavelet Transform

The decomposition of a signal into a collection of basis functions made up of contractions, expansions, and translations of a mother Wavelet function $\psi(t)$, referred to as the Wavelet Transform (WT). The Wavelet reduction method is based on the multi-resolution signal decomposition method devel-

oped by (101).The Wavelet Transform is an efficient noise reduction technique. It is employed to decompose a signal into scaled and shifted representations of specific Wavelets. There are Wavelet families that can be used. Two filters are used in the decomposition process, convolving the input signal and subsequently decimating it into detail coefficients (high frequency component) and approximation coefficients (low frequency component). The procedure is carried out repeatedly until a final level is attained. At each level, the approximation coefficient is used to decompose the original data n times. A graphic representation of the breakdown process at each stage is provided in figure 2.11.

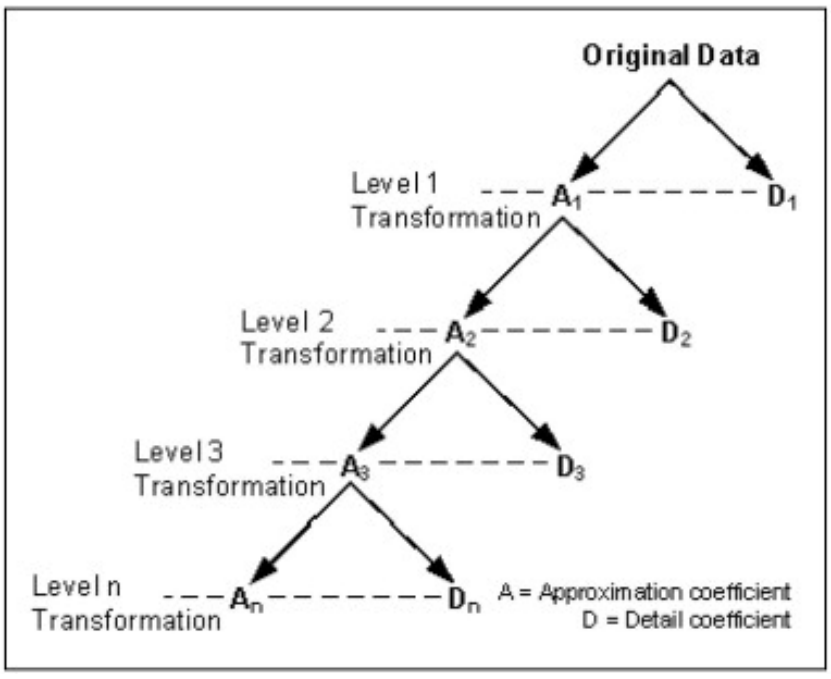


Figure 2.11: Wavelet decomposition (9)

The Wavelet Transform technique is employed for both temporal and frequency domain analysis. At different frequency bands, the original signal is divided into a large number of components. The wavelet Transform of a

signal x(t) is defined as:

$$X(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} x(t)\psi\left(\frac{t-b}{a}\right) dt \tag{2.16}$$

Where the parameters and variables are defined as follows:

t**:** The time variable of the signal.

$x(t)$**:** The input signal being analyzed in the time domain.

a**:** The scaling parameter ($a > 0$). It controls the dilation (stretching) or contraction (compressing) of the wavelet. Large values of a correspond to a stretched wavelet (low frequencies, broad time window), while small values correspond to a compressed wavelet (high frequencies, narrow time window).

b**:** The translation parameter. It represents the time shift, moving the wavelet along the time axis to analyse local characteristics of the signal at different time instances.

$\psi(\cdot)$**:** The mother wavelet, which serves as the prototype transform function. The term $\psi^*(\cdot)$ denotes its complex conjugate (used if the mother wavelet is complex-valued).

$X(a,b)$**:** The resulting wavelet coefficients, representing the correlation between the signal and the shifted, scaled mother wavelet.

In this thesis, the Wavelet Transform serves as a dual-purpose mechanism to overcome the temporal averaging limitations of traditional spectral

analysis by synthesizing Mel Frequency Wavelet Coefficients (MFWCs) (Appendix A), while also providing data reduction to manage the high dimensionality of time-domain reservoir processing (Appendix B).

2.7. Audio Signal Analysis Using Reservoir Computing

Audio processing is fundamentally a challenge of temporal modelling because sound is never a static data point rather, it is a sequential transmission of vibrations. Within the landscape of artificial intelligence and digital signal processing, Reservoir Computing has established itself as a transformative framework for managing temporal data. While conventional deep learning models—such as those used for high-end generative tasks—rely on exhaustive, multi-layer training (102), they often face significant hurdles when applied to real-time audio. Traditional Recurrent Neural Networks are frequently utilized for their ability to remember sequential information. However, the training of these networks is notoriously inefficient. The requirement to optimize every internal connection through back-propagation results in high energy consumption and the persistent risk of gradient-related instability (103). RC decisively resolves these inefficiencies by maintaining a fixed internal architecture, shifting the entire learning burden to a single output layer. This structural choice ensures that the system is not only faster to deploy but inherently more stable for processing the continuous, fluctuating streams of data that define modern audio.

In the literature, Reservoir Computing has been prominently applied to

audio processing tasks that involve temporal modelling and classification, leveraging its core strength in processing sequential data with minimal training complexity. A canonical example is speech recognition. When we speak, our voices produce a raw wave of sound, but this wave is often too messy for a computer to handle directly. Usually, we simplify the sound into a map called Mel Frequency Cepstral Coefficients, which highlights the parts of the sound that the human ear cares about the most. The raw or preprocessed audio signal (e.g., Mel-Frequency Cepstral Coefficients) is fed into the reservoir's recurrent network of randomly connected nodes (99). The high-dimensional and transient dynamics of the reservoir nonlinearly project the temporal audio patterns into a spatial state, making it easier for a simple linear readout layer to classify phonemes or words. This approach avoids the computationally expensive process of training the entire recurrent network, as only the final readout layer is trained, often with ridge regression. Research has demonstrated successful applications in isolated digit recognition and speaker identification, showcasing RC's effectiveness even with modestly sized reservoirs.

Building on these principles, RC has been applied to more complex auditory scene analysis problems, including sound event detection and classification. Unlike speech, general audio scenes comprise overlapping events from diverse sources. Techniques like Deep Echo State Networks stack multiple reservoir layers to create a hierarchical temporal feature extraction pipeline (104). In this layered system, the first reservoir handles the quick, tiny details of the sound. It then passes its information up to the next reservoir, which looks for patterns that take a bit longer. This stacking method

makes Reservoir Computing model long-term dependencies in speech more effectively than shallow reservoirs (105)

A particularly promising research direction is the hardware implementation of reservoirs for ultra-low-power, always-on audio processing. The fixed nature of the reservoir layer makes it ideal for implementation on neuromorphic hardware, FPGAs, or using memristor (106) arrays. These physical reservoirs can process audio signals with extreme energy efficiency, enabling applications like keyword spotting on edge devices without constant cloud connectivity. Research in this area focuses on mapping the reservoir dynamics directly onto physical substrates (73), where the computation is performed using the inherent properties of the hardware, leading to massive gains in speed and power reduction (14).

Overall, Reservoir Computing offers an appealing framework for tackling challenging non-linear audio processing problems and may be used as an extra tool in a variety of signal processing and classification applications (107). The network is essentially temporal due to recurrent reservoir connections, making it particularly advantageous for audio processing. In comparison to other non-linear models, they frequently have a significantly lower computing complexity and are fairly simple to use. They are also resistant to changes in parameter values (108).

2.7.1. The Reservoir Computing Framework for Acoustic Analysis

A strong temporal component can be seen in many difficult computing problems (109). Recurrent neural networks (RNNs) are well adapted to handle

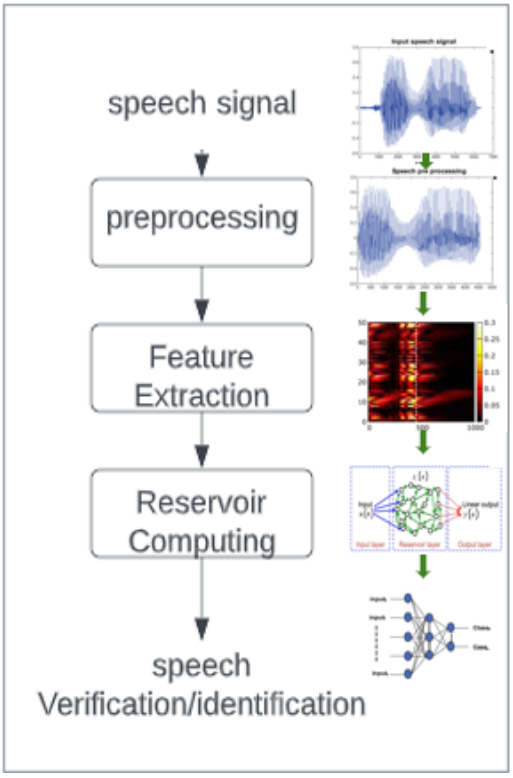


Figure 2.12: Audio signal processing using Reservoir Computing

pattern classification and regression applications with a high temporal component. However, the gradual convergence of the majority of present learning rules makes their implementation difficult. The use of reservoirs provides a convenient solution to this issue. Reservoir Computing (RC) is increasingly being used as a conceptually simple yet powerful method for temporal processing (110). Reservoirs are increasingly being used for audio applications like acoustic modelling, automatic speech recognition, etc. A reservoir system that can identify the fundamental sounds of continuous speech has been developed in a short amount of time. The system already performs at a cutting-edge level, but there is indication that there is still room for major improvement

The figure 2.12 shows the steps involved in speech signal processing using a reservoir computer. Noise is removed after pre-processing the raw speech samples. The reservoir receives extracted and processed features as inputs.

Pre-processing

The raw audio signal needs to be pre-processed. It is important to detect signal in the presence of noise. A speech signal consist of voiced part, unvoiced part and silence part in them. It is important to remove silence part from signal. In most of speech processing usually a Short frame is analysed at a time so we need to window the signals to short processable frame, all these processing is generally referred to as pre-processing stage.

Feature Extraction

The acoustic features are contained in frequency domain rather than amplitude of time varying signals. As a result, to extract features from the audio signals, several frequency decomposition techniques are typically applied. By altering the parametric representation of the speech waveform at the lowest data rate, features are extracted for further processing and analysis. Numerous biologically realistic signal processing algorithms have been documented in the literature, including MFCC (Mel Frequency Cepstral Coefficient), Lyon Passive Ear, and Inner Hair Cell models, etc (92) (98). A detailed comparison is provided by (111), (109). The purpose of feature extraction is to illustrate a speech signal by predetermined number of components of signal. This is because all information in acoustic signal processing is cumbersome to deal with and some information is irrelevant to the task

performed on signal. The method of feature extraction involves converting the speech waveform into a parametric representation at a significantly lower data rate for further processing and analysis.

Reservoir Dynamics

Reservoirs are made use for implementing the application like speaker verification, Speaker identification, speech classification etc. RC offers a paradigm for using a dynamical system as a reservoir to compute complex functions. The RC framework may be applied in many of the same circumstances as recurrent feed-forward neural networks and is perfect for the processing of dynamic, temporal real-time inputs (18). Additionally, RC has benefits like fault tolerance and the ability to learn from a time-varying input through a dynamic reservoir that results in a higher-dimensional representation of the signal through non-linear transformation, where different points on the reservoir are measured and linearly combined to reproduce any output signal (112).

Separation and approximation are two crucial characteristics that must be included in a stable reservoir model. The readout function's capacity to categorise the state vectors sampled from the reservoir is referred to as a readout function's approximation property (30). The ability to distinguish between two different input sequences is referred to as the separation property. This is significant since a decent classification accuracy requires the readout network to be able to distinguish between two distinct input patterns. The readout network will not be able to distinguish between the two patterns and will therefore be unable to categorize which pattern belongs to

a given class if the output responses of a reservoir for two different inputs are the same. The size of the reservoir, node type, likelihood of local and global connections, and other significant elements all play a role in the vast design space for building a stable reservoir. Other elements, such as input features that are utilised to disturb the reservoir, in addition to the intrinsic dynamics of the reservoir, affect a reservoir's overall performance. A stable reservoir's short-term memory capacity (72), which depends on a variety of variables including membrane threshold, reset voltage, and leaky integration, is another important component. The advantage of bigger reservoirs is that they increase the dimensionality of the reservoir states and data becomes more visible to the readout neurons (113).

2.8. Related Work

The application of non-linear state models to temporal signal analysis has a long history, particularly when using sequence-based models to handle the high variability found in human speech. (114). This empirical evaluation reveals a clear distinction between heavy, iterative gradient-based deep architectures and resource-frugal, bio-inspired dynamical substrates.

Traditional architectures establish high-accuracy baselines for sequence modelling tasks but remain constrained by severe computational overheads. Multi-layer Convolutional Neural Networks (CNNs) and deep Residual Networks (ResNets) are frequently applied to speech classification by handling hand-crafted spectrogram configurations (102), (115). However, their massive parameter profiles - often scaling between 2 and 20 million trainable connections - require substantial gradient update cycles during training.

To overcome static window constraints, Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks are widely used to maintain long-term contextual continuity (116). Yet, optimizing these recurrent connections through standard Backpropagation Through Time (BPTT) introduces vanishing and exploding gradient instabilities, alongside high memory bottlenecks that make real-time, low-power edge deployment difficult.

To lower these active training demands, neuromorphic computing paradigms shift the processing mechanism toward asynchronous, event-driven dynamics. Bio-inspired structures like Spiking Neural Networks (SNNs) and Liquid State Machines (LSMs) use randomly connected internal topologies where weight transformations are restricted to a localized or linear readout layer (109), (1). While these models achieve highly energy-frugal operating profiles on edge devices, they often experience a precision penalty when forced to handle continuous acoustic transients via non-differentiable spiking transformations (117).

To bridge this precision-efficiency gap, Reservoir Computing (RC)—formalized via Echo State Networks (ESNs) and structured filter networks—isolates the entire optimization process to a simple linear readout layer, leaving the high-dimensional internal recurrent substrate fully fixed (52), (108). In standard software frameworks, hierarchical and deep reservoir networks operating over pre-computed Mel-Frequency Cepstral Coefficients (MFCCs) achieve high classification performance (118), (105).

Recent advancements have expanded the general research landscape by implementing these dynamical properties directly within unconventional hardware media. For example, all-optical and optoelectronic delay-loop reservoirs leverage light-wave modulation through a single physical non-linear node

with delayed feedback, significantly expanding processing speeds for acoustic tasks (119), (120). Similarly, physical hardware implementations using dynamic memristor networks, superconductor devices, and mechanical strain within piezoelectric (PZT) substrates demonstrate that intrinsic material dynamics can passively extract complex, non-linear speech components without traditional digital domain transforms (106), (121), (122).

Our time-domain Reservoir based framework builds on these foundations. Rather than relying on rigid, resource-intensive Fourier, Mel-filterbank, we show that software-based reservoirs can be configured to perform intrinsic time-domain feature convolutions and mimic acoustic cepstral outputs natively, matching the accuracy of multi-layer monolithic pipelines while drastically cutting down operational overhead.

A comprehensive mapping of these modern architectural paradigms, evaluated across Ti-46 and Audio MNIST datasets, is summarized in Table 2.1.

Method	**FFT Required**	**Filterbank**	**Parameters**	**Time Domain**	**Parallel Filters**
CNN / ResNet	Yes	Explicit	High	No	Yes
LSTM Networks	Yes	Explicit	High	No	No
AudioNet	Yes	Explicit	High	No	Yes
Liquid-SNN / LSM	No	Spiking	Moderate	Yes	Implicit
Piezoelectric Cube	No	Physical strain	Low	Yes	No
Standard RC	Yes	Explicit	Low	No	Yes
Reservoir method (Mimic)	No	Reservoir dynamics	Very Low	Yes	Implicit
Reservoir method (Convolution)	No	Reservoir dynamics	Very Low	Yes	Implicit

Table 2.1: Complexity and Architectural Comparison of Feature Extraction Methods

2.8.1. Key Exploration of RC in Audio Processing

We are focusing on current exploration areas in Reservoir Computing. These exploration paths aim to move us away from traditional, power-hungry software toward systems that are as lean and adaptive as biological ears.

Breaking the Feature Bottleneck with Resource-Efficient RC: Recent research, particularly in the context of Neuromorphic Computing and Physical Reservoir Computing (PRC), has identified the feature extraction bottleneck as the key hurdle for true resource efficiency. The goal is now to utilize the inherent non-linear dynamics of the reservoir to perform the feature extraction function itself.

Direct Temporal Processing in Hardware: Modern work in physical RC, leveraging technologies like dynamic memristors or photonic systems, aims to implement the RC core in hardware that naturally processes signals in the temporal domain (123). These systems, by their physical nature, perform the necessary non-linear transformation directly on the input signal, bypassing the need for digital FFT or DCT computation entirely (124).

Deep and Hybrid RC: To boost performance and simplify implementation, researchers are investigating Deep Reservoir Computing (DRC), where multiple reservoirs are stacked or connected in sequence. This approach maintains the training simplicity of RC while enhancing representational power, and is actively being explored in high-speed platforms like photonic systems for speech recognition (125).

Efficient Multi-Tasking: The inherent fixed-weights architecture of the RC reservoir is uniquely suited for multi-task learning, where a single, costly transformation (the reservoir) is shared across multiple linear readouts trained for distinct tasks (50). This framework minimizes the total parameter count and training time compared to training multiple specialized deep neural networks.

Building on these exploration areas, we examine how our specific approach

redefines the standard processing pipeline.

2.9. The Audio Processing Bottlenecks and the Reservoir Computing Alternative

The efficient processing of sequential, time-varying data is a cornerstone of speech and audio technology. For decades, the dominant paradigm has relied on a two-stage pipeline: first, the conversion of raw audio into a hand-engineered, perceptually-motivated feature representation. Second, the application of a statistical or neural model for classification or regression. While effective, this paradigm suffers from inherent computational bottlenecks and information loss, which become critical in low-power, edge-computing scenarios. Deep neural networks, particularly Recurrent Neural Networks (RNNs) and their variants (LSTMs, GRUs), possess strong sequence modelling capabilities but, their extensive training time and computational cost necessitate the exploration of simpler, yet effective alternatives. Reservoir Computing (RC), underpinned by the Echo State Network (ESN) (51), emerged as a potent solution, significantly reducing training complexity by keeping the large recurrent layer fixed and training only the linear output layer. This work traces the evolution from traditional feature extraction to modern end-to-end deep learning, culminating in the argument for Reservoir Computing (RC) as a uniquely efficient framework capable of unifying feature extraction and temporal modelling.

2.9.1. Traditional Feature Dependence in Audio Processing

For many years, the strong performance of speech and audio technology has been based on the fundamentals of human auditory perception, which has resulted in a well-established reliance on manually created, frequency-domain features. The standard was established by the groundbreaking work that led to Mel Frequency Cepstral Coefficients (126), (47), which focused on the non-linear connection between perceived frequency and real frequency (the Mel scale) in order to capture acoustic information pertinent to human hearing. Though useful for invariant acoustic modelling, this multi-stage processing pipeline is a significant source of computing complexity and information loss, which hinders modern low-power applications (45).

The MFCC process is inherently complex. In this context, computational complexity refers to the high execution cost and resource demands measured by the sheer volume of mathematical operations —specifically floating-point operations— and memory accesses required per unit of time. This repetitive cycle of domain transformations creates a substantial computational burden; for continuous audio, the FFT must be calculated thousands of times per second. Recent research into hardware acceleration indicates that these initial stages—the FFT and filter bank—often account for more than half of the total processing time. Consequently, this high operational complexity manifests as increased processing latency and power consumption, necessitating sophisticated hardware workarounds and memory optimization to make feature extraction feasible on resource-constrained edge devices.

Beyond the cost of computation, the traditional methodology also causes

an irreversible loss of temporal fidelity. When a signal is windowed and transformed into the frequency domain, precise time-localized features are damaged by nature. This makes it nearly impossible to resolve rapid auditory events that are crucial for tasks requiring precise temporal alignment. Because the features supplied to the subsequent sequence models are already temporally averaged, the resolution of the entire system is fundamentally limited by the very first steps of the processing chain.

The traditional RC applications treated the input features as externally generated, thereby inheriting the high cost and temporal information loss associated with time-to-frequency domain conversion, which is a major constraint in high-speed and neuromorphic hardware implementations (48). This reliance on pre-computed spectral features essentially forces a dynamic system to ignore the continuity of the acoustic waveform.

The Reservoir acts as a continuous filter that retains the nuances of the raw signal, bypassing the need for rigid, power-hungry transformations. Our work specifically addresses critical gap by demonstrating that, ESN can function as an intrinsic feature extractor. This methodology is distinct because it is designed to achieve the computational efficiency and temporal fidelity of the best modern neuromorphic RC systems, but without requiring specialized hardware, thus offering a universally applicable alternative to the traditional feature dependence.

2.9.2. End-to-End Audio Processing

This feature dependence has recently been challenged by the deep learning community. The concept of end-to-end speech processing (127) has

gained prominence, where models attempt to learn the feature representation directly from the raw audio waveform, bypassing all conventional preprocessing. Early attempts showed promise using large CNNs followed by LSTMs (128). More recent work, leveraging advanced architectures, has confirmed that features learned directly from the waveform can outperform MFCCs, particularly in multi-task scenarios (129).

However, this shift primarily relocates the computational burden rather than eliminating it. The models required to learn these complex transformations directly from raw data are typically extensive, demanding significantly more training data, parameters, and energy than traditional systems (130). Additionally, these end-to-end deep learning pipelines often function as uninterpretable black boxes where continuous raw audio waveforms are fed into millions of heavily optimized parameters. While the final classification output may be accurate, tracing how or why the network extracted specific phonemes or speaker identities from the raw audio waves remains structurally obscured. Consequently, there remains a clear need for a methodology that captures the rich dynamics of raw audio without the complexity and high training overhead associated with deep, monolithic network architectures.

Our work attempts to bridge this gap by offering the simplicity and efficiency of the RC framework while overcoming the significant temporal and computational constraints of conventional feature dependence. By harnessing the non-linear dynamics of the ESN in the time domain, we attempt to develop features that preserve temporal integrity, offering a true alternative to both the complex MFCC pipeline and the resource-intensive large-scale end-to-end deep learning models.

2.9.3. Novelty of the Time-Domain RC Feature Extractor

Our work presents the Reservoir Computing Time-Domain Feature Extractor (RC-TDFE), which is a significant divergence from current approaches in both modern deep learning and conventional signal processing. By applying the Echo State Network architecture to function as an end-to-end feature generator on raw audio waveforms, we directly address the shortcomings of traditional processing. In order to overcome the two constraints of computing expense and temporal inaccuracy, this strategic change makes use of the inherent qualities of RC.

- **Direct and Intrinsic Time-Domain Operation:** By removing the requirement for external signal processing modules, reservoir-based time domain feature extraction creates a pipeline that is genuinely simpler. Our approach entirely avoids all intermediary domain conversions (FFT, filter-banks, DCT), in contrast to approaches that depend on the computationally costly and temporally distorting time-to-frequency-to-cepstrum transformations (131). The time-domain input is quickly converted into a high-dimensional state space by the fixed, non-linear reservoir, where the activations operate as the intrinsic audio features. The speed, low latency, and low power required for effective neuromorphic applications are directly obtained by this core simplification (53), (124).

- **Irreversible Temporal Information Loss Mitigation:** The RC-TDFE naturally avoids the temporal blurring brought on by the windowing and frequency transformation processes because it only works

in the time domain. In contrast to traditional feature sets, this guarantees the mitigation of irreversible temporal information loss (46). For complicated tasks like phoneme border identification and forced alignment, the reservoir's fading memory characteristic enables it to capture important temporal context while preserving high temporal fidelity.

- **Neuromorphic Alignment and Resource Efficiency:** This approach is perfectly aligned with the emerging paradigm of neuromorphic hearing. The ESN, as a biologically inspired recurrent mechanism, efficiently captures acoustic information using only simple linear regression for training the readout layer. This dramatically reduces the burden of training enormous deep learning models (129). By transforming the most complex task (feature generation) into a highly efficient, fixed computation, our work offers a lightweight, scalable, and resource-efficient foundation for next-generation, real-time auditory processing hardware.

The work successfully closes a critical gap in the literature by demonstrating that Reservoir Computing can serve a dual role: not just as an efficient sequence modeller (its traditional application), but also as a lightweight, intrinsic audio feature extractor. This is achieved by configuring the Echo State Network to operate end-to-end on raw audio waveforms.

This integration produces a processing pipeline that is extremely streamlined, low-latency, and resource-efficient. The research goes beyond simply optimising current techniques by removing the key bottlenecks of traditional methods by enabling the Reservoir Computing based Time-Domain Feature

Extractor to produce features directly. In particular, it avoids MFCCs' computationally costly and temporally distorted time-to-frequency conversions (such as the FFT). For tasks that call for precise alignment, this guarantees the preservation of temporal integrity. In the conclusion, the study develops a new end-to-end paradigm for audio signal processing that is inherently appropriate for neuromorphic, energy-efficient applications.

3 Dataset and Experiments

3.1. Dataset

We conducted our preliminary experiments using the TI-46 corpus, subsequently validating and substantiating our findings through the significantly more extensive and diverse Audio MNIST dataset. This tiered approach allowed for initial model tuning on a controlled, high-fidelity dataset before testing scalability and robustness against a larger variety of acoustic profiles.

The TI-46 Corpus

The TI-46 corpus is a foundational speech dataset collected by Texas Instruments in 1980 and distributed through the Linguistic Data Consortium (LDC). It contains approximately five hours of isolated spoken words recorded by 16 speakers (eight males and eight females). The vocabulary is comprised of 46 words: the digits 0–9, ten operational control words (e.g., 'enter', 'erase', 'start', and 'stop'), and the 26-letter English alphabet.

The data was originally captured in a sound-isolated acoustic enclosure at a 12.5 kHz sampling rate with 12-bit quantization. Because of its high signal-to-noise ratio and clean labelling, TI-46 is a primary benchmark for evaluating speaker-dependent and speaker-independent recognition algorithms. In our study, it provided a precise environment to establish baseline performance

metrics.

The Audio MNIST Dataset

To ensure the generalizability of our results across broader demographic variations, we utilized the Audio MNIST dataset (132). Inspired by the classic MNIST handwritten digit benchmark, this open-source collection consists of 30,000 audio recordings of English digits (0–9). The dataset represents a significant increase in scale and diversity compared to TI-46, featuring 60 different speakers of various ages, genders, and regional accents, with each participant providing 500 samples.

Audio MNIST is typically provided at a 48 kHz sampling rate, offering a high-density feature set suitable for deep learning architectures. It is particularly effective for training Convolutional Neural Networks (CNNs) using two-dimensional spectrogram representations or Mel Frequency Cepstral Coefficients (MFCCs). The inclusion of detailed speaker metadata allows researchers to perform rigorous cross-validation for both digit classification and speaker identification tasks. This dataset is publicly accessible via Kaggle and the original GitHub repository. By validating our initial findings against this larger corpus, we ensured that our proposed model maintains high accuracy and robustness when faced with the natural variability inherent in human speech.

3.2. Experiments

We start our experiments by analysing the usefulness of reservoir as a classifier. Here we have used Mel Frequency Cepstral Coefficient as the feature for the analysis. This experiment helps us to analyse the usefulness of MFCC as

a feature in audio analysis as it is commonly used feature due to its ability to mimic human audio processing system.(This is detailed in chapter 4)

Following the success of Reservoir Computing (RC) as a classifier, we evaluate RCs' potential as a biologically inspired 'ear' for front-end feature extraction. We investigate whether the reservoir's high-dimensional, non-linear dynamics can autonomously project raw audio into feature spaces analogous to Mel Frequency Cepstral Coefficients (MFCCs), thereby bypassing traditional signal processing overhead.

In first experiment, we design a reservoir to replicate the behaviour of the MFCC coefficients generated by Matlab's standard function. A primary challenge is the high dimensionality of the raw audio signal, which created significant computational constraints. To address this, we implement a windowing mechanism to segment the non-stationary speech into quasi-stationary frames. (This is detailed in chapter 5)

In our second approach, we model a reservoir to perform direct convolution between the raw audio signal and synthesized time-domain equivalents of a Mel Filter-Bank. This methodology is grounded in the signal processing principle that frequency-domain filtering—the core of the MFCC extraction process—is mathematically equivalent to convolution in the temporal domain. Since reservoirs are inherently dynamical systems optimized for time-series data, we exploit this property to extract spectral energy without the computational overhead of a Fourier Transform.

To implement this, we synthesize the time-domain equivalent of Mel Filter-Banks by analytically decomposing each triangular frequency-domain filter into its constituent spectral components. By precisely parametrizing a

series of sine waves and applying the principle of superposition, we generate complex oscillating waveforms that function as temporal impulse responses. The raw audio signal is then convoluted with these synthesized signals, allowing the reservoir's internal fading memory to capture the evolving spectral characteristics of the speech. This approach effectively transforms the reservoir into a hardware-friendly, real-time auditory processor that preserves the phase and temporal nuances often lost in traditional spectral analysis.

But the biggest challenge of this experiment is data reduction. We have explored a great range of data reduction method. We first tried methods like max/min pooling, mean, peak to peak, absolute max,matrix- dimension reduction methods, Wavelet Transform etc. Among them Wavelet methods showed best performance. But since it takes away the whole essence of our experiment to simplify audio signal processing, we have decided to highlight on simple data reduction methods. Among the simple methods, Peak to peak and absolute max pooling have given better performance. We have been using one reservoir to extract 14 MFCC coefficients by training 14 output layers separately. To improve the performance we are also training 14 individual reservoirs (smaller reservoirs), each tuned individually to produce each coefficient. (This is detailed in chapter 6)

While experimenting for data reduction we have found that a combination of MFCC and Wavelets are best in audio signal analysis. There are many literature that explores the potential of these two features. In the state of art method of extraction of Wavelet based Mel Frequency coefficient, Wavelet Transform is done prior to MFCC part or after MFCC part. In either case the whole step of calculation of both MFCC and Wavelet Transform is done

to obtain the Wavelet Transform based Mel-scaled feature extraction. This makes the whole process complicated. So we develop an approach which, uses the time domain feature extraction method to extract the Mel Frequency Wavelet Coefficient, reducing the method's complexity and increasing its efficiency. The time-domain capability of a Reservoir Computing technique is also made use to improve the performance of the entire system. In MFCC extraction process we use a reservoir to implement the convolution process. here, since there is cosine and sine convolution the all-round classification efficiency reduces if we are using a reservoir for feature extraction. Hence we are only using a reservoir for classification purpose. (Therefore this experiment is included as appendix.A)

Followed by these experiments we have also thought of exploring the potential of Reservoir Computing in audio analysis without a feature extraction stage or in other words feature-free audio analysis.

Building upon our initial benchmarks, we explore the potential of Reservoir Computing (RC) as a framework for efficient audio analysis. A key motivation for this study is the elimination of the traditional feature extraction stage (e.g., MFCC) in favour of feature-free audio analysis. By feeding raw temporal signals—dimensionally reduced through simplified techniques—directly into the dynamical system, we leverage the reservoir's inherent ability to project input sequences into a high-dimensional state space. In this capacity, the reservoir functions as a natural, non-linear temporal filter. We have intentionally utilized simple dimensionality reduction methods, such as absolute max pooling or peak to peak, to preserve the core objective of our study, maintaining a streamlined, efficient processing pipeline that

does not rely on complex pre-processing transformations.

In our first approach, we utilize a standard Echo State Network (ESN) configuration. The (dimensionally reduced) raw audio waveform is injected into a single, large-scale reservoir consisting of randomly interconnected neurons with fixed weights. This method relies on the Echo State Property, where the current state of the reservoir $\mathbf{x}(t)$ is a non-linear function of its previous state and the current input $u(t)$. The primary advantage of this single-reservoir approach is its computational efficiency. Since only the weights of the linear readout layer are trained the system avoids the vanishing gradient problems associated with traditional Recurrent Neural Networks (RNNs). This allows for the rapid processing of the temporal dependencies present in the TI-46 and Audio MNIST digits without the overhead of pre-computed audio features.

To capture the complex, multi-scale temporal dynamics inherent in human speech, we have also explored the Deep Reservoir Computing concept. Unlike the single-layer approach, Deep Reservoir Computers consists of a hierarchy of stacked reservoirs where the state of one layer serves as the input for the subsequent layer. Specifically, we implement the Parallel feature-free Reservoir Computing (PFRC) approach. This architecture functions as a multi-resolution ensemble, processing the raw data through multiple parallel reservoir pathways to simultaneously capture various temporal perspectives. By decoupling the signal across different layers and pathways, the PFRC can effectively represent audio input, providing a more robust and comprehensive analysis of the audio signal. (This is detailed in chapter 7).

A comprehensive discussion and comparative analysis of the methods are

presented in Chapter 8.

4 Reservoir Computing for Audio Classification

This chapter evaluates the capability of Reservoir Computing (RC) to classify audio signals using features extracted through conventional signal processing methods. By establishing this baseline, we aim to determine how effectively the reservoir's dynamical mapping can categorize acoustic data when provided with standard pre-processed inputs.

The classification stage serves as the final component of our end-to-end Reservoir Computing (RC) architecture. Following feature extraction—conducted either by a time-domain RC reservoir (RC-1) or a Matlab-based MFCC function—the resulting data is processed by a dedicated RC Classifier (RC-2) to produce the final output. This architecture is designed to validate the efficacy of the generated features by performing the core tasks of audio signal processing: digit recognition and speaker recognition.

4.1. Classification Architecture and Feature Inputs

Our investigation models the classification stage as the final decision-making layer, confirming that the entire pipeline can be unified within the RC framework. The classifier reservoir operates in a supervised manner, where its in-

put is the feature vector generated by the preceding stage—either the novel RC based features (for system validation) or conventional MFCC features (for baseline comparison).

The classifier is utilized to perform two crucial validation tasks, reflecting the dual information embedded in speech:

1. **Digit Recognition:** The classifier reservoir maps the feature vector to one of ten phonetic classes (the digits 0 to 9), validating the feature set's fidelity to phonetic content

2. **Speaker Recognition:** The classifier reservoir maps the feature vector to the class representing the specific speaker, validating the feature set's fidelity to individual acoustic characteristics.

4.2. Experimental Setup and Optimization

To provide a thorough and objective assessment in comparison to the recognised classification benchmarks, the experimental configuration for the classifier reservoir was standardised. Figure 4.1 illustrates the classification pipeline where features are input into the reservoir.

The input preparation involves calculating the MFCC of each digit utterance and then concatenating these features of the utterance into a single, continuous input sequence. This sequence is fed to the reservoir, allowing the network to utilize its temporal memory to differentiate between sequential information embedded within the features. For instance, in cross-validation setups (like the 5-fold cross-validation used on smaller corpus), the input matrix is structured to maximize the network's exposure to diverse training

samples.

4.2.1. Reservoir Parameters

The specific parameters for the classifier reservoir are tuned for optimal performance in sequence classification:

- **Node Count:** The reservoir is composed of **400 nodes**, providing sufficient non-linear dimensionality for complex state mapping while maintaining computational tractability.
- **Spectral Radius (ρ):** The spectral radius of the recurrent weight matrix ($S_x.W_{\text{res}}$) is scaled precisely to $\rho = \mathbf{0.8}$. This setting scales the largest eigenvalue of the network to govern global stability and sustain the echo state property while preventing chaotic state divergence.
- **Sparsity:** A sparsity of **80%** is implemented (connections established with a 20% probability). This promotes diverse, decoupled temporal dynamics within the network while conserving physical resource requirements.
- **Leakage Rate (α):** A leakage rate (S_x.leak) of $\alpha = \mathbf{0.3}$ is utilized. This parameter controls the exponential decay rate of the nodes, dictating the balance between the network's short-term memory capacity and its fast responsiveness to incoming input signals.
- **Washout:** A **washout period of 50 time steps** is applied. This initial portion of the state matrix is discarded during training to ensure that the linear readout utilizes only stable, fully developed internal trajectories, eliminating initialization transient effects.

- **State Update Dynamics:** The discrete-time state transitions of the leaky Echo State Network are governed by the following non-linear activation equation:

$$X(i+1) = ((1 - S_x.\text{leak}) \cdot X(i)) + S_x.\text{leak} \cdot \tanh\left(\left(X(i) \cdot S_x.W_{\text{res}}\right) + \left(u(i) \cdot S_x.W_{\text{in}}\right)\right) \tag{4.1}$$

  where $X(i)$ is the internal reservoir state at time step i, $u(i)$ is the input vector, $S_x.W_{\text{in}}$ represents the input weight matrix, and $S_x.W_{\text{res}}$ denotes the recurrent reservoir weight matrix.

- **Input Weight Scaling:** To guarantee bounded, balanced input scaling, the raw input weight matrix W_{in} is normalized and scaled symmetrically into the $[-1, 1]$ range via the min-max operation:

$$S.W_{\text{in}} = \left(\frac{W_{\text{in}} - mn}{mx - mn} \cdot 2\right) - 1 \tag{4.2}$$

  where mn and mx denote the minimum and maximum boundaries of the input weight matrix.

This standardized configuration of the classifier reservoir provides the essential benchmarking platform. The final evaluation of our entire methodology hinges on demonstrating that the features generated by the novel reservoir based feature extractor can drive this classifier reservoir to achieve performance metrics that are competitive with or superior to those achieved using conventionally extracted MFCC features.

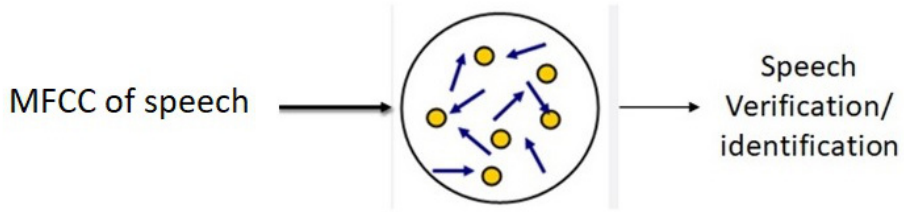


Figure 4.1: Reservoir is used as a classifier after pre-processing using MFCC

4.3. Experimental Methodology

This section details experimental methodology and analysis employed to determine the accuracy and reliability of the Reservoir Computing (RC) system when operating as a classifier reservoir. The primary goal was to obtain statistically robust performance metrics for both digit recognition and Speaker Recognition across two diverse datasets: the controlled TI-46 dataset and extensive Audio-Mnist dataset.

4.3.1. Methodology for Statistical Robustness

We used a strong two-tiered validation approach to guarantee that the reported classification accuracy was statistically reliable and reduced potential bias:

1. **5-Fold Cross-Validation:** We apply the standard 5-fold cross-validation technique. Four of the dataset partitions are used for training, while the remaining one is kept aside for blind testing, therefore the classifier reservoir is trained five times. Importantly, the classifier is evaluated on the complete dataset following each training iteration in order to evaluate the model's full capabilities and potential for overfitting. As

a result, we are able to compute performance measures for testing and training accuracy independently.

2. **Mitigation of Random Seed Effects:** The entire 5-fold cross-validation process is methodically repeated ten times, each time using a different random seed for reservoir construction, in order to lessen the impact of the random initialisation of the reservoir's internal weights. For both training and testing accuracy across each classification task, this yields a total of 50 independent performance measures (5 folds $\times$ 10 repeats).

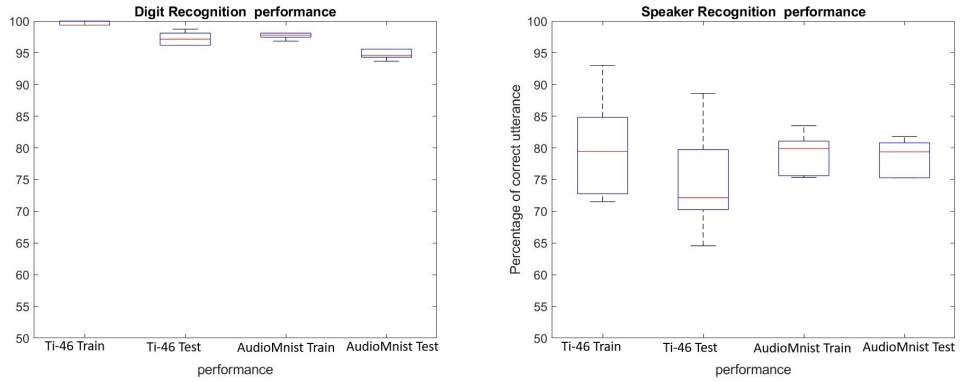
4.4. Performance Measurement and Visualization

For both the training and testing stages, the main performance metric—the percentage of accurate utterance—is gathered from the total of 50 experimental runs for each task and dataset. Box plots are then used to assemble and visualise these statistics.

Figure 4.2 displays box plots showing the reservoir's training and testing performance as a classifier. These visualisations allow for immediate interpretation of the system's stability and generalisation capability:

The detailed analysis of these box plots confirms that the classifier reservoir achieves high accuracy and stability across both the digit and speaker recognition tasks, validating the overall effectiveness of the RC-based feature generation and classification pipeline.

It is important to note that since the classification task involves 10 distinct classes, the baseline for a random guess is defined at 10%



(a) Digit Recognition performance (b) Speaker Recognition performance

Figure 4.2: Performance of Reservoir as a classifier

Experiments	Digit Recognition				Speaker Recognition			
	Ti-46		AudioMnist		Ti-46		AudioMnist	
	Train	Test	Train	Test	Train	Test	Train	Test
RC as classifier	100	99.37	98.24	98.09	79.43	72.15	79.90	79.38

Table 4.1: Summary of the result for Digit and Speaker Recognition using reservoir as a classifier

Models	**Accuracy (%)**
LSM (109)	94.0
Liquid-SNN (133)	77.7
Reservoir Computing (MEMS)(134)	78
Reservoir- as classifier(method in the chapter)	99.36
Reservoir-based(Reservoir MFCC-Mimicking)	61.22
Reservoir-based(Time convolution)	91.82

Table 4.2: Comparison of Performance of models with Ti-46 dataset for digit recognition

The tables 4.2 and 4.3 shows a comparison of the performance of different audio signal processing methods using Ti-46 and Audio-Mnist datasets respectively for digit Recognition.

Models	Accuracy (%)
CNN (135)	98.6
LSTM(135)	97.02
AudioNet(Deep-NN) (132)	92.53
Liquid-SNN (133)	82.65
Reservoir- as classifier(method in the chapter)	98.08
Reservoir-based(Reservoir MFCC-Mimicking)	60.82
Reservoir-based(Time convolution)	70.83

Table 4.3: Comparison of Performance of models with Audio-Mnist dataset for digit recognition.

From figure 4.2, and tables 4.2 and 4.3, we can see how well the reservoir, which uses the MFCC Matlab function performs. This boxplot clearly demonstrates the reservoir's capacity to carry out classification and regression tasks with greater accuracy. It achieves a high score on the Ti-46 dataset and on the more challenging Audio-MNIST dataset. More significantly, it accomplishes this feat with a fraction of the computational complexity typically associated with high-performance audio classifiers, underscoring its efficacy and practical potential. A comprehensive discussion and comparative analysis of the methods are presented in Chapter 8, table 8.1 and table 8.2.

4.5. Summary and Conclusion

The findings of this chapter confirm that Reservoir Computing a highly efficient alternative for audio signal processing. By establishing this baseline, we have demonstrated that classifier reservoir can categorize acoustic data with exceptional precision when supplied with standard pre-processed inputs. The RC-based classifier matches or exceeds the accuracy of traditional

high-performance models while operating at a fraction of the computational complexity. This underscores the practical potential of RC for real-time applications and edge-computing environments where computational resources are at a premium. Ultimately, this chapter validates the reservoir's dynamical mapping as a sophisticated tool for modern acoustic feature classification.

5

Feature Extraction by Mimicking MFCC Function

Building on the established success of Reservoir Computing as a classifier, this chapter investigates reservoirs' capacity to function as a biologically inspired ear for front-end feature extraction. This study evaluates an alternative approach to MFCC extraction that utilizes Reservoir Computing to streamline the feature extraction process. We explore whether a reservoir's high-dimensional, non-linear dynamics can generate acoustic latent space representations, here specifically aiming to replicate the behaviour of standard Mel-Frequency Cepstral Coefficients.

The conventional MFCC extraction pipeline, despite its high effectiveness, suffers from inherent computational bottlenecks. These inefficiencies are directly caused by the pipeline's mandatory reliance on sequential, resource-intensive time-to-frequency transformations, most notably the Fast Fourier Transform (FFT). The repeated execution of the FFT, along with subsequent filter bank operations, introduces significant processing overhead and latency. This severely constrains the ability to deploy sophisticated speech processing systems on energy-limited edge devices and within low-power neuromorphic hardware where energy efficiency is paramount.

The proposed method harnesses the reservoir's high-dimensional states

to efficiently map raw acoustic data into discriminative feature vectors. RC, specifically implemented using an Echo State Network (ESN), known for its ability to model complex, time-dependent signals efficiently due to its fixed, randomly connected recurrent layer (the reservoir). By exploiting the inherent temporal processing capabilities of these reservoir systems, we propose a method to bypass the need for traditional frequency-domain conversions entirely.

Our core hypothesis is that the complex, non-linear dynamics generated within the fixed reservoir can be taught to implicitly perform the sophisticated digital signal processing required for MFCC generation. Our architecture thus demonstrates how reservoir computers can serve as highly efficient, intrinsic feature extractors while simultaneously maintaining the crucial perceptual fidelity and structural dimensions of standard MFCCs. This is a critical distinction: we are not abandoning the perceptually sound MFCC feature set, but rather eliminating the inefficient computational steps traditionally required to produce them. The objective shifts from calculating the FFT and filter banks to training a simple, linear readout layer to extract the desired feature from the reservoir's rich internal state. We are specifically investigating whether a reservoir can be precisely modelled to implement the functions performed by every stage of the audio signal processing pipeline, effectively condensing the multi-stage DSP pipeline into a single, efficient computation. Our ultimate aim is to model a reservoir as a dedicated feature extractor that operates to mimic the output of the conventional MFCC Matlab function, thereby dramatically simplifying this bottleneck stage of audio signal processing.

We present a complete, real-time audio processing framework built upon the principle of feature mimicking. This methodology directly leverages the inherent temporal processing capabilities of the Reservoir Computing RC system to function as a unified processor. The framework showcases the reservoir's capacity to handle end-to-end signal processing—from raw waveform input directly to a compact feature vector output—with remarkable accuracy.

5.1. MFCC Features with Reservoirs

This critical step is achieved by training the reservoir to precisely mimic the output of a standard feature generator, such as the MFCC function. By training the reservoir's linear readout layer to map the internal recurrent state to the ground-truth MFCC values, the system implicitly learns the complex non-linear transformation required for feature extraction.

Crucially, because the reservoir is mapping the raw time-domain input directly to the MFCC equivalent, the architecture completely bypasses the need for the traditional, multi-stage digital signal processing pipeline. This eliminates the computationally expensive and time-consuming steps, such as the Fast Fourier Transform (FFT), spectral filtering, and Discrete Cosine Transform (DCT). Experimental results rigorously validate that our RC-based approach not only simplifies the computational workflow but also accurately retains the critical acoustic features needed for robust speech recognition. This innovation paves the way for a viable, low-power solution for implementing sophisticated speech processing in resource-constrained environments. For the pre-processing stage, we focused on the MFCC feature set, extracting

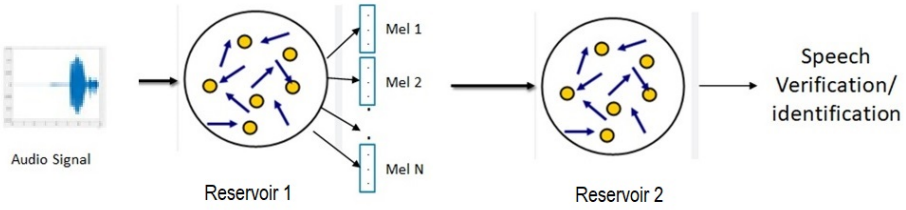


Figure 5.1: Reservoir is used for end-to-end audio processing.

the first 14 MFCC coefficients from the speech signals. These 14 coefficients represent the most essential short-term spectral features of the audio, effectively capturing the characteristic shape of the vocal tract and serving as the common standard for speech signal processing. To ensure the functionality, robustness, and generalizability of our reservoir based time domain feature extraction system, we conducted extensive comparative experiments utilizing two diverse speech datasets: The TI-46 Dataset and Audio-Mnist Dataset.

To confirm correct functionality and better performance of our proposed system, we conducted two primary sets of experiments: one focusing on speaker recognition (identification) and the other on digit recognition (speech content classification). The consistent performance of these experiments confirms that the proposed time-domain RC architecture can effectively replace traditional digital signal processing pipelines. These results demonstrate that reservoir-based feature extraction is a computationally efficient alternative capable of maintaining high classification accuracy without the need for frequency-domain conversion.

5.2. RC-Based MFCC Feature Extraction in Time-Domain

This experiment implements a reservoir-driven architecture designed to map raw audio directly to the Mel Frequency Cepstral Coefficient (MFCC) feature space. By utilizing the inherent dynamical properties of the reservoir, the system aims to replicate the traditional MFCC extraction pipeline without relying on frequency-domain transformations. The focus of this implementation is to validate whether the RC system can maintain the representational accuracy of standardized MFCC coefficients while operating entirely within a time-domain framework.

5.2.1. Reservoir Training and Performance Validation

The initial phase focuses on training the reservoir system to accurately replicate the MFCC algorithm. This is treated as a supervised learning task where the target signal for the reservoir's training is precisely defined by the values calculated by the conventional Matlab MFCC function for the corresponding audio sample. This ensures that the RC system learns to map the raw waveform to a perceptually and structurally correct feature space.

To measure the fidelity and efficiency of the reservoir's performance in replicating the target coefficients, we utilize the Normalized Mean Square Error (NRMSE). The NRMSE quantifies the deviation between the reservoir's output and the analytical target, providing a standardized and reliable measure of prediction accuracy. The normalized mean square error of reservoir 1, which is trained to mimic the MFCC as calculated by Matlab's function,

is displayed in Table 5.1. The NRMSE value clearly emphasizes the ability of a reservoir to mimic MFCC extraction.

MEL	Mel 1	Mel 2	Mel 3	Mel 4	Mel 5	Mel 6	Mel 7
NRMSE-Ti	0.129	0.175	0.085	0.032	0.038	0.024	0.013
NRMSE-AMnist	0.130	0.176	0.083	0.038	0.044	0.026	0.016
MEL	Mel 8	Mel 9	Mel 10	Mel 11	Mel 12	Mel 13	Mel14
NRMSE-Ti	0.018	0.015	0.013	0.015	0.013	0.014	0.009
NRMSE-AMnist	0.018	0.017	0.015	0.019	0.014	0.015	0.011

Table 5.1: NRMSE of Reservoir-1 extracting MFCC, based on 14 mel coefficients

The table 5.1 relies on two distinct dataset to validate the reservoir's capacity to emulate standard feature extraction algorithms : the TI-46 corpus and the more extensive Audio MNIST dataset. Specifically, the target signals (y_{target}) used to train the linear readout layer for table 5.1 are precisely mapped in a supervised manner from the raw audio files to their corresponding frequency-domain coefficients calculated via the conventional Matlab MFCC function. The evaluation tracks the Normalized Mean Square Error (NRMSE) across these datasets to ensure that the time-domain reservoir accurately tracks structurally correct feature profiles

5.2.2. End-to-End Processing Architecture

Our complete processing framework is designed for end-to-end efficiency. The overall system involves a cascade of two reservoirs: First reservoir (feature extraction reservoir) performs MFCC feature extraction, followed by a second reservoir (classification reservoir) performing audio identification/verification (classification) as shown in figure 5.1.

A critical architectural consideration is the large disparity in the dimensionality of the input and output signals. The raw audio data (input into feature extraction reservoir is typically a long time series, denoted by the size $[N \times 1]$ (where N is the number of audio samples). The corresponding MFCC output (the output of feature extraction reservoir and input to classification reservoir is a compact feature matrix of size $[M \times 14]$, where M is the number of frames and the columns represent the 14 coefficients. Crucially, N is significantly greater than M ($N \gg M$).

To manage the significant dimensionality reduction and ensure the reservoir operates efficiently, a precise windowing technique is applied. While windowing is a foundational method for analysing non-stationary audio signals (90), preliminary experiments conducted in this research demonstrated that a structured windowing approach is necessary to align the temporal dynamics of the reservoir with the target feature set. Specifically, the raw audio signal is segmented so that each resulting window, when processed by the reservoir, corresponds to exactly one discrete data point (one frame) of the MFCC coefficient output.

5.2.3. Windowing Strategy and Mathematical Derivation

To determine the appropriate window size for this time-domain segmentation, we adopted a method that establishes numerical equivalence with the frame count of the standard MFCC function. We invert the standard MFCC framing process to determine the required window size based on the final frame count M.

The first step is calculating the number of non-overlapping windows ($N_{windows}$) that the audio signal is divided into. We found that the standard MFCC process, factoring in common frame overlap ratios (usually 50%), often yields an effective number of non-overlapping frames that can be approximated by the equation:

$$N_{windows} = \frac{M_{MFCC} + 2}{2} \tag{5.1}$$

where M_{MFCC} is the total number of MFCC samples (frames) calculated by the reference Matlab function. Once $N_{windows}$ is calculated, the precise Window Size for the RC input is determined by dividing the total number of audio samples (S_{audio}) by this number of effective non-overlapping windows:

$$\text{Window Size} = \frac{S_{audio}}{N_{windows}} \tag{5.2}$$

This ensures that the sequential processing of audio segments, equal to the calculated Window Size, will result in an output vector that aligns dimensionally with the M_{MFCC} target frames. This alignment ensures that each reservoir output corresponds precisely to a single MFCC data point, which is essential for evaluating the system's performance. By establishing this dimensional consistency, the system's fitness can be calculated directly without resorting to arbitrary averaging or temporal interpolation. This approach ensures a more rigorous evaluation of the reservoir's mapping accuracy by maintaining a one-to-one relationship between the processed states and the reference features.

The feature extraction reservoir is then created and configured such that

its input size is equal to the calculated window length of the signal. The audio signal corresponding to one window length is provided as input to the reservoir, and the target for the reservoir's training is the corresponding single MFCC coefficient frame. We rigorously evaluate the performance by measuring the NRMSE between the reservoir's output and the target coefficient to confirm the feature-extraction reservoir's accuracy.

5.2.4. System Validation and Performance Testing

Following the extraction of RC-TDFE features via the reservoir framework, we must now validate the end-to-end system's functional utility and verify its performance across diverse speech tasks. To do this, the time-domain MFCCs generated by the feature extraction reservoir is fed directly into a subsequent classifier reservoir. We then perform comprehensive speaker and digit recognition experiments to evaluate the efficacy of this integrated methodology.

Initial validation is conducted using the TI-46 dataset, confirming the system's foundational ability to extract features that retained phonetic and speaker information. Subsequently, we verify the system's performance using the larger and more acoustically diverse Audio MNIST dataset, thereby confirming the system's scalability and robustness.

The initial finding is that this reservoir-based time domain feature extraction methodology works fine, successfully generating competitive time-domain features. However, recognizing that substantial gains are often achieved through fine-tuning, we are actively investigating various factors that can impact and hence improve the performance of the RC-based feature extraction

method.

5.2.5. Reservoir Parameters

The specific parameters for the classifier reservoir are tuned for optimal performance in sequence classification:

- **Node Count:** The reservoir is composed of **950 nodes**.
- **Spectral Radius (ρ):** The spectral radius of the recurrent weight matrix ($S_x.W_{\text{res}}$) is scaled precisely to $\rho = \mathbf{0.8}$.
- **Sparsity:** A sparsity of **80%** is implemented (connections established with a 20% probability).
- **Leakage Rate (α):** A leakage rate (S_x.leak) of $\alpha = \mathbf{0.3}$ is utilized.
- **Washout:** A **washout period of 50 time steps** is applied.
- **State Update Dynamics:** The activation equation:

$$X(i+1) = \tanh\left(S_x.\text{leak} \cdot \left(X(i) \cdot S_x.W_{\text{res}}\right) + u(i) \cdot S_x.W_{\text{in}}\right) \quad (5.3)$$

  where $X(i)$ is the internal reservoir state vector at time step it, $u(i)$ is the input vector, $S_x.W_{\text{in}}$ represents the scaled input weight matrix, and $S_x.W_{\text{res}}$ denotes the recurrent reservoir weight matrix.

- **Input Weight Scaling:** The raw input weight matrix W_{in} is normalized and scaled symmetrically into the $[-1, 1]$ range via the min-max operation:

$$S.W_{\text{in}} = \left(\frac{W_{\text{in}} - mn}{mx - mn} \cdot 2\right) - 1 \quad (5.4)$$

where mn and mx denote the minimum and maximum boundaries of the input weight matrix.

5.3. Improving Performance of MFCC Extraction

We identified that the window length and, consequently, the effective frame overlap used in the temporal decimation phase, has a significant impact on the performance of the end-to-end system. This is a critical parameter, governing the trade-off between local acoustic detail captured and the overall length of the feature vector. Based on our findings, we are analysing three distinct scenarios to optimize the windowing strategy:

Setup 1: Baseline Windowing

This scenario establishes the baseline performance of the system, utilizing the window length obtained directly from the derived baseline equation (Equation 3.1). This calculation yields the window size that minimizes effective frame overlap, providing a standardized, minimally redundant feature set.

Setup 2: Dense Windowing (Decreased Window Size)

In this method, we investigate the impact of increasing the temporal resolution of the feature extraction. This is achieved by decreasing the window size, which, mathematically, results in a corresponding increase in the number of overlapping windows. By sampling the raw audio more frequently, the resulting MFCC features provide a denser temporal representation of the

acoustic event. Initial findings have shown that this method performs better than the first method, demonstrating the benefit gained from increased contextual information.

Setup 3: Tuned Windowing (Task-Specific Optimized Overlap)

The success of Scenario 2 led to a specialized investigation into optimizing the windowing based on the classification task. Recognizing that lower frequencies (lower Mel Coefficients) are critical for digit identification, and higher frequencies (higher Mel Coefficients) are vital for speaker identification, we developed a non-uniform windowing strategy:

- **For Speaker Identification:** We increased the number of overlapping windows by decreasing the window size for higher frequencies (higher-order Mel Coefficients) to maximize the resolution of subtle speaker-specific spectral details.

- **For Digit Identification:** Conversely, we increased the number of overlapping windows by decreasing the window size for lower frequencies (lower-order Mel Coefficients) to focus resources on extracting the robust phonetic information necessary for accurate digit classification.

This approach tailors the feature extraction process itself to the final classification goal, maximizing the informativeness of the features for each respective task.

5.3.1. Parameter Assignments for windowing

Baseline Windowing

The baseline windowing strategy sets a fixed, standard reference point for short-time audio analysis. Based on the underlying feature extraction equations, the baseline architecture utilizes a static window size. for example for Ti-46 dataset window size $W_{\text{base}} = 250$ time steps, accompanied by a window increment length (hop size) of $H_{\text{base}} = 125$ time steps. This configuration ensures a standard 50% frame overlap, balancing temporal tracking with computational payload restrictions on edge hardware.

Dense Windowing

In this case we are keeping the window increment length fixed to match the baseline framework ($H_{\text{dense}} = 125$ time steps), the active window frame size is compressed to $W_{\text{dense}} = 200$ time steps. By shrinking the window size while keeping the hop rate steady, the system extracts more localized states from the signal per frame, creating a denser representation of immediate spectral shifts.

Tuned Windowing

The tuned windowing approach introduces channel-dependent multi-rate sampling to accommodate the distinct physical natures of separate acoustic classification tasks. Because lower and higher frequency bands require different structural granularities, the windowing properties are modified dynamically based on whether the target task is isolated digit recognition or speaker iden-

tity verification.

Digit Recognition Configuration: For the digit recognition task, temporal duration tracking of phonetic structures takes priority.

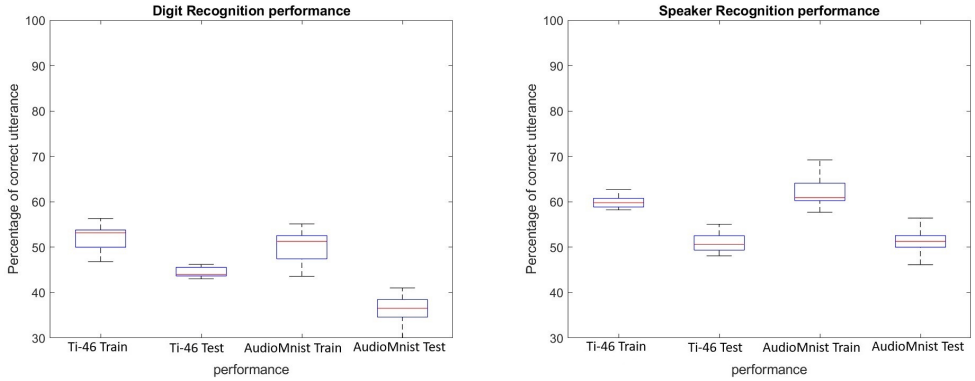
- **Channels 1–8 (Lower Mel Coefficients):** The sampling frequency (F_s) is halved ($F_s/2$) for these initial channels. Structurally, halving the sampling rate doubles the resulting total frame count (window number) across the temporal span, granting twice the temporal resolution to capturing foundational low-frequency phonetic shifts.
- **Channels 9–14 (Higher Mel Coefficients):** To match the doubled frame count established by the first 8 channels without altering their high-frequency capture properties, the extracted values for channels 9 through 14 are duplicated sequentially. This maps the entire high-frequency envelope evenly across the expanded time matrix.

Speaker Identification Configuration: Conversely, speaker identification relies heavily on high-frequency, fine-grained glottal characteristics and unique timbre profiles located in the upper spectrum. The multi-rate strategy is therefore inverted:

- **Channels 9–14 (Higher Mel Coefficients):** The sampling frequency (F_s) is manipulated to double the frame count across these upper-indexed channels. This localized oversampling captures high-resolution biometric vocal traits.
- **Channels 1–8 (Lower Mel Coefficients):** The lower-indexed coefficient values are systematically duplicated to align their matrix di-

mensions with the doubled frame structures generated by the high-frequency channels.

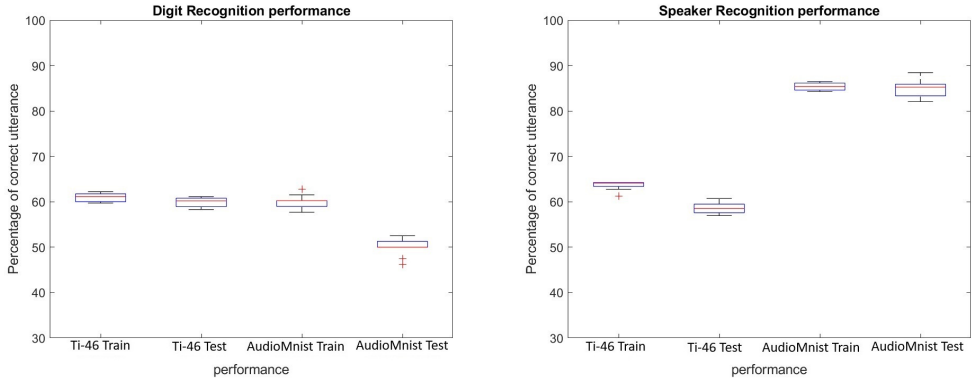
5.4. Results



(a) Digit Recognition Performance of Baseline windowing

(b) Speaker Recognition Performance of Baseline windowing

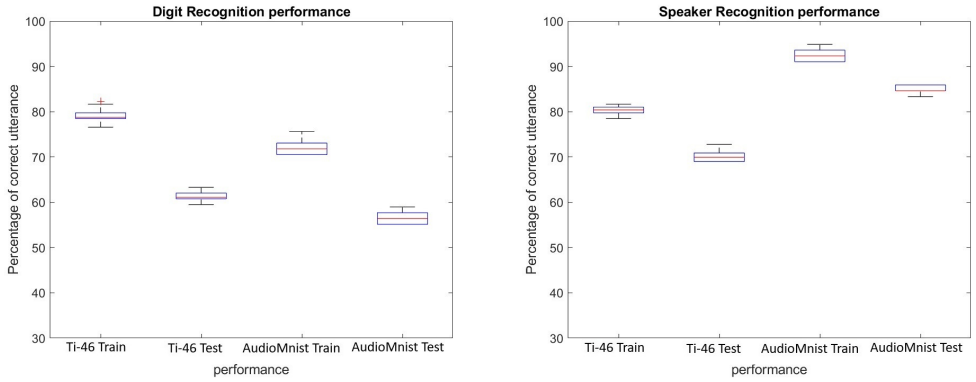
Figure 5.2: Speaker and Digit Recognition performance of Baseline windowing



(a) Digit Recognition Performance of Dense windowing

(b) Speaker Recognition Performance of Dense windowing

Figure 5.3: Speaker and Digit Recognition performance of Dense windowing



(a) Digit Recognition Performance of Tuned windowing

(b) Speaker Recognition Performance of Tuned windowing

Figure 5.4: Speaker and Digit Recognition performance of Tuned windowing

Experiments	Digit Recognition				Speaker Recognition			
	Ti-46		AudioMnist		Ti-46		AudioMnist	
	Train	Test	Train	Test	Train	Test	Train	Test
Baseline windowing	53.16	43.99	51.28	36.53	59.87	51.24	62.87	53.84
Dense windowing	61.22	63.32	62.02	53.22	66.22	58.22	88.24	86.88
Tuned windowing	78.96	61.29	74.34	60.82	80.88	70.88	91.87	84.89

Table 5.2: Comparison of different experiments for Digit and Speaker Recognition

The training and testing performance are shown as box-plots. We have conducted both Digit and Speaker recognition using Ti-46 dataset and AudioMnist data set. The box-plots of training and testing performance of the reservoir as a classifier are shown in figure 5.2, 5.3, and 5.4.

Table 5.2 summarises the results of three experiments. This chapter has investigated the use of Reservoir Computing (RC) as a direct alternative to the traditional MFCC extraction pipeline. By comparing the reservoir-based feature extraction against standard Matlab benchmarks, the results demonstrate that the RC framework accurately captures essential acoustic features without transitioning into the frequency domain. Chapter 8, table 8.1 and

table 8.2 provides a more detailed comparison of this method with other methods.

5.5. Summary and Conclusion

This chapter demonstrates that Reservoir Computing provides a highly efficient, time-domain alternative to traditional MFCC extraction by eliminating the computational overhead of Fourier Transforms. Our findings confirm that a reservoir architecture could extract features similar to standard Matlab-based methods while significantly reducing parameter count and maintaining a streamlined time-domain pipeline. Ultimately, this approach simplifies hardware implementation and enables real-time audio processing without the need for complex frequency-domain conversions.

While time-domain mimicking methods theoretically operate within the temporal domain, they are fundamentally hampered by their attempt to replicate Mel Frequency Cepstral Coefficients (MFCCs). The system is forced to imitate features that necessitate complex frequency-domain transformations necessary when conventionally computing MFCCs, resulting in a 'functional mismatch' with the reservoir's original architecture. These demerits are effectively overcome by the time-domain convolution method discussed in the following chapter, which eliminates the need to emulate frequency-based features by remaining aligned with the reservoir's more natural inherent temporal processing dynamics.

6

Feature Extraction by Convolution

The functional mismatch of the mimicking approach identified in the previous chapter has been the inspiration for improving the MFCC extraction framework, which is detailed in this chapter. By synthesizing Mel Filters directly within the time domain and using reservoir-based convolutions, we leverage the reservoir's inherent dynamics to replace traditional domain transformations. This unified approach facilitates the extraction of discriminative features from raw audio while significantly reducing the computational overhead associated with conventional processing.

Feature extraction is a critical step in speech signal processing, with Mel Frequency Cepstral Coefficients being one of the most widely used features. However, conventional MFCC extraction relies on time-frequency transformations, which introduce computational complexity. To address the complexities of the MFCC extraction pipeline, we introduce a simplification based on Reservoir Computing (RC). As RC is inherently capable of operating in the time domain (TD), we are eliminating the need for multiple domain conversions. This is achieved by replacing those MFCC extraction steps with 'convolution' operation, enabling direct Mel Frequency Convolution Filter(MFCF) extraction without time-frequency transformations. This approach not only reduces computational overhead but also aligns seamlessly

with Reservoir Computing architectures.

While Mel Frequency Cepstral Coefficients represent numerical values within the cepstral domain, features extracted with the proposed approach remain entirely in the time domain. Consequently, labelling them a 'Mel Frequency Cepstral Coefficients' would be technically inaccurate. To better reflect their temporal and convolutional nature, Mel-inspired features are termed Mel Frequency Convolution Filters (MFCF) in this thesis.

As stated in our hypotheses, we construct an experimental framework that aims at end-to-end audio processing. This framework is tested to assess its viability for energy-efficient, real-time speech signal analysis. The streamlined architecture maintains classification performance with lower computational complexity, validating the use of time-domain Reservoir Computing for acoustic feature extraction. Our findings mark a significant step toward developing low-power, high-performance audio processing systems, which will facilitate their integration into practical applications.

6.1. Time-Domain Mel Filter-Bank Synthesis

Our work establishes a novel capacity for time-domain audio pre-processing by directly synthesizing the Mel equivalent filter bank operation in the time domain, allowing the Echo State Network (ESN) to learn the feature extraction process directly from the raw waveform. This novel approach enables the Reservoir Computing (RC) system to operate as a true, end-to-end time-domain feature extractor, entirely circumventing the need for computationally expensive domain conversions.

6.1.1. Synthesis of the Mel Filter-Bank Signal

The conventional Mel Filter-Bank consists of a series of triangular weighting functions applied in the frequency domain. These filters are designed to isolate spectral energy within specific frequency bands, simulating the non-linear response of the human ear. Our approach replicates this function not through spectral analysis, but through direct time-domain synthesis (136).

For each of the N target Mel Coefficients, we analytically determine the set of constituent frequencies and their necessary parameters that, when combined, accurately mimic the resulting spectral characteristics of the desired triangular Mel Filter. This process requires mapping the spectral properties of the frequency-domain filter onto a set of time-domain components. The specific frequency and parameter sets corresponding to each Mel Coefficient are systematically catalogued. In table 6.1, where the detailed composition for the first Mel Coefficient is given as an example. We synthesise our mel wave based on the equation given below.

To synthesize the time-domain impulse response for the first Mel coefficient we use data given in table 6.1, each Δf_i denotes a constituent frequency (in Hz) and each corresponding parameter acts as its scaling amplitude weight (A_i). By applying the principle of superposition, these weighted discrete components are linearly combined into a single, complex time-domain waveform via equation

$$x(t) = \sum_i (\text{parameter}_i) \cdot \sin\left(\frac{2\pi \cdot \Delta f_i \cdot t}{F_s}\right) \tag{6.1}$$

Where:

- $x(t)$ represents the synthesized time-domain waveform (impulse response) corresponding to the targeted Mel coefficient.
- parameter$_i$ denotes the scaling amplitude weight (A_i) assigned to the i-th constituent frequency component, structurally shaping the ascending and descending slopes of the triangular filter.
- Δf_i is the specific constituent discrete frequency (measured in Hz) of the i-th sine wave component extracted from the filter-bank profile.
- F_s represents the sampling frequency (in Hz), which normalizes the continuous-time components into the discrete-time domain to prevent spectral aliasing.
- t represents the discrete time-step index, ranging from 0 to the maximum window length (typically between 100 and 350 time-steps in this implementation) to define the finite duration of the impulse response.

This resulting oscillating waveform represents the time domain impulse response of the filter.

parameter=	0.002454697	0.004909393	0.00736409	0.009818787
Δf=	131	141	151	161
parameter=	0.007781114	0.00561907	0.003457026	0.001294982
Δf=	171	181	191	201

Table 6.1: Parameters and frequencies for first Mel coefficient

Signal Generation via Superposition

A waveform is then constructed corresponding to each specific Mel Coefficient. For each entry in the parameter table, we synthesise a sine wave

corresponding to its frequency and parameter set. Crucially, we then superimpose all these synthesized sine waves to obtain a single, complex Mel Filter-Bank signal in the time domain. This synthesized waveform acts as the time-domain impulse response of the filter, and this systematic synthesis process is repeated for all N Mel Coefficients, creating a full set of distinct time-domain filter bank signals. The final visualized result is shown in figure 6.1. We showcases the Mel Filter-Bank not as a set of static triangles in the frequency domain, but as a series of complex, oscillating waveforms in the time domain.

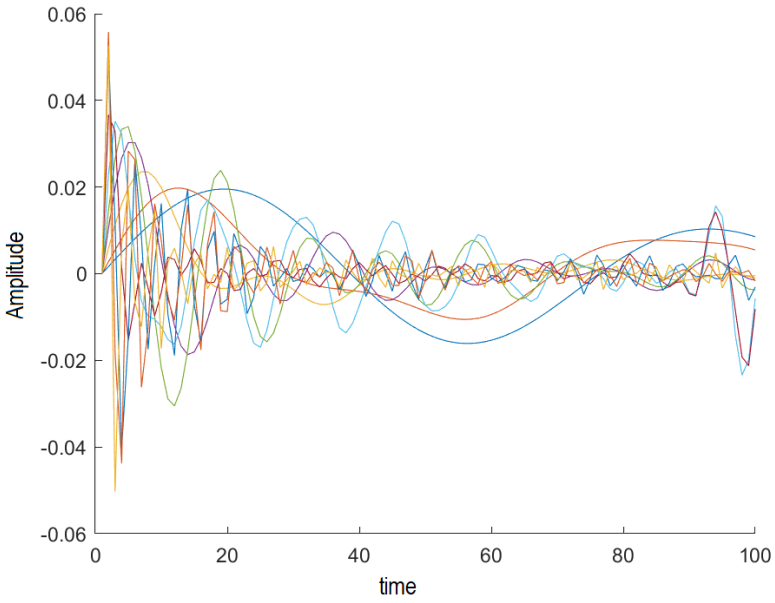


Figure 6.1: Time-domain filterbank

Figure 6.1 illustrates the time-domain translation of the Mel filter-bank, showcasing the physical waveforms synthesized via the superposition process. As observed in the figure, the lower channels exhibit slower, low-frequency oscillations with broader envelopes, which structurally replicate the tight spac-

ing of the initial Mel filters. Conversely, the higher-indexed traces display progressively rapid oscillations with narrower, densely packed peaks. This visual evolution directly reflects the underlying parameter distribution: as the centre frequency increases, the component frequencies (Δf_i) scale upward, while the corresponding amplitude parameters modulate the structural decay. Consequently, figure 6.1 visualizes the transformation of abstract, static frequency-domain triangles into complex, dynamic time-domain waveforms ready for direct signal convolution.

6.2. Time-Domain Mel Feature Extraction

6.2.1. Reservoir Training for Time-Domain Convolution

In signal processing, convolution in time domain is the same as applying a filter in the frequency domain. Therefore, the core objective of this stage is to train the Reservoir Computing system to perform the time-domain convolution operation between the input audio signal and each synthesized time-domain filter bank signal. The desired output for each Mel Coefficient at any given time is the mathematically calculated result of the convolution between the raw audio signal and the respective synthesized filter bank signal. The raw audio signal is fed into the reservoir continuously, creating a complex, high-dimensional state trajectory. We then train a simple, linear readout layer to map this complex state directly to the calculated convolution result. By training a separate readout layer for each target Mel Coefficient, we effectively train the reservoir to perform multiple convolution operations

simultaneously. Once trained, the reservoir's readout directly outputs the Mel inspired audio feature in the time domain. This process successfully eliminates the need for the computationally demanding FFT operation, as its function is now absorbed into the dynamic, fixed weights of the reservoir and the training of the readout layer. The result is a highly simplified feature extraction pipeline that preserves the desired perceptual properties of the Mel scale while achieving the superior computational efficiency inherent to the RC framework. However, this time-domain convolution results in a significantly larger volume of samples than traditional frequency-domain methods. This creates a need for strategic data reduction.

6.2.2. Data Reduction for Experimental Framework Integration

Following the successful synthesis and application of the time-domain Mel Filter-Bank signals via the Reservoir Computing (RC) system—where the ESN implicitly learned the convolution operation—the subsequent critical challenge is temporal decimation. The convolution of the audio signal (which has a certain number of samples) with the filter bank signal (which also has a finite, non-zero length) naturally results in an output signal that is longer, i.e. consists of more samples, than the original audio. This elongated signal is computationally inefficient and contains redundant information, primarily in the form of convolution residue. Therefore, it becomes necessary to significantly reduce the number of data points without sacrificing the rich acoustic information captured by the RC system.

The two goals are to isolate the essential features and format the data for

seamless integration with the subsequent analytical stages, particularly the second classification reservoir used later in the experiment. The decimation process begins with signal pruning. We first trim the data-points which are beyond the length of the original audio signal. This trailing segment of the convoluted signal carries minimal meaningful information about the input audio and is largely a residue of the convolution operation. Removing it isolates the core signal. Next, to standardise the output for the experimental framework, the remaining signal is subjected to windowing. The goal is to obtain a discrete sequence of output features that aligns numerically with the standard output of conventional methods. We split the signal into windows such that the total number of windows is precisely equivalent to the number of MFCC output samples that would be generated by a standard implementation (e.g., using the Matlab MFCC function) on the same audio input. This equivalence ensures direct comparability and structural compatibility with the second reservoir, which expects a feature vector of a predetermined, standardized size. Finally, to produce the final, single feature point for each window, we employ a specific peak-to-peak sampling strategy. Out of each defined window, we select just one data-point based on the peak-to-peak amplitude measurement within that segment. This method effectively captures the maximum energy or variability within that short temporal window, serving as the final, temporally decimated feature vector (as visualized in Figure 6.2). This process of windowing and peak-to-peak sampling preserves the inherent simplicity of time-domain feature generation and ensuring that complex data reduction does not compromise the goal of straightforward, end-to-end audio processing. Simultaneously, it produces a

compact, standardized output ideal for subsequent classification tasks. Figure 6.2 illustrates this Mel Frequency Convolution Filter (MFCF) extraction process within the time domain.

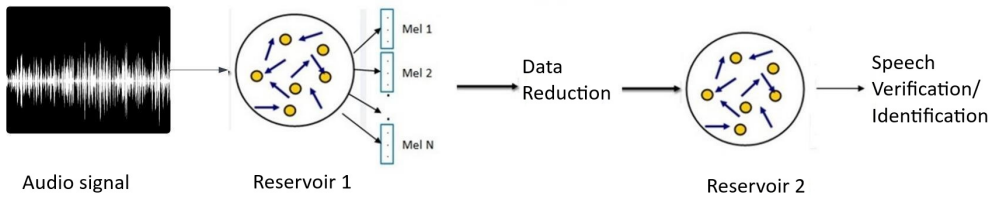


Figure 6.2: Time-domain MFCF extraction

6.3. Single Reservoir with Multi-Output Architecture

Our initial experimental validation of the Reservoir Computing Time-Domain Feature Extractor (RC-TDFE) focuses on maximizing computational simplicity by implementing a single-reservoir, multi-output architecture. In this baseline setup, a singular Echo State Network (ESN) is employed for the entire feature extraction process. The core functionality requires the ESN to implicitly learn the time-domain convolution operation for all 14 target Mel Frequency Cepstral Coefficients (MFCCs) selected. To achieve this, the single reservoir is connected to 14 distinct output layers (readouts), with each layer being individually trained to generate one specific Mel Frequency Convolution Filter(MFCF), from MFCF_1 through MFCF_{14}. This design aims to test the limits of the RC framework's ability to model multiple, complex, and si-

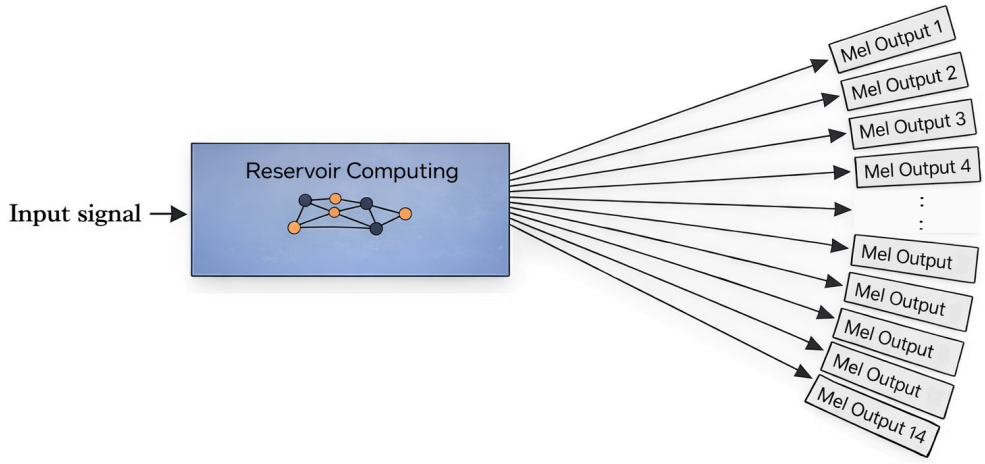


Figure 6.3: Single Reservoir architecture for feature extraction

multaneous temporal dynamics using minimal hardware resources. Figure 6.3 shows a schematic of single RC architecture. Within this single-reservoir constraint, we explore two distinct methodological approaches for integrating the time-domain filter bank operation with the reservoir's processing flow. Crucially, the difference between the two approaches lies in the sequencing of the convolution and the temporal windowing/decimation steps.

6.3.1. Continuous Convolution Followed by Decimation

In this first approach, the single reservoir operates continuously on the unsegmented, raw audio stream, where it is trained to perform an implicit time-domain convolution across all 14 synthesized Mel filter-bank channels simultaneously. The 14 output layers simultaneously predict the corresponding convolved signals, resulting in 14 continuous output streams, each con-

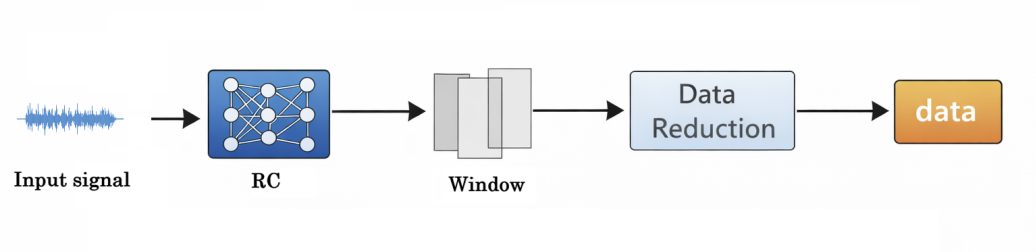


Figure 6.4: Convolve then window method

siderably longer than the original audio due to the convolution residue. The subsequent stages involve temporal decimation to normalize the feature dimensions. Following the output of the reservoir, the data points corresponding to the convolution residue—those exceeding the original audio signal length—are trimmed as they carried minimal meaningful information. The remaining signal is then divided into a series of overlapping windows. The number of windows is chosen to be equivalent to the number of MFCC output samples that would be generated by a standard, frame-based frequency-domain feature extractor (e.g., the Matlab MFCC function). This precise sizing is essential to ensure the output structure fits seamlessly into the experimental framework for the second, downstream classification reservoir. Finally, from each defined window, a single representative data point is extracted using a peak-to-peak amplitude measurement, yielding the final, compressed time-domain MFCF feature vector. This method leverages the reservoir's fading memory to process long-term temporal context before the final data compression. Figure 6.4 illustrates the method.

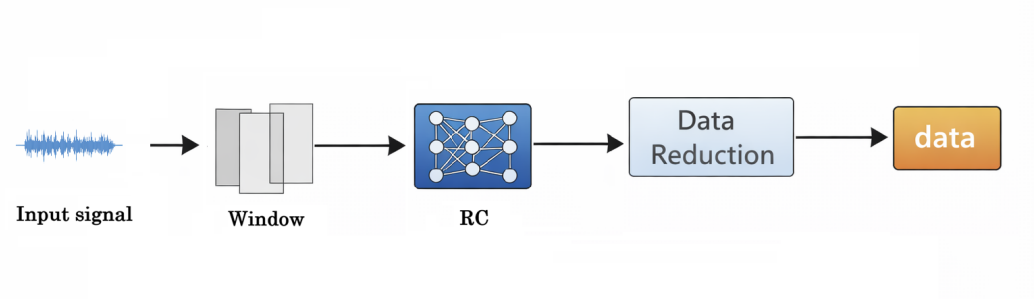


Figure 6.5: Window then convolve method

6.3.2. Frame-by-Frame Convolution

The second approach adopts a more traditional frame-based processing structure, prioritizing the segmentation of the input signal before processing. In this design, the audio signal is first segmented into overlapping windows, where each window is designed to eventually contribute a single Mel Coefficient. The single reservoir is then utilized to process the windowed audio signal. For each individual window, the reservoir is tasked with performing the implicit convolution with the entire synthesized Mel Filter-Bank signal, followed by a slicing and a final data reduction operation to obtain one Mel time-domain Coefficient for that window. Conceptually, this method is designed to mimic the short-time analysis characteristic of conventional FFT-based methods. While intuitively appealing for maintaining strict local control over the analysis frame, this approach places greater strain on the reservoir to rapidly discard and rebuild context with every new window. The continuous resetting of the input context prove detrimental to the reservoir's ability to utilize its inherent temporal memory across frame boundaries effectively. Figure 6.5 illustrates the method.

6.3.3. Quantitative Comparison and Methodological Selection

A thorough quantitative comparison is conducted between Approach 1 and Approach 2, focusing on two critical performance metrics: computational speed and prediction accuracy, measured using the Normalized Root Mean Square Error (NRMSE). The results clearly indicate that Approach 1 is better, it demonstrated quicker execution times, primarily because the reservoir only processes the input stream once, with the feature segmentation occurring passively downstream. More significantly, Approach 1 consistently produces a smaller NRMSE, indicating a higher fidelity and lower error in the predicted Mel Coefficients. This finding suggests that allowing the single reservoir to continuously view the input signal, leveraging its continuous fading memory, provided a richer, more contextually complete internal state for the readout layers to extract the implicit convolution result. The performance in both efficiency and accuracy led us to select Approach 1 as the baseline methodology for our experiments and analysis.

6.4. The Transition to the Multi-Reservoir System

With the single reservoir established as our baseline, we investigated, whether the system is constrained by the raw processing power of one reservoir or by the inherent difficulty of tuning a non-specialized parameter set.

To isolate these factors, we implement an intermediate validation step: a

decoupled architecture using 14 identical reservoirs, each assigned to a single output layer. By keeping the configuration parameters uniform across all 14 units, we create a direct comparison to the single-reservoir setup.

The experimental output from this multi-reservoir is identical to the single-reservoir system. This confirms that adding more recurrent units does not inherently improve performance if they are all governed by the same generalized dynamics. This finding provides the final justification for our advanced setup: a parallel feature extraction where 14 reservoirs are trained and optimized individually to capture the unique spectral features of each MFCF.

The primary bottleneck in the single-reservoir approach was the reliance on a singular parameter set to model 14 distinct spectral features. This universal setup prevented ideal fidelity for any single Mel Coefficient, although being computationally efficient.

To overcome this, we transitioned to a parallel feature extraction engine. In this architecture, we utilize 14 independent reservoirs, each possessing its own unique hyper-parameter configuration. This specialization allows each reservoir's internal dynamics—such as its spectral radius and connectivity—to be tuned specifically to the frequency characteristics of its assigned Mel band.

6.4.1. Multi-Reservoir Architecture for Feature Extraction

The Reservoir Computing Time-Domain Feature Extractor (RC-TDFE) is defined by an architecture consisting of 14 distinct and parallel Reservoir

Computing (RC) systems. This design represents a methodological departure from single-model feature extraction, where one complex network is tasked with modelling the entire feature vector simultaneously. Instead, the complex process of generating the full 14-dimensional feature vector is decomposed into 14 parallel and independent operations. Each dedicated reservoir is assigned the sole responsibility of extracting a single, specific Mel-Frequency Convolution Filter (MFCF) feature. This parallelization strategy enables a highly targeted, specialized, and robust approach to feature generation, fundamentally enhancing the fidelity and overall accuracy of the extracted acoustic features.

6.4.2. Architectural Decomposition and Parallelism

The decision to transition from a single, monolithic RC system to a distributed, multi-reservoir framework is rooted in the inherent complexity of the feature extraction task. While the Mel Filter-Bank structure is perceptually motivated, the mathematical characteristics of the resulting coefficients vary widely. For instance, the lower-order coefficients (MFCF 1–4) typically capture the broad spectral envelope, which is crucial for phoneme identity, while higher-order coefficients (MFCF 8–14) capture fine spectral details often related to speaker vocal tract differences and noise components. Modelling this wide range of phenomena effectively with a single set of fixed reservoir weights and a single, generalized readout layer proves inefficient. Figure 6.6 shows a schematic of Multi-reservoir architecture.

By dedicating an individual, unique reservoir to each coefficient, the architecture leverages the principle of task-specific specialization. This allows

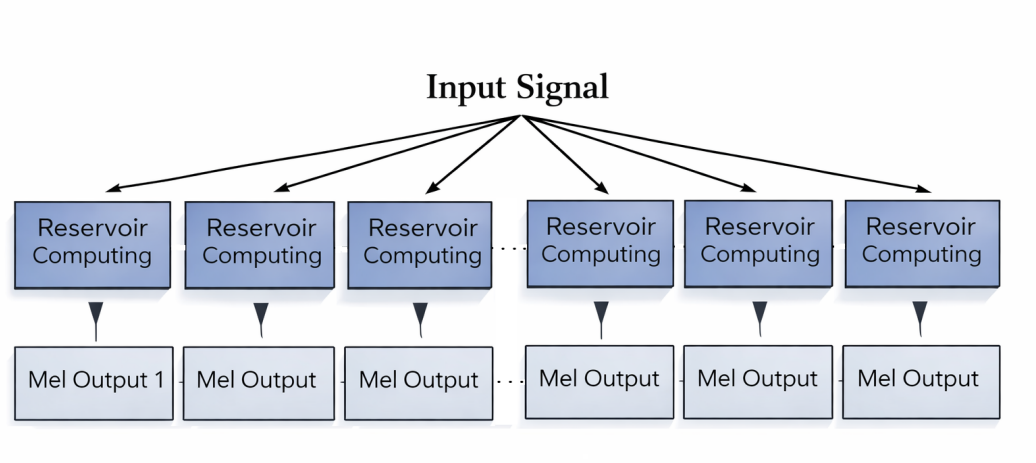


Figure 6.6: Multi-Reservoir architecture for feature extraction

the dynamics of each reservoir—determined by its internal connectivity, spectral radius, and input scaling—to be tailored to the particular input features and memory requirements necessary to accurately predict its assigned coefficient. The system avoids the limitations imposed by a generalized model that must compromise its performance across different spectral regions. In effect, we are creating 14 specialized experts, each optimized for a narrow, well-defined task. This approach minimizes the competing demands placed on a single model and enhances the model capacity without increasing the computational complexity of the core recurrent processing, as all reservoirs operate concurrently on the same raw input stream.

6.4.3. Optimization via NRMSE

The efficacy of the multi-reservoir architecture is realized through a independent optimization protocol. The design mandates that each individual reservoir is being treated as a separate computational unit, independently tuned and optimized with the objective of accurately predicting its desig-

nated Mel Coefficient. This optimization involves a systematic exploration of parameter space, including reservoir size, spectral radius, sparsity, and input scaling, to optimize performance for each coefficient.

The goal is to systematically adjust each reservoir's parameters to achieve the lowest possible NRMSE for its specific coefficient. This focused optimisation ensures maximum fidelity for each component of the 14-dimensional feature vector. The low NRMSE of each system proves that the reservoir has learned the implicit convolution operation for that filter's impulse response. This decentralised, error-driven tuning process guarantees a robust and highly accurate collective output across the feature space. Table 6.2 shows the Normalised mean square error (NRMSE) of the feature extraction reservoir.

NRMSE	Mel 1	Mel 2	Mel 3	Mel 4	Mel 5	Mel 6	Mel 7
single RC	0.1138	0.2216	0.1784	0.0216	0.0094	0.0079	0.0077
multi RC	0.0099	0.0053	0.0045	0.0062	0.0067	0.0069	0.0070
NRMSE	Mel 8	Mel 9	Mel 10	Mel 11	Mel 12	Mel 13	
single RC	0.0068	0.0069	0.0074	0.0065	0.0066	0.0053	
multi RC	0.0061	0.0061	0.0062	0.0062	0.0062	0.0053	

Table 6.2: NRMSE of Reservoir-1 extracting MFCC, based on 14 Mel Coefficients

6.4.4. Impact of Multi-Reservoir Approach

The methodological rigour of the multi-reservoir, coefficient-specific optimization translates directly into significant and valuable improvements in the performance of time-domain MFCF extraction. Because each coefficient is generated by an expert system dedicated to minimizing error for that

specific component, the resulting feature vector is intrinsically more representative of the underlying speech signal compared to features extracted by generalized models. The enhanced fidelity and minimized noise component of this feature set offer substantial advantages for subsequent machine learning tasks.

The validation of this advanced feature set is directly observed in downstream task performance. As evident in our experimental results, the features generated by the selectively composed, tailored multi-reservoir system lead to notable improvements in both digit recognition accuracy and speaker identification performance. In the context of digit recognition, the improved fidelity in the lower-order coefficients (spectral shape) enhances the discriminative power between phonemes. For speaker identification, the greater accuracy in the higher-order coefficients (related to vocal tract details and speaker characteristics) provides a sharper distinction between individuals. The tailored reservoir system is a highly effective strategy for improving speech processing systems. It shows that architectural specialisation and decentralised optimisation are crucial for superior feature extraction quality in resource-efficient, time-domain RC frameworks. This robust performance validates the novel end-to-end paradigm, confirming that the RC-TDFE not only simplifies the pipeline but also fundamentally improves the quality of the resulting acoustic features.

6.4.5. Reservoir Parameters

The specific parameters for the classifier reservoir are tuned for optimal performance in sequence classification:

- **Node Count:** The reservoir is composed of **35** nodes for each reservoir.

- **Spectral Radius (ρ):** The spectral radius of the recurrent weight matrix ($S_x.W_{\text{res}}$) is scaled precisely to $\rho = \mathbf{0.8}$.

- **Sparsity:** A sparsity of **80%** is implemented (connections established with a 20% probability).

- **Leakage Rate (α):** A leakage rate ($S_x.\text{leak}$) of $\alpha = \mathbf{0.3}$ is utilized.

- **Washout:** A washout period of **50** time steps is applied.

- **State Update Dynamics:** The activation equation is:

$$X(i+1) = \tanh\left(S_x.\text{leak} \cdot \left(X(i) \cdot S_x.W_{\text{res}}\right) + u(i) \cdot S_x.W_{\text{in}}\right) \quad (6.2)$$

  where $X(i)$ is the internal reservoir state vector at time step it, $u(i)$ is the input vector, $S_x.W_{\text{in}}$ represents the scaled input weight matrix, and $S_x.W_{\text{res}}$ denotes the recurrent reservoir weight matrix.

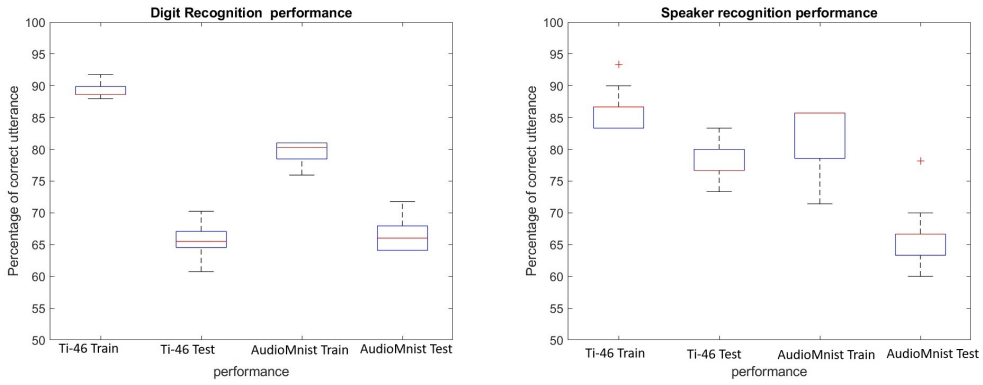
- **Input Weight Scaling:** The input weight matrix W_{in} is normalized and scaled symmetrically into the $[-1, 1]$ range via the min-max operation:

$$S.W_{\text{in}} = \left(\frac{W_{\text{in}} - mn}{mx - mn} \cdot 2\right) - 1 \quad (6.3)$$

  where mn and mx denote the minimum and maximum boundaries of the input weight matrix.

6.5. Results

The train and test performance of feature extraction in time domain using convolution method utilizing single reservoir is shown in box-plots 6.7.



(a) Digit Recognition Performance of single RC experiment

(b) Speaker Recognition Performance of single RC experiment

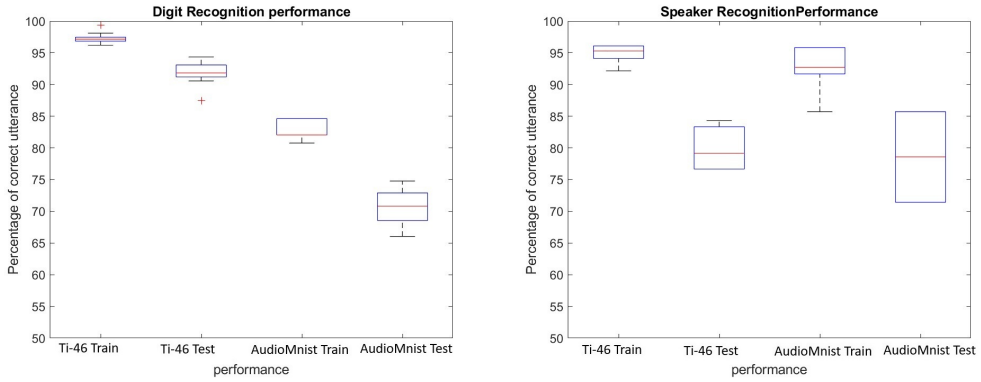
Figure 6.7: Speaker and Digit Recognition performance of Single RC Experiment

The train and test performance of feature extraction in time domain using convolution method utilizing 14 individual reservoirs is shown in box-plots 6.8.

Experiments	Digit Recognition				Speaker Recognition			
	Ti-46		AudioMnist		Ti-46		AudioMnist	
	Train	Test	Train	Test	Train	Test	Train	Test
Single Reservoir	88.61	65.52	82.64	68.17	87.09	76.83	86.23	66.93
Multi Reservoir	97.15	91.82	82.06	70.83	95.28	79.17	92.71	78.57

Table 6.3: Comparison of the two methods for Digit and Speaker Recognition

The table 6.3 summarises the result. Table 4.2 and 4.3 show how well a reservoir is able to extract MFCF in the time domain. For the case where the



(a) Digit Recognition Performance of multiple RC experiment

(b) Speaker Recognition Performance of multiple RC experiment

Figure 6.8: Speaker and Digit Recognition performance of Multiple RC Experiment

time domain MFCF is applied to the input of the second classifier reservoir, the resulting performance is shown as box plots in figure 6.7 and 6.8. The plot shows the ability of reservoir in extracting MFCF feature in time domain. Even with the simple max-pooling/peak-to-peak method that we have used to reduce the number of data points, we are able to get good performance. We were able to get better results with methods like Wavelet Transform if used as the data reduction method; however, they complicate the system and detract from the main objective of our study. Our focus for future work is therefore to further develop a simple methodology to reduce the number of data points without losing relevant information of the signal.

6.6. Summary and Conclusion

This chapter demonstrates that Reservoir Computing (RC) systems can extract Mel Frequency features directly in the time domain, utilizing reservoir-

based convolutions to bypass the usual need for domain conversions. This streamlined approach enables end-to-end audio processing that remains efficient even when incorporating simple data-reduction techniques like max-pooling. By delivering competitive performance with significantly less computational overhead, this framework provides a low-complexity alternative to traditional frequency-domain methods. The result is a unified architecture where the reservoir dynamics are natively aligned with the raw temporal details of the acoustic input. Chapter 8 table 8.1 and table 8.2 provides a more detailed comparison of this method with other methods.

7

"Feature-Free" Audio Processing

This chapter explores the potential of Reservoir Computing for "feature-free" audio processing, that is, audio processing without separate feature extraction stage. We investigate whether the reservoir's inherent dynamics can classify acoustic signals without any dedicated feature extraction stage, and thus simplifying the system into a single-stage, end-to-end processor.

This study seeks to answer whether a reservoir computer can directly process raw audio signals, thereby bypassing the complex feature extraction steps that are standard in audio pattern recognition. The goal is to eliminate the long-standing dependence on carefully designed, handcrafted features, which constitute a significant intellectual and computational bottleneck in traditional audio pipelines. Conventional methods mainly rely on techniques such as Mel-Frequency Cepstral Coefficients (MFCCs) to accomplish two crucial tasks: first, to significantly reduce the high dimensionality of the audio signal, and second, to highlight perceptually relevant information that is in line with human auditory mechanics. In contrast, we leverage the inherent temporal dynamics and high-dimensional projection of RC to model a "feature-free" audio processing framework. This architectural shift aims to produce a more robust, efficient, and general-purpose system that operates independently of specialized Digital Signal Processing (DSP).

7.1. Exploring Single RC for Raw Audio Processing

7.1.1. Raw Audio Data Size Challenge

A central and inescapable challenge in pursuing a "feature-free", single-step RC approach is the data-intensive nature of raw audio. Digital audio signals captured at standard sampling rates result in lengthy, high-resolution time-series waveforms. Presenting an input vector of tens of thousands or even hundreds of thousands of samples directly to a standard reservoir for tasks like word classification is computationally impractical and often too dense for the network to extract meaningful high-level information efficiently.

The reservoir's ability to create stable, discriminative recurring states may be undermined by the amount of redundant or low-value information in an unprocessed signal. In order to make the problem tractable and reduce the signal size into an input dimension that the reservoir's fixed structure can handle, some sort of data reduction is therefore a practical necessity rather than a matter of methodological preference. Making sure this needed reduction didn't undermine our main goal of simplification by merely adding another layer of complex non-linear processing was the main limitation.

7.1.2. Managing Data Size via Pre-Processing

The data reduction task stands as the most important step in attaining end-to-end, "feature-free" audio processing. Our goal of simplification would be compromised if we used a complicated, mathematically demanding, non-

linear transformation since it would only substitute one computationally costly feature extraction step with another. Rather, it is an explicit design constraint here to use only straightforward, lightweight pre-processing methods intended simply for dimensionality reduction rather than complex feature extraction. These techniques, which concentrate on reducing the signal size while essentially maintaining its structure and temporal characteristics, were selected for their computational simplicity.

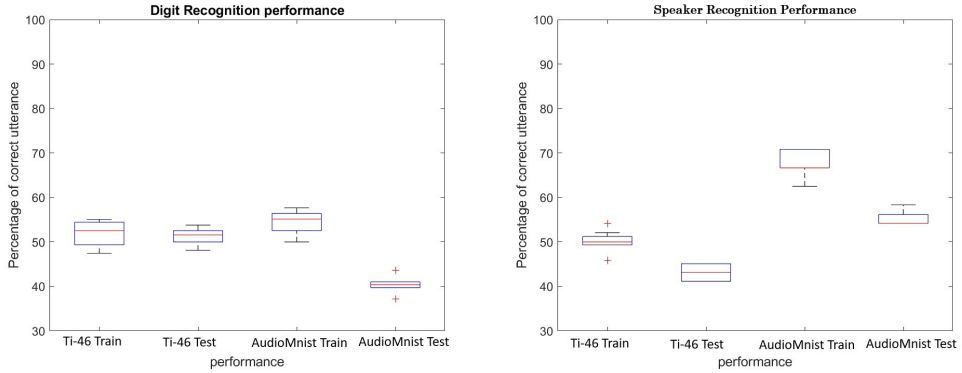
1. **Temporal Windowing:** The raw audio signal is segmented into small, defined time frames relevant to phonetic events. The raw audio signal is segmented into fixed-length frames of $N = 250$ samples. This specific window size was determined through empirical testing, as both larger and smaller windows significantly degraded classification performance.

2. **Down-Sampling:** Within each temporal window, we apply simple elementary data reduction techniques, such as max-pooling or peak-to-peak detection.

These methods efficiently select a single representative value or a small summary statistic from many samples within a window. For example, peak-to-peak detection highlights the fundamental envelope and important transient features for speech perception. These simple, almost instantaneous processing steps can be easily implemented in simple hardware. By using only these simple, almost instantaneous processing steps, we reduce data intensity without adding complexity or computational cost like spectral analysis. The resulting lower-dimensionality sequence retains the overall temporal envelope and energy profile of the original sound, preparing it for the reservoir. This

prepared signal is then fed directly into the reservoir, tasking it with the role of both a dynamic feature extractor and a classifier.

7.1.3. Performance Measurement and Visualization



(a) Digit Recognition performance (b) Speaker Recognition performance

Figure 7.1: Digit and Speaker Recognition using "feature-free" audio processing using shallow reservoir

The capacity of a single reservoir to process audio utilising the raw audio signal without any feature extraction is demonstrated by the box plots7.1.

Challenges of Raw Signal Processing

Feeding "feature-free" raw audio samples directly into a standard Echo State Network (ESN) is often suboptimal due to two fundamental constraints:

1. **Temporal Scale Invariance:** A standard ESN possesses a fixed timescale determined by its spectral radius and leakage rate. However, speech consists of multi-scale temporal events, such as transient phonemes and slower syllabic envelopes.

2. **Lack of Frequency Decomposition:** Unlike the biological cochlea, a single reservoir node cannot inherently separate a signal into frequency bands. Without this spectral analysis, the network struggles to distinguish the frequency-based patterns fundamental to audio classification.

7.1.4. From Shallow to Deep RC Raw Audio Processing

In seeking to maximize the Reservoir Computing (RC) system's capacity for "feature-free", end-to-end raw audio understanding, we extend our investigation to include the necessity and potential utility of a Deep Reservoir Computing(DRC) approach. While the canonical RC model, the Echo State Network (ESN), is inherently defined by its architectural simplicity—consisting of a single recurrent layer (the reservoir) with fixed, random internal weights—the sheer complexity involved in transforming a lengthy, high-resolution, unprocessed audio signal into a low-dimensional, discriminative, and class-separable state space suggested that the computational capacity of a single layer might prove insufficient for the task.

7.2. Sequential Deep Reservoir Architectures

This experiment investigate a Series-Connected Deep Reservoir Computing (DRC) architecture, a multi-layered framework engineered to enhance the precision of audio recognition without feature extraction stage. By moving beyond the limitations of a single-layer reservoir, this design implements a hierarchical strategy for temporal feature extraction.

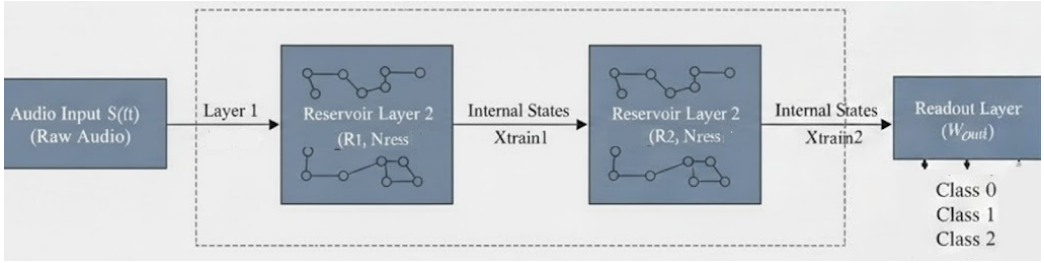


Figure 7.2: Deep series "feature-free" audio signal processing

The architectural process begins with the primary reservoir, which receives the raw acoustic input and maps it into a high-dimensional space. This first stage captures the immediate temporal dynamics of the signal. Rather than extracting a final classification at this point, the system harvests the internal hidden states of this primary reservoir. These states, which serve as a filtered representation of the initial audio, are then passed as a continuous input stream into a secondary reservoir. Figure 7.2 shows the deep series architecture.

By processing these pre-encoded states, the secondary reservoir is able to perform higher-order integration. Finally, the output of this secondary layer is fed into a trained readout mechanism for audio identification. This sequential flow effectively increases the system's fading memory and computational depth.

As illustrated in Figure 7.2 , the architecture follows a linear series configuration. The architecture connects two reservoir layers in series to process raw acoustic waveforms. Instead of extracting an intermediate classification, the primary layer (RC1) acts as a high-dimensional non-linear filter, capturing immediate temporal dynamics and passing its continuous hidden-state vectors directly as inputs into the secondary layer (RC2). Both reservoirs

update synchronously at temporal frame rate; however, their internal fading memory scales are governed independently by their respective hyper-parameters.

- **Primary Input:** Raw audio signals, represented as $S(t)$, are fed into the first layer.
- **Reservoir Layer 1:** A reservoir consisting of neurons. It processes the audio input to generate high-dimensional internal states.
- **Second Layer Input:** The internal state vectors generated by first reservoir (Xtrain1) are passed directly as the input to the subsequent layer.
- **Reservoir Layer 2:** A secondary reservoir that further transforms the temporal features provided by the first layer.
- **Readout Layer (W_{out}):** The internal states of the second reservoir (Xtrain2) are used to train the readout layer, mapping the deep temporal features to the final digit/speaker classification.
- **State Update Dynamics:** The discrete-time state transitions of the leaky Echo State Network are governed by the following non-linear activation equation:

$$X(i+1) = ((1 - S_x.\text{leak}) \cdot X(i)) + S_x.\text{leak} \cdot \tanh\left(\left(X(i) \cdot S_x.W_{\text{res}}\right) + \left(u(i) \cdot S_x.W_{\text{in}}\right)\right) \tag{7.1}$$

where $X(i)$ is the internal reservoir state at time step i, $u(i)$ is the input vector, $S_x.W_{\text{in}}$ represents the input weight matrix, and $S_x.W_{\text{res}}$ denotes the recurrent reservoir weight matrix.

- **Input Weight Scaling:** To guarantee bounded, balanced input scaling, the raw input weight matrix W_{in} is normalized and scaled symmetrically into the $[-1, 1]$ range via the min-max operation:

$$S.W_{\text{in}} = \left(\frac{W_{\text{in}} - mn}{mx - mn} \cdot 2 \right) - 1 \tag{7.2}$$

  where mn and mx denote the minimum and maximum boundaries of the input weight matrix.

- **Node Count:** The first reservoir is composed of **250 nodes** and the second reservoir is composed of **500 nodes**.

- **Spectral Radius (ρ):** The spectral radius of the recurrent weight matrix ($S_x.W_{\text{res}}$) is scaled precisely to $\rho = \mathbf{0.8}$.

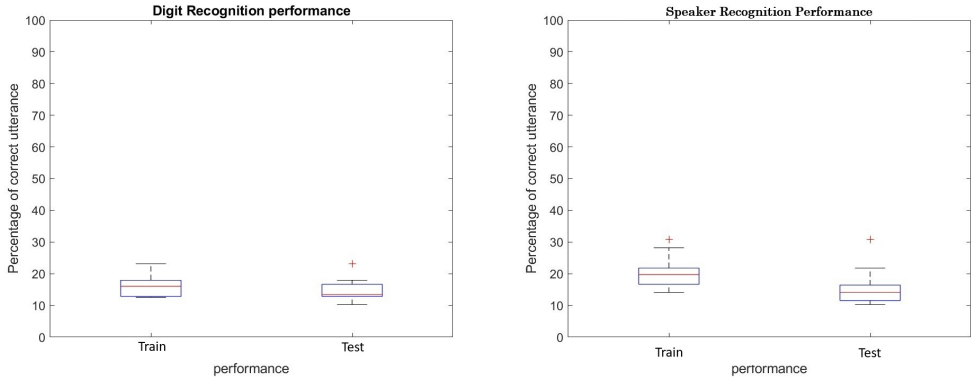
- **Sparsity:** A sparsity of **80%** is implemented (connections established with a 20% probability).

- **Leakage Rate (α):** A leakage rate (S_x.leak) for first and second reservoirs are $\alpha = \mathbf{0.3}$. This parameter controls the exponential decay rate of the nodes.

- **Washout:** A **washout period of 50 time steps** is applied. This initial portion of the state matrix is discarded during training to ensure that the linear readout utilizes only stable, fully developed internal trajectories, eliminating initialization transient effects.

7.2.1. Results for Sequential Deep RCs

The train and test performance of digit and speaker utilizing Ti-46 dataset using series deep “feature-free” method is shown in box-plots



(a) Digit Recognition performance (b) Speaker Recognition performance

Figure 7.3: Performance of series deep reservoir used for “feature-free” audio processing

This experiment serves as a preliminary evaluation of the series-connected deep reservoir architecture for “feature-free” audio signal processing. However, the performance metrics observed during testing did not meet the necessary benchmarks for viability. Consequently, the data is insufficient to support any definitive inferences or broader conclusions regarding the efficacy of this specific configuration at this time, instead as an indication that the current model requires fundamental realignment.

However, the performance discrepancies observed reveals a fundamental trade-off between feature extraction and signal degradation. When using a series-deep connection, the first reservoir functions as a high-dimensional non-linear filter, extracting low-level acoustic features from the raw audio. However, this creates an information bottleneck; by the time the data reaches

the second reservoir, it is no longer the original signal but rather the internal state of the first layer. If hyper-parameters-like the spectral radius or leak rate-are not perfectly tuned, this first stage washes out critical high-frequency details. This signal washing forces the second reservoir to learn from blurry data, explaining the poor accuracy.

The shallow model highlights the inherent difficulty of processing raw audio without pre-processing. Typically, audio signals are converted into Mel Frequency Cepstral Coefficients (MFCCs) to simplify the task. Without this assistance, a single reservoir must perform the heavy lifting of frequency analysis itself. When a Deep Series Reservoir is supplied with Mel

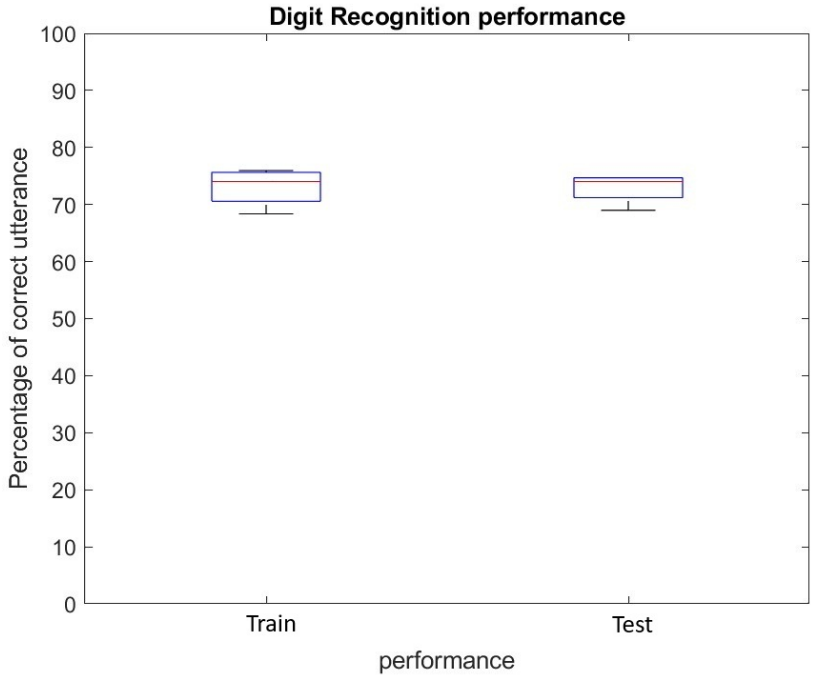


Figure 7.4: Performance of deep series RC with MFCC as input

Frequency Cepstral Coefficients (MFCCs) as input rather than raw audio, it typically yields better classification results as shown in figure 7.4, because

the MFCCs serve as a pre-conditioned, low-dimensional representation of the acoustic manifold. Raw audio signals contain high-frequency redundancies and noise that can overwhelm the stochastic dynamics of a reservoir, whereas MFCCs transform the signal into perceptually relevant spectral features that are already semi-linearly separable. This is particularly advantageous in a series-connected design because the informational smoothing effect—where important high-frequency transients are lost before reaching subsequent layers—is prevented by providing stable, pre-processed MFCC features to the first layer.

However, even in these experiments, the performance of the Deep Series Reservoir is lower compared to the performance of a shallow reservoir fed with MFCCs. This discrepancy suggests that the second reservoir may be amplifying errors or losing critical information if it does not receive a direct copy of the input signal from the first layer. To rectify this, one potential fix is to provide the second reservoir with a copy of the input to maintain signal integrity throughout the hierarchy. Based on this hypothesis, we modified the Deep Series Reservoir by providing the second layer with a direct copy of the input signal alongside the output from the first reservoir.

While this architectural adjustment yielded a noticeable improvement in performance as shown in figure 7.5—confirming that maintaining signal integrity is vital for deeper hierarchies—the results did not surpass the baseline shallow model. This persistent gap suggests that the raw audio signal, even when processed through multiple reservoir stages, lacks the discriminative power of hand-engineered spectral features.

Building on the observation that even with input copies the series-style

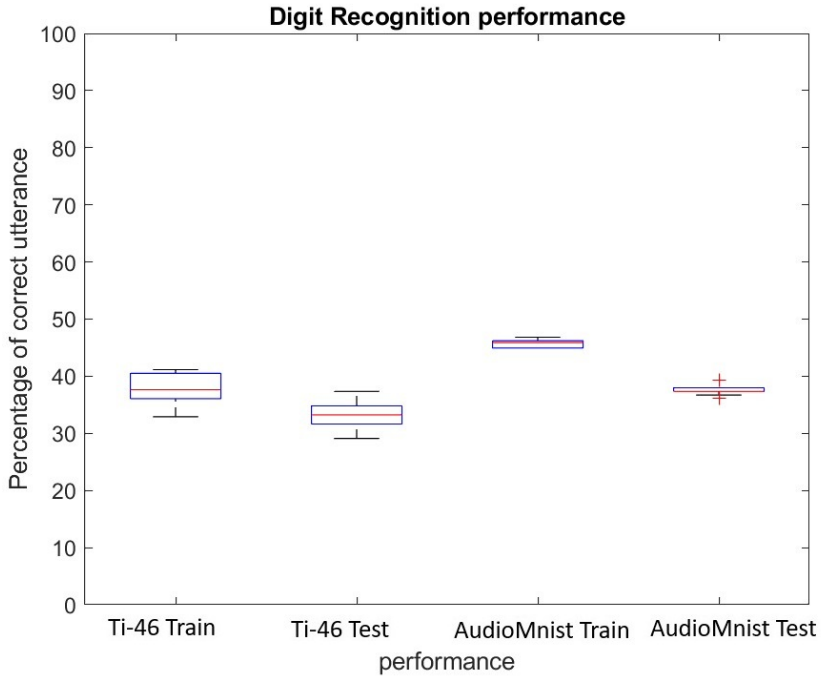


Figure 7.5: Output of series deep "feature-free" with second layer receiving a copy of Input

deep reservoir struggled, we implemented a parallel configuration. In this setup, both reservoirs receive the raw input signal simultaneously rather than in a sequential chain. By decoupling the second layer from the transformed output of the first, we ensure that the second reservoir is not forced to process potentially degraded or overly filtered information. This parallel approach allows each reservoir to act as an independently —potentially operating at different temporal scales—thereby preserving the richness of the raw audio across the entire hidden state and preventing the error propagation inherent in deep serial hierarchies.

7.3. Parallel "feature-free" Reservoir Architectures

The Parallel "feature-free" Reservoir Computing approach works more like a multi-resolution ensemble, simultaneously recording many temporal perspectives of the same raw data, whereas the sequential (series) approach imitates the hierarchical abstraction of CNNs. It underscores why moving to a parallel strategy is more effective. It allows multiple independent reservoirs to be tuned to capture different aspects of the signal, providing a more robust analysis than a single layer or a degrading series chain.

In the parallel "feature-free" architecture, the implementation of heterogeneous leak rates facilitates a multi-scale temporal analysis of the raw audio signal, which is essential for capturing the complex dynamics of speaker-specific identities. By assigning a lower leak rate (α_{low}) to one reservoir, the system establishes a slow integrator with an extended fading memory. This reservoir effectively smooths the high-frequency fluctuations of the raw waveform to capture global acoustic characteristics, such as the speaker's rhythm and long-term resonance.

Simultaneously, the second reservoir, configured with a higher leak rate (α_{high}), serves as a fast tracker. This layer is more sensitive to the immediate temporal changes in the input signal, allowing the network to preserve the fine-grained textures of speech, such as rapid consonantal attacks and pitch variations. Because these reservoirs operate in parallel, the final readout layer can simultaneously leverage both the long-term temporal context and the immediate acoustic transients. This functional diversity prevents the

information redundancy typically found in shallow models and avoids the catastrophic signal degradation observed in series-connected architectures, thereby providing a more robust and comprehensive representation of the speaker's voice directly from the time-domain input.

By diversifying α_i across the parallel layers, the model ensures that the concatenated state vector $X(t) = [x_1(t); x_2(t)]$ presented to the readout layer contains a multi-resolution profile of the acoustic signal, capturing both instantaneous and evolutionary vocal features.

Reservoir Parameters

- **Primary Input:** Raw audio signals, represented as $S(t)$, are fed into the first layer.
- **Reservoir Layer 1:** A reservoir consisting of neurons. It processes the audio input to generate high-dimensional internal states.
- **Second Layer Input:** Raw audio signals, represented as $S(t)$, are fed into the input to the subsequent layer.
- **Reservoir Layer 2:** A secondary reservoir that further transforms the temporal features provided by the first layer.
- **Readout Layer (W_{out}):** The internal states of the both reservoirs (Xtrain1 and Xtrain2) are used to train the readout layer, mapping the deep temporal features to the final digit/speaker classification.
- **State Update Dynamics:** The discrete-time state transitions of the leaky Echo State Network are governed by the following non-linear

activation equation:

$$X(i+1) = ((1 - S_x.\text{leak}) \cdot X(i)) + S_x.\text{leak} \cdot \tanh\left(\left(X(i) \cdot S_x.W_{\text{res}}\right) + \left(u(i) \cdot S_x.W_{\text{in}}\right)\right) \tag{7.3}$$

where $X(i)$ is the internal reservoir state at time step i, $u(i)$ is the input vector, $S_x.W_{\text{in}}$ represents the input weight matrix, and $S_x.W_{\text{res}}$ denotes the recurrent reservoir weight matrix.

- **Input Weight Scaling:** To guarantee bounded, balanced input scaling, the raw input weight matrix W_{in} is normalized and scaled symmetrically into the $[-1, 1]$ range via the min-max operation:

$$S.W_{\text{in}} = \left(\frac{W_{\text{in}} - mn}{mx - mn} \cdot 2\right) - 1 \tag{7.4}$$

where mn and mx denote the minimum and maximum boundaries of the input weight matrix.

- **Node Count:** The first reservoir is composed of **250 nodes** and the second reservoir is composed of **500 nodes**.

- **Spectral Radius (ρ):** The spectral radius of the recurrent weight matrix ($S_x.W_{\text{res}}$) is scaled precisely to $\rho = \mathbf{0.8}$. This setting scales the largest eigenvalue of the network to govern global stability and sustain the echo state property while preventing chaotic state divergence.

- **Sparsity:** A sparsity of **80%** is implemented (connections established with a 20% probability). This promotes diverse, decoupled temporal dynamics within the network while conserving physical resource requirements.

- **Leakage Rate (α):** To capture the multi-scale temporal dynamics inherent in speech, both reservoirs are configured with distinct time scales. First reservoir utilizes a leakage rate (S_x.leak) for of $\alpha = \mathbf{0.3}$ and second reservoir is $\alpha = \mathbf{0.4}$. This parameter controls the exponential decay rate of the nodes.

- **Washout:** A washout period of **50 time steps** is applied. This initial portion of the state matrix is discarded during training to ensure that the linear readout utilizes only stable, fully developed internal trajectories, eliminating initialization transient effects.

The parallel architecture avoids the single point of failure inherent in series models. In a parallel setup, every reservoir receives the raw audio signal directly and simultaneously. Because each reservoir is initialized with different random weights, they develop feature diversity, each noticing distinct nuances of the speaker's voice. This creates a redundancy where if one reservoir misses a specific frequency cue, another likely captures it, allowing the final readout layer to aggregate the most relevant information from all paths. This approach bypasses the bottleneck problem, resulting in an improvement over the shallow model. Figure 7.6 shows the parallel architecture.

7.3.1. Architectural Advantages of Parallel Models

The main idea behind the Parallel "feature-free" RC investigation is that the system might take use of a variety of dynamical behaviours to create a more complete representation by simultaneously feeding the raw audio input into several distinct reservoirs. The parallel design concentrates on breadth (using reservoirs with varied hyper-parameter configurations to analyse the

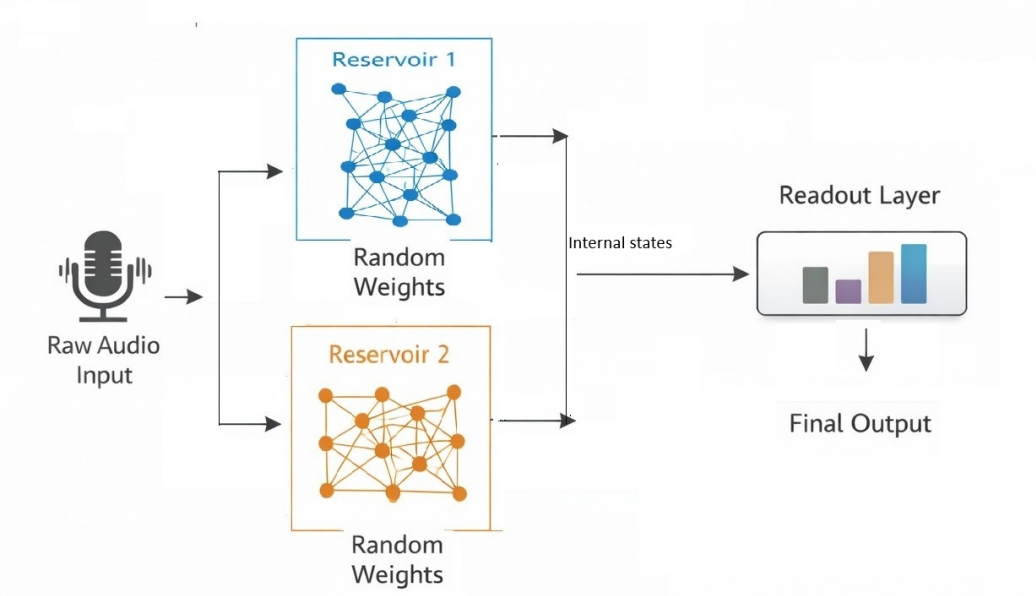


Figure 7.6: Parallel “feature-free” architecture

same signal over several temporal scales and resolutions) in contrast to the sequential paradigm, which aims to increase abstraction through depth. Using this idea for audio, we hypothesise a horizontal division of tasks in which the readout layer's feature set is enhanced by the combined dynamics of the reservoirs.

1. *Multi-Scale Temporal Integration:* By assigning different leaking rates and spectral radii to each parallel reservoir, the system can simultaneously capture distinct aspects of the audio. For instance, one reservoir can be tuned with fast dynamics to track rapid acoustic transients and phonemic transitions, while a second reservoir is tuned with slow dynamics to track fast acoustic transients and phonemic transitions.

2. *Richness and State-Space Redundancy:* Each reservoir offers a distinct non-linear projection of the raw audio since it is initialised with a dif-

ferent random internal weight matrix. When these diverse state vectors are concatenated, they create an expanded, high-dimensional feature space. This richness is governed by the principle that non-linear mappings into higher-dimensional spaces increase the likelihood of data becoming linearly separable. By providing the readout layer with a vast library of distinct projections, the system can more easily isolate specific signal components that might be indistinguishable within the limited state-space of a single-reservoir ESN.

3. *Signal Integrity and Memory Resolution:* The parallel architecture is fundamentally superior for raw audio processing because it expands the system's Memory-Resolution in a way that shallow and serial models cannot. Parallel reservoirs each retain direct, unfiltered access to the original input, in contrast to serial (deep) architectures where the raw signal is gradually filtered and distorted as it moves through subsequent layers. This ensures that acoustic features lost in one branch are captured in another, avoiding the information decay or cumulative non-linear distortion common to deep stacks. By merging these specialized, high-fidelity projections at the readout, the parallel approach achieves a multi-resolution analysis that captures the full hierarchical complexity of audio without sacrificing signal integrity.

By merging these parallel dynamics, the RC is theoretically capable of a multi-resolution analysis of raw audio, ensuring that both high-frequency localized features and long-term temporal context are preserved and available for the final identification task.

7.3.2. Implementation Challenges in Parallel RC

While the parallel architecture bypasses the signal propagation instabilities found in sequential stacking, it introduces its own set of non-trivial implementation hurdles:

1. *Dimensionality and Computational Overhead:* The primary challenge in parallel RC is the curse of dimensionality. Because the states of all reservoirs are concatenated before reaching the readout, the size of the output weight matrix grows linearly with the number of reservoirs. This significantly increases the memory footprint and the computational cost of the training phase, which scales cubically with the total number of internal nodes.

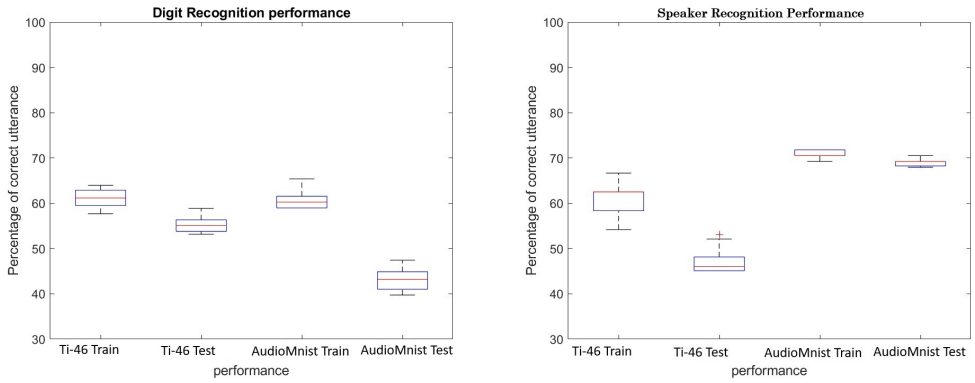
2. *The Challenge of Diversity Enforcement:* For a parallel architecture to be effective, the reservoirs must be sufficiently different from one another. If the hyper-parameters (spectral radius, input scaling, sparsity) are too similar, the parallel branches produce highly correlated states, leading to redundancy without any gain in predictive power. Finding the optimal spread of hyper-parameters to ensure complementary dynamics requires a more nuanced search than tuning a single reservoir.

3. *Feature Normalization:* Integrating dynamics from reservoirs with vastly different scales can lead to numerical instability in the readout. A reservoir with high-amplitude activations might drown out the subtle but critical information provided by a more dampened reservoir. This necessitates careful feature scaling or regularization strategies to ensure

that the linear readout can effectively weigh the contributions of each parallel branch.

7.3.3. Performance Measurement and Visualization

The box plots7.7 show how a deep reservoir may process audio using the raw audio data without any feature extraction.



(a) Digit Recognition performance (b) Speaker Recognition performance

Figure 7.7: Digit and Speaker Recognition using deep "feature-free" parallel audio processing using reservoir

Experiments	Digit Recognition				Speaker Recognition			
	Ti-46		AudioMnist		Ti-46		AudioMnist	
	Train	Test	Train	Test	Train	Test	Train	Test
Shallow feature-free Audio processing	53.06	51.22	56.98	41.79	50.27	44.02	68.11	58.28
Parallel feature-free Audio processing	61.98	56.86	61.81	43.44	63.98	46.32	72.89	71.21

Table 7.1: Comparison of different methods for Digit and Speaker Recognition

Table 7.1 summarises the result.

The architectural shift from shallow to deep configurations reveals that while serial connections often suffer from signal washing and initialization instability, parallel reservoirs provide a more robust framework for capturing

diverse acoustic features. By distributing raw audio across independent reservoirs simultaneously, Parallel "feature-free" Reservoir Computing (PFRC) bypasses the information bottlenecks and fine-tuning inherent in stacked layers. Chapter 8 table 8.2 provides a more detailed comparison of this method with other methods used in the thesis.

7.4. Summary and Conclusion

In short, while a single-reservoir approach reduces architectural complexity, it often lacks the representational depth required for peak performance. PFRC overcomes this limitation by decoupling signals across multiple pathways, delivering a comprehensive analysis of the raw audio. Ultimately, this demonstrates that Reservoir Computing can function as a self-contained processor that projects signals into a higher-dimensional space, enabling audio processing without the need for traditional handcrafted features.

Despite these advantages, this framework exhibits significant instability and inconsistency. The model's performance is highly sensitive to minor fluctuations in hyper-parameters and variations in input data, leading to high variance in output results. This stochastic behaviour suggests that while the PFRC offers an alternative to traditional complex feature extraction, further research is required to stabilize its dynamics and ensure reliable, reproducible performance across diverse acoustic environments.

8 Discussion

8.1. Reservoir as Classifier

We are training the reservoir as a classifier with the MFCC obtained using Matlab function to evaluate its capability in categorization tasks and address the validation of Sub-Hypothesis 1. This hypothesis states that 'A Reservoir Computing (RC) based classifier achieves comparable performance to conventional classifiers when utilizing MFCC features.' To test this, the reservoir is implemented with a lean architecture of only 100–400 nodes, representing a reduction in computational complexity by several orders of magnitude compared to traditional neural networks that typically demand significant temporal and energy resources.

The experimental results demonstrate the reservoir's capacity to carry out classification and regression tasks. Given the complexity of audio analysis, any neural network that does audio classification often needs a large number of nodes. The efficient use of resources makes the approach particularly well-suited for applications where computational resources, power, and time are constrained.

The results demonstrate that the RC-based classifier achieves superior performance when integrated with conventional feature extraction, validating

its robustness as a back-end engine. This efficiency underscores the model's practical potential, proving that high accuracy in complex tasks like audio processing does not necessitate massive, energy-intensive models. Such an approach is particularly well-suited for applications where computational resources, power, and time are constrained. By demonstrating that the reservoir achieves performance parity with conventional models while using significantly fewer resources, the reservoir's robust categorization capabilities were confirmed. Thus, Sub-Hypothesis 1 is validated.

8.2. Feature Extraction by Mimicking MFCC Function

In our investigation, we are utilizing a reservoir to mimic the MFCC extraction and validate our Sub-Hypothesis 2 which states that 'Reservoir Computing can successfully replicate key MFCC features, simplifying the feature extraction process exploiting time-domain operation.' To prove that reservoir dynamics can replicate MFCC features, which significantly simplifies the feature extraction pipeline, we are modelling reservoir to mimics MFCC extraction. The low NRMSE values for most coefficients demonstrates the reservoir's capability to perform this task. We utilize the output of feature extraction reservoir(first reservoir-RC-1) as the input to the Classification reservoir (second reservoir RC-2) and the resulting output is shown as box plots. The plot shows the reservoir's capability in mimicking MFCC feature extraction. The outcome of our method is comparable to the result of Matlab based MFCC.

We have found that the window length we consider has a significant impact on the performance of the system. Based on this finding, we are analysing three scenarios.

As explained earlier, to calculate the number of non-overlapping windows, we use the equation $N = (M) + 2)/2$, where M is the number of MFCC samples. In the first method (baseline window experiment), we are using the window length obtained based on this equation. In the second method (dense window experiment), we have increased the number of overlapping windows by decreasing the window size. This method performs better than the first method. In the third method (tuned window experiment), we have increased the number of overlapping windows by decreasing the window size for higher frequencies(Higher Mel Coefficients) for speaker identification and increased the number of overlapping windows by decreasing the window size for lower frequencies(Lower Mel Coefficients) for digit identification.

This method's primary benefit is that it reduces computing complexity, particularly when executing the Fourier Transform, by doing away with the necessity for intricate time-frequency conversion. It also facilitates the implementation of a real-time audio signal processor, which is another significant benefit. A third benefit is that, by avoiding sophisticated computations, the system may be implemented with a basic hardware reservoir, leading to a processing system that can easily duplicate the MFCC extraction. This reduces the need for more complex systems.

In terms of speaker recognition, the outcome of our method is comparable (or even better) to the result of Matlab based MFCC. In dense and tuned window experiment, we are able to improve the performance of speaker recog-

nition compared to baseline window experiment. Digit recognition is more challenging than speaker recognition. In dense and tuned window experiment we are able to improve the performance of Digit recognition. We have also found that the performance is significantly impacted by the window size. In short, our approach to MFCC extraction is effective and seems promising. Thus, Sub-Hypothesis 2 is achieved.

8.3. Feature Extraction by Convolution

In this experiment we are trying to validate the sub-Hypothesis 3 which states that 'The effectiveness of reservoir-based feature extraction can be improved by training it on MFCC-inspired convolutions.' To prove this, we use a reservoir to obtain the time domain Mel Frequency Convolution Filter (MFCF) audio feature by convoluting the audio signal and Mel Filter-Bank in time domain. We utilize the output of this first reservoir that extract the audio feature in time domain as the input to the second reservoir, and the resulting output is shown as box plots.

The extraction of MFCF in time domain utilizing convolution is done in 2 method. In first method we are using a single reservoir for feature extraction. The reservoir used will have 14 output layer each individually trained to generate each of the Mel Coefficient.

In the second method we employ 14 distinct reservoirs, each reservoir is dedicated in extracting a single, specific MFCF. So, there will be 14 parallel and independent reservoirs for each of the MFCF feature extraction.

A significant merit of this method is its demonstrated ability to achieve comparable or even superior performance in the challenging task of digit

recognition when bench-marked against time domain mimicking method, all while operating exclusively in the time domain. The core reason for this success lies in the nature of Reservoir Computing (RC) itself: a reservoir is natively optimized to mimic time-domain dynamics rather than operations that require multiple domain transformations (such as moving from time to frequency). The evidence for this is found in the performance gap observed across different methodologies. When the reservoir is used to mimic the standard Matlab MFCC extraction (Chapter 5), which involves multiple domain transfers, the performance gap is significantly wider because the reservoir struggles to replicate frequency-domain logic. In contrast, when the reservoir mimics the MFCF extraction via time-domain convolution (Chapter 6 and 'MFCF extraction Via time-domain convolution' in Appendix B), the gap is minimized. Because this second method performs all processing natively in the time domain without domain switching, it aligns perfectly with the reservoir's inherent strengths, demonstrating that RC reaches its highest efficiency when it operate in time domain.

This approach entirely bypasses the computational overhead and potential information loss associated with time-frequency transformations like the Fast Fourier Transform (FFT). It underscores the efficacy of the specialized reservoir architecture, proving that its precision in generating clean, independent Mel inspired features directly from the raw signal translates into robust performance for the most demanding classification tasks without requiring a frequency-domain intermediate.

Table 8.1 shows a comparison of the number of neurons used by different audio signal processing methods. The number of neurons in a network

Model Architecture	Task / Dataset	Test %	Parameters/Complexity
CNN	Audio-MNIST (Digit Recognition)	98.63	2M–10M parameters
Word embedding	Spoken Language Identification	92.20	1M–5M parameters
ResNet50	Speech Classification Baseline	80.20	∼5M parameters
CapsNet	Speech Command Profiling	88.76	10M–20M parameters
CNN LSTM	Continuous Speech Processing	80.52	5M–15M parameters
SVM	Acoustic Token Classification	83.32	Support Vector dependent
Log. Regression (137)	Parameter Scaling Baseline	80.23	1M parameters
Acoustic model	Phonetic Feature Extraction	73.23	Dataset dependent
RF (VGG16)	Speech Feature Mapping	71.90	10K–100K trees
LSTM-CNN (138)	Spoken Language Identification	68.33	5M–15M parameters
2D ConvNet Bi-GRU	Speech Emotion Recognition (SER)	65.23	10M–20M parameters
Extended LSM (ELSM) (117)	TIDIGITS (Isolated Digit Speech)	86.3	2000 Spiking Neurons
RC (Matlab MFCC)	**TI-46 / Audio-MNIST (Digit)**	**98.30**	**400 Neurons**
RC (Mimic MFCC)	**TI-46 / Audio-MNIST (Digit)**	**73.89**	**RC1: 950 / RC2: 400**
RC (Time Domain)	**TI-46 / Audio-MNIST (Digit)**	**94.33**	**RC1: 450 / RC2: 400**

Table 8.1: Performance measures obtained from various language identification and speech processing techniques (137) (139) (140) (117)

is calculated based on architecture implementation and hyper parameters such as the number of hidden layers, the number of units (neurons) in each layer, and the input/output dimensions. As can be seen from the table, Our method is the most lightweight and effective model, achieving high accuracy with smaller number of neurons. By utilizing significantly less parameters while maintaining the audio feature extraction in the time domain, our

method demonstrates good performance. The refinement of feature extraction through MFCC-inspired convolutions showed that the effectiveness is notably enhanced by the reservoir based method. Thus, Sub-Hypothesis 3 is achieved.

8.4. “Feature-Free” Audio Processing

A primary objective of this study is to determine if a Reservoir Computer (RC) could process raw audio signals directly without a feature extraction stage and thus prove our sub-hypothesis 4, which states that ’A reservoir-based architecture, operating directly on the raw audio waveform, can capture sufficient discriminative acoustic information, eliminating the need for separate feature extraction stage.’ Here we are trying to bypass the complex, domain-specific feature extraction stages standard in audio pattern recognition and conducted experiments feeding raw waveforms into both single-layer and hierarchical reservoir architectures to prove this hypothesis.

We are particularly deliberate in our choice of data reduction techniques. To address the high dimensionality of raw audio while preserving the temporal essence of the signal, we utilise simple methods such as absolute max pooling and peak-to-peak reduction. We have avoided more computationally intensive or transformative methods, as we believe such complicated pre-processing can tamper with our core objective: evaluating the intrinsic capacity of the reservoir to handle raw temporal dynamics. By sticking to a streamlined reduction process, we ensure that any success in pattern recognition is attributable to the reservoir’s dynamical states rather than the complexity of a feature-engineered front end.

8.4.1. From Shallow to Deep Architectures

The architectural evolution from shallow to deep configurations reveals a critical trade-off between structural complexity and signal integrity. A shallow reservoir serves as the initial baseline; however, without the aid of preprocessing, a single-layer system lacks the precision required for complex frequency analysis. While Deep Reservoir Computing (DeepRC) theoretically offers a hierarchy of temporal representations—where layers capture a spectrum of fast transients and slow global patterns—the implementation method significantly dictates success.

In our experiments, a series-deep connection yielded underwhelming results due to the phenomenon of signal washing. In this configuration, the first reservoir acts as a high-dimensional non-linear filter that often obscures high-frequency details like pitch and timbre. Consequently, subsequent layers receive a blurred internal state rather than the original audio features. While precise calibration of the spectral radius and leak rate could theoretically mitigate this degradation, the fine-tuning process is notoriously tedious and highly sensitive to minor adjustments.

The parallel model bypasses the information bottlenecks inherent in serial chains. By feeding the raw audio to multiple independent reservoirs simultaneously, the architecture fosters feature diversity. This redundancy ensures that the readout layer can aggregate a broad spectrum of acoustic cues, capturing nuances that a single or stacked reservoir might overlook. The parallel reservoir architecture offers notable robustness and feature diversity,

8.4.2. Evaluation of Actual Performance and Findings

While our parallel “feature-free” architectures demonstrate that raw audio waveforms can be processed directly to capture discriminative acoustic information, an evaluation of the empirical results reveals substantial performance trade-off. This approach serves as a highly promising, hardware-lean proof of concept; however, its classification accuracy remains strictly lower than our alternative pipelines.

Specifically, when benchmarked against configurations where the reservoir performs analytical time-domain convolutions, the “feature-free” framework experiences a noticeable drop in absolute accuracy. This discrepancy demonstrates that while simple absolute max pooling and peak-to-peak reduction preserve fundamental global structures, they discard the fine-grained spectral boundaries necessary to extract essential information from the audio signal. Furthermore, because the analytical time-domain convolution architecture itself maintains a highly optimized, low-overhead computational complexity, the marginal savings offered by the “feature-free” framework do not justify such a significant compromise in classification performance.

Consequently, Sub-Hypothesis 4 is only partially validated. A “feature-free” parallel reservoir architecture can autonomously isolate acoustic patterns from raw waveforms, successfully eliminating the rigid requirement for a separate feature extraction front end; yet, it does so at the expense of peak classification performance.

Furthermore, we observed a persistent challenge regarding model instability. Small variations in weight initialization or hyper-parameter drift can trigger unpredictable fluctuations in the reservoirs’ internal states, poten-

tially compromising the long-term reliability of the readout output. This susceptibility to architectural instability highlights a fundamental limitation of relying solely on structural scaling to compensate for the absence of clean acoustic features.

8.5. Inference

Table 8.2 presents a comparative performance analysis of various audio processing and recognition methods across two primary tasks: Digit Recognition and Speaker Recognition. The evaluation is conducted using two benchmark datasets, Ti-46 and Audio-Mnist, with results broken down into Training and Testing accuracies.

	Digit Recognition				Speaker Recognition			
	Ti-46		AudioMnist		Ti-46		AudioMnist	
	Train	Test	Train	Test	Train	Test	Train	Test
Reservoir as classifier	100	99.37	98.24	98.09	79.43	72.15	79.90	79.38
Reservoir-based MFCC by Mimicking	78.96	61.29	74.34	60.82	80.88	70.88	91.87	84.89
Reservoir-based MFCC by Convolution	97.15	91.82	82.06	70.83	95.28	79.17	92.71	78.57
Shallow feature-free Audio processing	53.06	51.22	56.98	41.79	50.27	44.02	68.11	58.28
Deep feature-free Audio processing(parallel)	61.98	56.86	61.81	43.44	63.98	46.32	72.89	71.21

Table 8.2: Comparison of different methods for Digit and Speaker Recognition

The analysis of our experimental results leads to a definitive conclusion regarding the architectural trade-off between depth and efficiency. While increasing the hierarchy of a Deep Reservoir Computing system offers incremental improvements in capturing multi-scale dynamics, it simultaneously introduces significant structural complexity and hyper-parameter sensitivity. We propose that the optimal path forward is not found in further complicating the reservoir's depth, but in implementing a simplified, reservoir-based

time-domain feature extraction front end.

Our approach is centred on the principle of time-domain convolution. By configuring the reservoir to function as a bio-mimetic ear, we facilitate the extraction of Mel Frequency Cepstral-like features without ever leaving the temporal domain. This provides the critical frequency decomposition required for high-accuracy speech recognition while entirely bypassing the computational latency and memory overhead of the Fourier Transform.

Ultimately, our findings demonstrate that a time-domain front end leverages the reservoir's natural strengths—specifically its fading memory and non-linear high dimensional mapping—to perform the heavy lifting of feature engineering. This strategy avoids the structural overhead of massive deep reservoir hierarchies as well as complex computation, offering a superior balance between computational simplicity and classification robustness. We conclude that for real-time audio analysis, a shallow classifier (RC) coupled with a time-domain convolutional front-end reservoir provides the most efficient, hardware-friendly architecture for next-generation auditory processors.

The successful validation of these four pillars confirms the overarching hypothesis: 'an end-to-end Reservoir Computing approach is fully capable of performing all essential signal processing functions—from feature extraction to classification—within a unified, time-domain audio processing system.'

9

Conclusions and Future Work

9.1. Conclusions

The transition from traditional, power-hungry digital signal processing to the next generation of energy-efficient, real-time audio systems represents a critical frontier in modern computational research. For decades, the industry has relied on frequency-domain transformations such as the Short-Time Fourier Transform and Mel Frequency Cepstral Coefficients to bridge the gap between raw sound waves and machine understanding. While these methods are mathematically robust, they are inherently expensive in terms of computational cycles and energy consumption, largely because they require the conversion of time-series data into a different mathematical realm before any meaningful analysis can begin. This creates a significant bottleneck for embedded systems, wearable medical devices, and always-on voice assistants that must operate within strict battery constraints. To address this bottleneck, this thesis introduces a comprehensive and novel framework that simplifies audio and speech signal processing by leveraging the unique dynamics of Reservoir Computing in combination with time-domain techniques. By bypassing the traditional reliance on frequency-domain analysis, we offer a biologically inspired alternative that processes raw waveforms with high

efficiency, maintaining both the temporal precision and spectral sensitivity necessary for advanced speech recognition tasks.

At the heart of this framework is Reservoir Computing, a paradigm shift in the way we utilize recurrent neural networks. Conventional recurrent networks often struggle with high training costs and vanishing gradients because every connection within the network must be optimized. In contrast, a reservoir consists of a fixed, randomly connected dynamical system that maps input signals into a high-dimensional state space. Because the internal recurrent weights are never changed during the learning process, the only component that requires training is a simple linear readout layer. This architecture provides several immediate advantages: it is incredibly fast to train, requires minimal memory overhead, and possesses an inherent fading memory that makes it naturally suited for time-varying signals like audio. Our work investigates the reservoir not merely as a classifier, but as a powerful non-linear feature extractor. By exploiting the intrinsic dynamics of the reservoir, we can transform raw, oscillating audio signals into rich, informative representations without the need for handcrafted features or explicit spectral transforms. This effectively bridges the gap between signal transformation and learning, creating a unified perspective where the reservoir serves as a seamless interface between the physical world of sound and the digital world of classification.

Beyond traditional methodologies, this research investigated the viability of end-to-end audio analysis. To rigorously validate this approach, we conducted a series of distinct experiments designed to test the versatility of Reservoir Computing across different levels of complexity. The first experi-

ment investigated the lower bounds of hardware requirements by using the reservoir as a stand-alone classifier. We were able to demonstrate that a network with as few as 100 nodes could achieve competitive accuracy on speech recognition benchmarks. This result is particularly significant for the field of edge computing, as it proves that high-level intelligence can be packaged into extremely small architectures. When we consider that modern deep learning models often require millions or even billions of parameters, the ability of a 100-node reservoir to distil speech into actionable data underscores a level of efficiency that is orders of magnitude beyond current standards. This minimalist approach allows for the deployment of speech intelligence on low-cost microcontrollers that were previously thought to be too limited for such tasks.

The second experiment addressed the most common objection to time-domain processing: the potential loss of spectral information. Because humans perceive sound on a non-linear scale—specifically the Mel scale—most speech systems use MFCCs to mimic the human ear's sensitivity. We challenged the reservoir to mimic this conventional feature extraction process. Our findings revealed that the reservoir's dynamic response can naturally approximate MFCC-like representations. By projecting the raw waveform into a high-dimensional non-linear space, the reservoir inherently captures the relevant spectral characteristics usually reserved for frequency-domain mathematics. This mimics the logarithmic compression and filter-bank analysis of traditional pipelines but does so through the natural physics of the dynamical system. This experiment confirmed that we do not need to tell the system how to hear like a human; the reservoir's dynamics can learn to

approximate those perceptual models on their own.

Building on this success, the third experiment involved the implementation of an MFCC-like pipeline entirely within the time domain. Rather than using an FFT to generate filter banks, we utilized convolution with synthesized Mel Filters applied directly to the raw waveform. This method allows the system to operate on a sample-by-sample basis, maintaining a level of temporal precision that is often lost in the windowing process required by Fourier Transforms. By eliminating the FFT stage, we significantly reduce the processing latency and the mathematical complexity that negatively affect real-time responsiveness of the system (141). Furthermore, this approach leverages the inherent flexibility of Reservoir Computing (RC); unlike traditional architectures restricted to a single, rigid task, the RC framework can adaptively process continuous streams for multiple concurrent objectives (142), (143). To further optimize the system, we propose that energy efficiency can be significantly increased by utilizing a Liquid State Machine (LSM) rather than an Echo State Network (ESN), as the spiking nature of LSMs offers a more power-frugal alternative for temporal data (144), (145). This time-domain implementation proves to be a more efficient architecture, opening the door for real-world applications where instantaneous response is required, such as in high-fidelity hearing aids or industrial acoustic monitoring systems.

We also explored the possibility of direct audio analysis bypassing the need for intensive manual feature extraction. Our findings indicate that this streamlined approach is not only viable but highly promising; specifically, we demonstrated that the integration of deep reservoir architectures significantly

enhance performance of "feature-free" audio signal processing.

Unlike conventional deep learning approaches that require massive datasets and extensive training, Reservoir Computing offers a lightweight alternative that is ideally suited for neuromorphic hardware. The fixed structure of the reservoir draws direct parallels to the auditory processing found in the human brain, specifically the way the cochlea acts as a mechanical, time-domain filter bank. The efficiency gained by avoiding FFT computations and reducing network complexity directly translates to longer battery life and smaller device footprints.

An important contribution of this work is the demonstration that complex, handcrafted feature extraction pipelines are not strictly necessary for effective speech processing. By leveraging the inherent dynamics of Reservoir Computing, we show that raw audio waveforms can be transformed into informative representations suitable for classification and recognition tasks. This challenges the traditional reliance on frequency-domain analysis and suggests new directions for the design of audio processing systems. Our framework provides a unified perspective on time-domain feature extraction and classification. Rather than treating feature extraction and classification as separate stages, the reservoir serves as a bridge between the two, seamlessly integrating signal transformation and learning. This end-to-end approach simplifies system design and reduces the need for domain-specific expertise in feature engineering.

In conclusion, our study highlights the substantial potential of Reservoir Computing for speech and audio processing tasks. This approach simplifies system design, reduces the need for domain-specific expertise in fea-

ture engineering, and significantly lowers the barrier for implementing high-performance audio intelligence on the edge. As the demand for intelligent, low-power solutions continues to grow, the integration of time-domain processing and Reservoir Computing stands out as a versatile paradigm. This research provides a foundation for future bio-inspired computing systems that process sound with the same elegance and efficiency as the human ear and brain, paving the way for a world where technology listens more effectively while using less power.

9.2. Future Work

RC as a Unified Feature Extractor in Time Domain

The success of generating Mel-Frequency Cepstral Coefficients (MFCCs) directly within the time domain using Reservoir Computing (RC) opens a new frontier for audio signal processing. Traditionally, features such as Wavelet Transforms, Lyon's Cochlear Model, etc were developed using complex frequency-domain transformations primarily to bypass the historical computational burden of high-dimensional convolution. These features can be easily generated using reservoir in time domain using the same concept. An attempt to develop wavelet Transform with Mel Filters is detailed in appendix as an example. (Appendix A)

Future research will focus on expanding this concept to recreate diverse bio-inspired models—such as Lyon's Cochlear model or such audio signal features within a single RC architecture. This approach eliminates the need for expensive domain-switching overhead, offering a streamlined pathway for

real-time audio analysis on low-power edge devices and neuromorphic hardware. By treating the reservoir as a self-contained engine for implicit convolution, we aim to demonstrate that traditional mathematical transformations can be surpassed by the efficient, temporal processing capabilities of RC.

Toward Hardware-Integrated, Low-Power Auditory Intelligence: Neuromorphic Ears

Another critical and promising area of exploration in modern artificial intelligence is the implementation of Reservoir Computing (RC) in physical hardware, particularly as a means to revolutionize energy-efficient audio signal processing. As intelligent systems increasingly migrate from centralized computing environments to edge devices, wearables, and embedded platforms, the limitations of conventional digital signal processing and deep learning approaches have become increasingly apparent. These methods, while powerful, often rely on high computational complexity, extensive training, and significant energy consumption, making them ill-suited for continuous, real-time auditory processing under strict power constraints. Reservoir Computing offers an alternative paradigm that aligns computation with the intrinsic dynamics of physical systems, opening a pathway toward next-generation auditory intelligence that is both efficient and biologically inspired.

Reservoir Computing is uniquely well suited for audio signal processing because it decouples complex temporal processing from costly training procedures. In RC, a fixed, high-dimensional dynamical system —the reservoir— transforms incoming temporal signals into rich internal representations, while only a simple readout layer is trained. This architectural simplicity enables

reservoirs to be implemented directly in hardware using analogue circuits, neuromorphic substrates, memristive devices, photonic systems, or other physical media.

The development of a more efficient audio processing pipeline using Reservoir Computing (RC) motivates me to transition these models directly into specialized neuromorphic hardware. Unlike traditional digital signal processing, which relies on power-hungry spectral transformations and deep layers of trained weights, RC allows for a fixed, high-dimensional dynamical system where only the readout layer requires optimization. By mapping this architecture onto a neuromorphic substrate, the temporal dynamics of the audio signal translate directly into the hardware, allowing computation to emerge from the system's intrinsic physics rather than through resource-heavy digital emulation.

This implementation aims to verify the performance of a fully on-chip audio reservoir, targeting the high-speed, low-power requirements of sensing. By leveraging the event-driven parallelism and fading memory inherent to neuromorphic devices, the system can capture complex temporal patterns and acoustic biomarkers in real-time. This approach inherently operates in the time domain, mirroring biological auditory processes and providing a robust solution for dynamic environments where, latency and energy efficiency are critical.

The concept of neuromorphic ears encapsulates this vision. By embedding intelligence directly into physical substrates, neuromorphic ears transform auditory perception from a resource-intensive computational task into an efficient, embodied process. By embedding intelligence directly into the

hardware, we can reframe advanced auditory processing as a pervasive and invisible utility. Such a system would enable continuous health monitoring, responsive brain-computer interfaces, and ambient smart environments that function seamlessly within strict power budgets, making sophisticated auditory intelligence more sustainable and accessible for edge applications.

Temporal Integration as the Foundation of Artificial Perception

While audio signal processing provides a compelling and practical focus, this line of research also highlights the need to move beyond audio and address a more fundamental question concerning the future of artificial perception. Perception is fundamentally temporal rather than modality-specific. The integration of temporal information is essential to the development of sound, vision, touch, and other senses. Instead of complicating each individual sense with its own heavy, separate architecture, we are building a unified way for AI to process information, creating systems that are more natural, efficient, and cohesive.

Reservoir Computing provides a path toward a more unified and all-purpose framework for perception by utilising time as a universal computational currency. A single reservoir-based architecture can project different temporal inputs into a high-dimensional space, from which significant patterns may be recovered using straightforward readouts, eliminating the need to create specialised algorithms for every single sense. This method promotes richer and more effective representations of the world by streamlining system design and facilitating seamless integration all sensory modalities. This expansion of time-domain processing is a significant step toward artificial

perception that is as effective and comprehensive as human perception.

In short, the exploration of Reservoir Computing in physical hardware for signal processing represents a foundational shift in how intelligent systems are conceived and realized. By simplifying audio processing to align with hardware-compatible reservoirs and integrating these systems with neuromorphic engineering, we can create energy-efficient systems that approach biological performance. More broadly, by embracing time as the unifying principle of perception, we are pointing toward a unified framework for artificial intelligence. This brings artificial perception closer to the depth and power of human experience by establishing the foundation for perceptual systems that are effective, flexible, and deeply interconnected with the real environment.

Appendix A : Combining MFCC and Wavelet Features in Time-Domain

Feature extraction is arguably the most critical process within the field of speech signal processing. The selection of an effective feature set directly dictates the performance and robustness of downstream applications, such as speaker identification and automatic speech recognition. For decades, the Mel Frequency Cepstral Coefficient (MFCC) has reigned as the dominant feature, owing its popularity to its effective resemblance to the filtering mechanisms that occur within the human auditory system. The fundamental strength of MFCCs lies in their capacity to provide a compact, robust representation of the signal's frequency content. By mapping the linear frequency spectrum onto the non-linear Mel scale, MFCCs successfully emphasize the crucial low-frequency bands where most speech energy and phonetic distinctions reside, aligning well with human perception of pitch. However, the primary drawback of MFCCs stems from their inherent time-averaging nature. The traditional extraction process, which involves a Discrete Fourier Transform (DFT) followed by the calculation of the log energy within static Mel Filter-Banks, provides only the frequency information of the signal but fails to adequately capture the precise temporal location of those frequencies.

A.1. Mel Frequency Wavelet Coefficient (MFWC)

The lack of fine-grained time-frequency resolution makes MFCCs less effective at analysing rapidly changing or non-stationary signals like speech, particularly when dealing with transient sounds, stop consonants, or sudden noise bursts. To overcome this limitation, researchers turned to tools with more flexible analysis capabilities, most notably the Wavelet Transform (WT). The WT is an appropriate tool for the analysis of non-stationary signals because it utilizes a flexible time-frequency window. This means the WT can provide both time and frequency information of the signal simultaneously, offering high-frequency resolution for low frequencies and high-time resolution for high frequencies. Despite its inherent strength in time-frequency analysis, the typical Wavelet Transform presents its own challenge when applied directly to speech. Standard Wavelet Transform often employ a uniform frequency scaling, which runs counter to the non-linear way the human ear perceives sound. This can result in poorer frequency resolution in the crucial low-frequency range and make the feature set less in line with human auditory perception than MFCCs. The recognition of these complementary strengths and weaknesses led to a concerted effort to develop a feature that incorporates the merit of both MFCC and the Wavelet Transform. This synthesis resulted in several techniques centred around Wavelet-based MFCCs, all attempting to leverage the Mel Frequency Cepstral Coefficient scale (mimicking human hearing perception) with the time-frequency resolution capabilities of the Wavelet Transform. The resulting feature, often termed the Mel Frequency Wavelet Coefficient (MFWC), offers advantages by ensuring

frequency analysis is perceptually relevant while simultaneously allowing for time-localized analysis of the signal's frequency components, enabling better analysis of transient sounds and noise variations within a speech signal (146).

A.1.1. Addressing Computational Complexity with Time-Domain Processing

In the state-of-the-art methods for extracting Wavelet-based Mel frequency coefficients, the process is inherently complex due to the sequential nature of the two major operations. Existing approaches typically involve applying the Wavelet Transform (WT) either prior to the MFCC part or after the MFCC part. In either conventional scenario, the entire calculation steps of both MFCC and the Wavelet Transform are required to obtain the Wavelet Transform-based Mel-scaled feature extraction. This sequential, multi-step dependency makes the whole process computationally complicated, demanding significant resources for the successive domain conversions and complex mathematical operations.

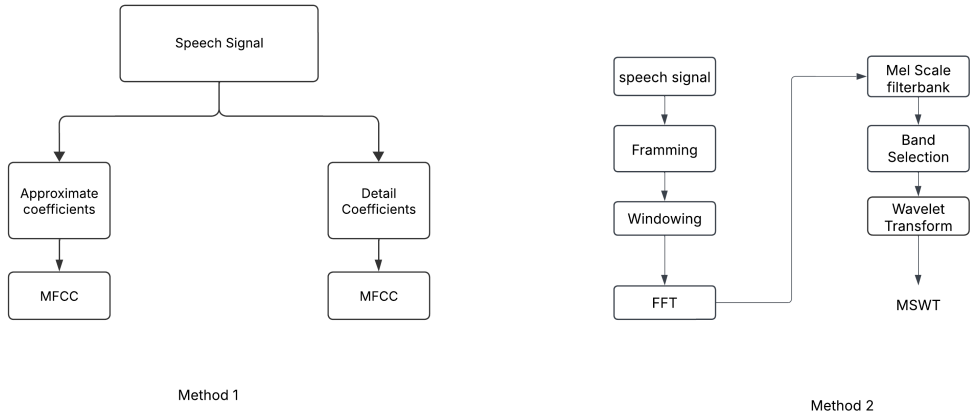


Figure A.1: Wavelet based Mel Frequency Coefficient extraction methods

Our research introduces a paradigm shift in MFWC extraction designed to radically reduce this complexity and increase efficiency. The core of our innovation lies in moving away from the conventional frequency-domain and sequential processing towards a time-domain feature extraction method. To achieve this, we harness the unique capabilities of Reservoir Computing (RC), an efficient framework derived from recurrent neural networks. The feasibility of this simplification rests entirely on the fact that the chosen implementation of Reservoir Computing (specifically, the feature extraction reservoir) is highly capable of operating efficiently and natively in the time domain (TD). Unlike conventional signal processing techniques that often require converting the time-domain audio signal into the frequency domain (via DFT) before applying filters, an RC system can process the raw, sequential time-series data directly.

In our novel approach, we use the time-domain capability of the Reservoir Computing technique to extract the Mel Frequency Wavelet Coefficient directly from the raw speech signal. This design allows us to bypass the two major bottlenecks of the conventional methods: we avoid the need to fully calculate both the traditional Fourier/Log-Mel steps and the complex Wavelet Transform steps sequentially, and the entire feature extraction process is integrated into a unified time-domain operation, perfectly aligning with the innate architecture of the RC system. This method effectively replaces the intricate mathematical steps of traditional spectral and wavelet analysis with the dynamic, non-linear processing capability of the reservoir. This streamlining dramatically reduces the method's complexity and increases its efficiency by minimizing computational overhead associated with domain con-

versions. Furthermore, the robust nature of the Reservoir Computing technique is also made use of to improve the performance of the entire system, leading to features that are not only computationally cheaper to obtain but also superior in their ability to capture relevant speech characteristics. This streamlined, time-domain approach represents a significant advance in the quest for low-latency, energy-efficient speech processing systems.

A.2. Methodology

The efficient extraction of features from raw speech data requires careful consideration of the computational environment, particularly in systems where resources are constrained, such as low-power edge devices. The limitations of MFCC—primarily its poor time-frequency resolution—necessitated the development of more robust features, specifically the Mel Frequency Wavelet Coefficient (MFWC). The MFWC is a crucial feature because it successfully combines the perceptually relevant frequency resolution of the Mel scale with the superior time localization capabilities of the Wavelet Transform. However, the conventional extraction of MFWC typically inherits and often compounds the computational complexities of the standard MFCC pipeline.

The standard practice for deriving the Mel Frequency Cepstral Coefficient (MFCC) involves a sequence of operations primarily conducted in the frequency domain, as shown in Figure 2.9. The fundamental motivation for operating in the frequency domain is computational simplification: the complex operation of convolution in the time domain becomes a much simpler multiplication in the frequency domain. Furthermore, this multiplication transforms into an addition in the log-frequency domain, simplifying the final

cepstral analysis. While computationally elegant in a centralized processing environment, this approach necessitates multiple domain conversions (time to frequency, frequency to log-frequency, and back via the Inverse Discrete Cosine Transform (IDCT)), which adds significant computational overhead and latency, particularly when aiming for real-time performance.

A.2.1. Designing Mel Filter-Bank

To overcome the inherent complexity of conventional frequency-domain processing, our methodology focuses on moving the entire MFWC extraction process into the time domain (TD). The challenge lies in recreating the effect of the frequency-domain Mel Filter-Bank and the Wavelet Transform using purely time-domain operations.

Our approach begins by creating a set of time-domain Filter-Bank signals corresponding to each coefficient of the Mel Filter-Bank. In the standard MFCC process, each Mel Coefficient corresponds to a triangular filter centred at a specific frequency in the Mel scale. The shape of this filter, when transformed back into the time domain, dictates the temporal response we need to replicate. We achieve this replication through synthesis.

For each Mel Coefficient, there is an associated set of frequencies and a corresponding set of parameters (such as magnitude and phase, which define the filter shape in the time domain). As an example, the structure for the first Mel Coefficient involves defined frequency and parameter values, as outlined in Table 6.1). We then leverage the principles of Fourier and Wavelet analysis, recognizing that any arbitrary time-domain signal (including a filter's impulse response) can be decomposed into a sum of sine and cosine waves.

Specifically, for the n-th Mel Coefficient:

1. We synthesize a sine wave corresponding to each key frequency and parameter associated with that Mel Coefficient.

2. We synthesize a cosine wave corresponding to each key frequency and parameter associated with that Mel Coefficient.

3. We then superimpose (sum) all the synthesized sine waves to generate the imaginary part of the final time-domain Mel Filter-Bank signal, which embodies the necessary Wavelet Transform properties.

4. Similarly, we superimpose (sum) all the synthesized cosine waves to generate the real part of the final time-domain Mel Filter-Bank signal.

By repeating this process for all Mel Coefficients, we synthesize all the necessary Mel Filter-Bank signals in the time domain (TDMFB). This synthesized TDMFB essentially acts as a set of time-domain basis functions, which, when applied to the audio signal, perform the combined function of Mel filtering and Wavelet decomposition. Figure 6.1 visually represents the resulting time-domain Mel filter bank signal, showcasing its unique temporal structure.

A.2.2. Extraction of MFWC via Convolution

The core principle of our feature extraction is the application of the synthesized TDMFB signals directly to the input audio signal. In the time domain, the operation equivalent to applying a frequency filter is convolution. This is the critical operational shift that allows us to bypass the Discrete Fourier Transform (DFT).

In order to obtain the Time-domain Mel Frequency Wavelet Coefficient (TMFWC), the input audio signal, $s(t)$, is convolved separately with the synthesized real part, $h_{real}^{(m)}(t)$, and the imaginary part, $h_{imag}^{(m)}(t)$, of the time-domain Mel-Wavelet Filter corresponding to each Mel coefficient m:

$$\text{Real}_m(t) = s(t) * h_{real}^{(m)}(t)$$

$$\text{Imag}_m(t) = s(t) * h_{imag}^{(m)}(t)$$

where $*$ denotes the convolution operation. This direct convolution yields a sequence of coefficients ($\text{Real}_m(t)$ and $\text{Imag}_m(t)$) that effectively encode the signal's energy localized in the Mel-frequency band m and time t, possessing the desirable time-frequency characteristics of a Wavelet Transform.

A.2.3. Calculating the Magnitude and Data Reduction

The output of the convolution provides the complex-valued Wavelet coefficients. The magnitude of a Wavelet Transform refers to the absolute value of these complex Wavelet Coefficients, essentially representing the strength or intensity of the signal components at different scales and locations in the time-frequency domain. It indicates how well the signal aligns with the chosen time-domain filter function at a specific time position. A larger magnitude value in a coefficient signifies a stronger presence of the corresponding frequency component within the signal at that specific time window.

The final TMFWC is obtained by calculating the magnitude of the resulting complex coefficients using the standard formula for the magnitude of

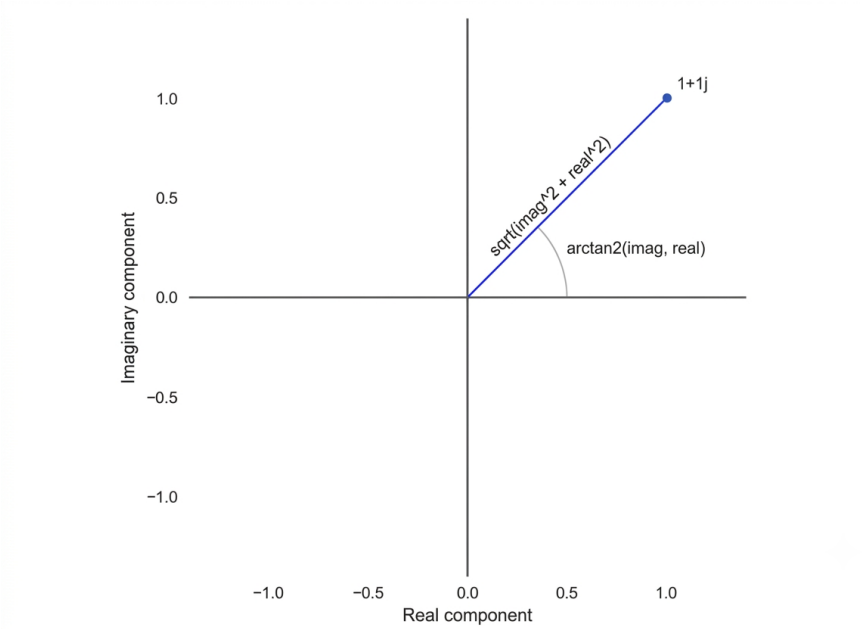


Figure A.2: Finding magnitude

a complex number:

$$\text{TMFWC}_m(t) = \sqrt{(\text{Imag}_m(t))^2 + (\text{Real}_m(t))^2}$$

We have made use of the concept illustrated in Figure A.2 to find the magnitude of the corresponding signal components. The resulting signal, $\text{TMFWC}_m(t)$, is a set of time-domain features, one sequence for each Mel band m.

A crucial characteristic of this time-domain approach is that the number of data points of the resulting TMFWC coefficient sequence is almost the same as the length of the input audio signal. This high dimensionality is typical of time-domain processing but poses a significant challenge for subsequent classification or learning tasks, particularly when feeding the data into a machine learning model like a Reservoir Computing system. High

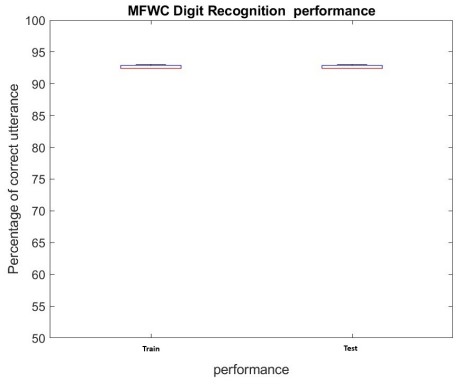
dimensionality increases computational load and can dilute the information concentration.

Therefore, we must employ a data reduction method before feeding the TMFWC data to the reservoir for classification. We have chosen to use the absolute max-pooling technique for this purpose. Max-pooling is a standard downsampling operation used in convolutional neural networks, but here it serves a specific informational purpose: in this case, the largest coefficient among a small interval of data corresponds to the coefficient with the strongest signal information (i.e., the time instance where the corresponding Mel-Wavelet energy was highest). By extracting only the maximum absolute value within a defined pool size (a small time window), we retain the most salient features—the highest energy events—while drastically reducing the overall data size. This focused data reduction method does not negatively affect the critical information content of the signal and prepares the TMFWC sequence for efficient processing by the Reservoir Computing architecture. The final pooled TMFWC signal is then fed directly to the reservoir for the ultimate classification task, demonstrating an end-to-end time-domain feature extraction and learning pipeline.

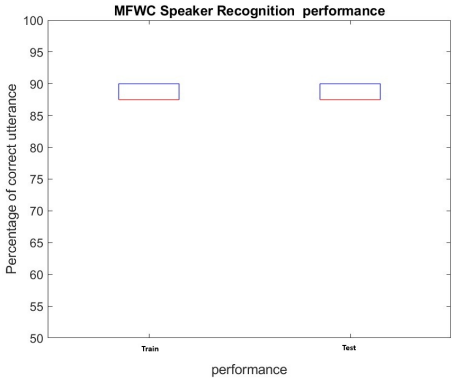
A.3. Experiment and Results

To evaluate the performance of the proposed methods we have used both the Ti-46 data-set and Audio-Mnist dataset for our studies. We have tried to identify the speaker as well as digit using the TMFWC as the feature. We have calculated the percentage of correct utterance across 10 reservoirs and plotted the result. The result is as shown in the figure A.3andA.4. The table

A.1 gives a summary of result.

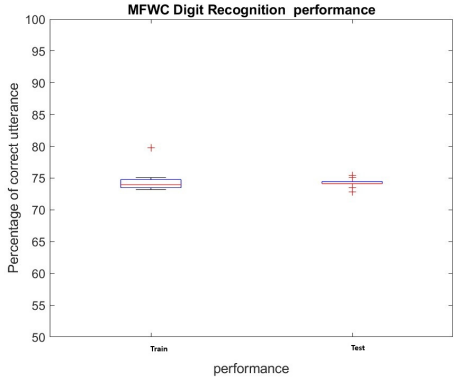


(a) Digit Recognition performance

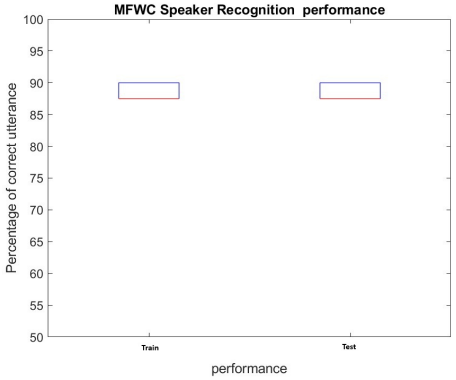


(b) Speaker Recognition performance

Figure A.3: Performance of our system with Ti-46 dataset



(a) Digit Recognition performance



(b) Speaker Recognition performance

Figure A.4: Performance of our system with Audio-Mnist dataset

By demonstrating methods for implementing efficient time-domain Mel Frequency Wavelet Coefficient (MFWC) extraction, we present a framework that improves the efficiency of real-time audio processing. Our approach reduces computational complexity, particularly in performing the Fourier and Wavelet Transforms, by eliminating the need for complex time-frequency

Experiments	Digit Recognition				Speaker Recognition			
	Ti-46		AudioMnist		Ti-46		AudioMnist	
	Train	Test	Train	Test	Train	Test	Train	Test
MFWC	93.03	93.04	74.75	73.13	88.5	87.9	88.40	87.9

Table A.1: Performance analysis of Digit and Speaker Recognition of TI-46 and Audio Mnist data set

conversion. These results are promising, as the system delivers competitive performance while significantly reducing computational overhead.

A.4. Discussion

In this chapter, we discussed the usefulness of the time-domain Mel Frequency Wavelet Coefficient (TMFWC) in the context of high-efficiency speech signal processing. We presented a novel methodology centered on generating the MFWC entirely in the time domain, a strategic shift that minimizes the significant computational complexity associated with existing, conventional extraction methods. By synthesizing time-domain Mel-Wavelet Filter Banks and employing direct convolution, we successfully eliminated the resource-intensive steps of Fourier and inverse transforms and sequential Wavelet decomposition that typically plague traditional frequency-domain feature engineering.

The TMFWC feature displayed significantly more discriminative power than other comparable features, including traditional Mel-scale based features (like standard MFCCs) or pure Wavelet-based Coefficients, when evaluated on critical tasks like speaker and digit recognition. This superior discrimination indicates that the time-domain synthesis successfully captures

both the perceptually relevant frequency information (the 'Mel' aspect) and the crucial localized temporal details (the 'Wavelet' aspect) with high fidelity.

Accompanied by the Reservoir Computing (RC) architecture as the classifier, the entire end-to-end method has substantially improved the overall efficiency of speech signal processing. The RC system is inherently suited for processing the high-dimensional, time-series TMFWC data efficiently, avoiding the need for complex, feed-forward deep network training. The combined simplicity of the feature extraction pipeline and the low-training-cost of the RC classifier positions this framework as a superior solution for real-time and energy-constrained environments. Ultimately, the successful validation of the TMFWC feature confirms its potential to act as a crucial building block for low-power, high-performance edge computing applications in speech and audio analytics.

A.5. Inference and Future Works

The experimental results across both datasets (TI-46 and Audio-MNIST) and both tasks (speaker and digit recognition) clearly demonstrate that the TMFWC feature, combined with the Reservoir Computing classification architecture, yields competitive performance metrics in terms of correct utterance percentage.

The successful validation of the TMFWC feature confirms its potential to act as a crucial building block for low-power, high-performance edge computing applications in speech and audio analytics. The system delivers performance comparable to traditional methods while fundamentally minimizing computational overhead, making it ideally suited for deployment in power-

constrained environments such as voice-activated devices, wearable technology, and low-latency, real-time speech recognition systems.

While the current framework achieves high efficiency by using Reservoir Computing as a classifier, the next logical evolution is to extend this reservoir philosophy to the feature extraction stage itself. Future research will explore the development of a Reservoir-based Mel-Frequency Wavelet Coefficient (R-MFWC) extraction method.

Appendix B: Supplementary Experiments

B.1. Reservoir Computing

To validate the reservoir's temporal memory and non-linear mapping capabilities, the system is benchmarked using the NARMA 10 task. As illustrated in the figure B.1, the tracking accuracy is high, with the predicted output closely following the target signal. The model achieves a Normalized Root

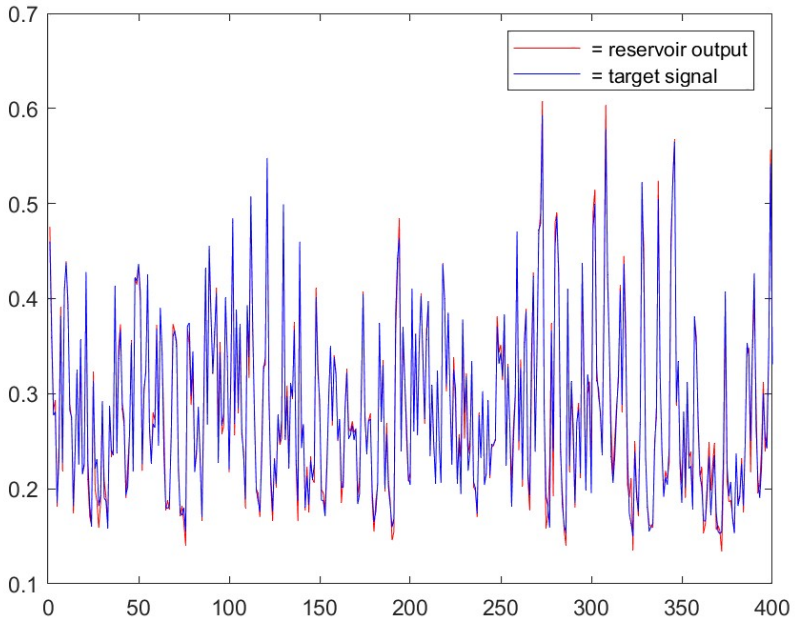


Figure B.1: Narma: target Vs reservoir output

Mean Square Error (NRMSE) of 0.13215 after a washout period of 50 samples. Furthermore, the reservoir demonstrate a Memory Capacity of 14.3887, confirming its ability to retain and process information well beyond the 10th-order dependencies required by the benchmark. These results verify that the reservoir is well-suited for complex tasks.

B.2. Signal Transformation Pipeline: From Time Domain to Cepstrum

The progression from a raw acoustic signal to the cepstrum involves a series of transformations designed to decouple the underlying physical components of sound. Initially, the signal exists in the **Time Domain** as a continuous representation of amplitude over time, $x(t)$. While this captures the raw pressure waves, the spectral characteristics remain latent. By applying the **Fourier Transform**, the signal is projected into the **Frequency Domain**, resulting in a **Spectrum** that identifies the constituent harmonics and their respective magnitudes.

To bridge the gap between physical sound and human perception, the **Log-Magnitude Spectrum** is calculated. This step is mathematically critical because speech is produced through the convolution of a source (vocal cords) and a filter (vocal tract); taking the logarithm transforms this multiplicative relationship into an additive one, expressed as:

$$\log(\text{Source} \cdot \text{Filter}) = \log(\text{Source}) + \log(\text{Filter}) \tag{B.1}$$

Finally, by applying the Inverse Fourier Transform (or the Discrete Co-

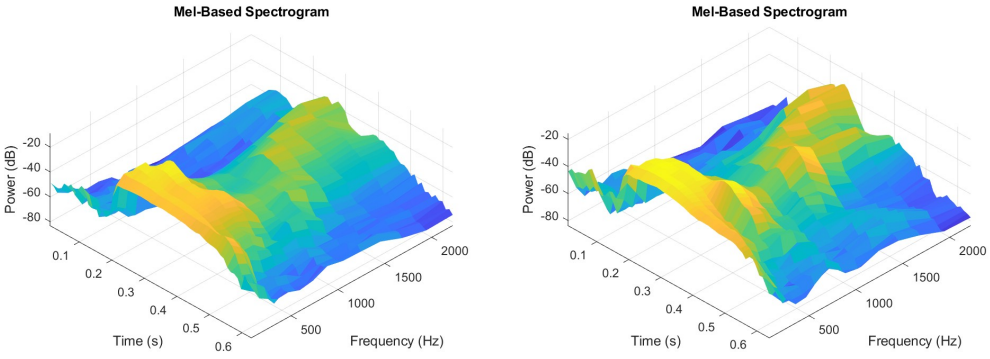
sine Transform for MFCCs) to this log-spectrum, we reach the **Quefrency Domain**, or the **Cepstrum**. This "spectrum of a spectrum" effectively separates the rapidly varying fine structure (source) from the slowly varying spectral envelope (filter), providing a compact, decorrelated representation of the signal's shape.

B.2.1. Temporal Evolution and Mel-Spectrogram Visualization

While the cepstral pipeline provides a snapshot of spectral features, the Mel-based Spectrogram serves as the temporal extension of this process, illustrating how these features evolve over time. It represents a significant departure from standard linear analysis by warping the frequency axis to mimic the non-linear sensitivity of the human auditory system. By projecting high-resolution Short-Time Fourier Transform (STFT) data onto a bank of 14 filters that broaden as frequency increases, the process performs intelligent dimensionality reduction, prioritizing perceptually relevant information while filtering out spectral noise.

This transformation effectively condenses the signal's energy into a compact feature space that preserves the spectral envelope, a step crucial for preventing the reservoir model from becoming overwhelmed by redundant data. The resulting 3D visualizations in Figure B.2 capture the critical temporal dynamics and transient power shifts characteristic of human speech.

As illustrated in Figure B.2, the Mel-based Filter-Bank successfully captures unique acoustic signatures across different datasets. In the Ti-46 sample (a), there is a prominent, sustained energy peak in the lower frequency bands



(a) Ti-46 sample's Spectrogram (b) Audio-Mnist sample's Spectrogram

Figure B.2: Comparative Mel-based Spectrograms showcasing time-varying power distribution.

early in the signal. Conversely, the Audio-MNIST sample (b) exhibits more dispersed, fluctuating energy across the 500 Hz to 1500 Hz range. These differences demonstrate that the 14-band reduction is sensitive enough to distinguish between varying phonetic utterances while maintaining the consistent feature dimensionality.

B.3. An Experiment for Cross-validating Matlab and Reservoir MFCC

To evaluate the fidelity of the reservoir-based MFCC extraction, we conduct a rigorous cross-validation experiment designed to test the functional alignment between our system and existing digital signal processing. The primary objective is to determine if the features synthesized by the reservoir-based generator are algorithmically consistent with those produced by established methods. By decoupling the training and testing feature sources, we could

effectively measure whether the reservoir has captured the universal characteristics of the Mel-frequency scale or has merely developed a proprietary mapping unique to its own internal architecture.

In the initial phase of the experiment, the reservoir-based classifier is trained using MFCC features synthesized entirely by the reservoir's internal approximation of the extraction pipeline. This establishes a baseline where the classifier learned to categorize audio data based on the reservoir's specific interpretation of MFCC. Following this, the performance is validated by testing the classifier on a separate dataset of MFCC features generated via the standard Matlab reference function. This transition from internal training to external testing serves as a critical stress test for the system's robustness. Figure B.3 shows the performance of the experiment.

This cross-validation approach ensures that the reservoir's approximation of the MFCC algorithm maintains high functional compatibility with established signal processing methods. If the classifier achieves high accuracy when processing Matlab-generated features—despite never having seen them during the training phase—it demonstrates that the reservoir has successfully replicated the mathematical transformations (such as the Mel-Filter-Bank integration and Discrete Cosine Transform) required for standard MFCC generation. Ultimately, this confirms that the reservoir-based extraction is a viable, transferable alternative to traditional computational frameworks in real-world applications.

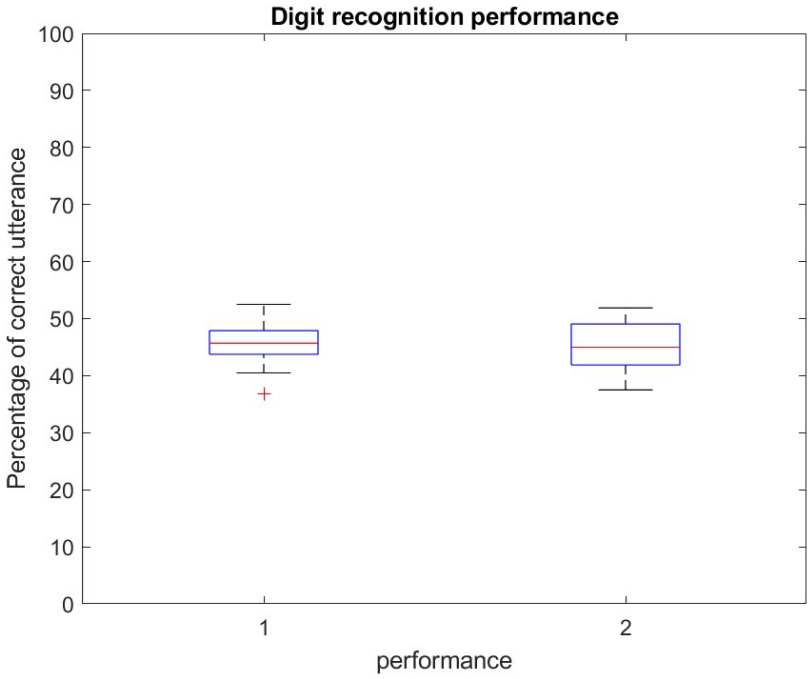


Figure B.3: Cross-validation result

B.4. MFCF Extraction Via Time-Domain Convolution

To evaluate the feasibility and performance of time-domain MFCF extraction, we design an experiment centred on the convolution method. In standard digital signal processing, MFCCs are typically calculated in the frequency domain; however, this experiment transitions those operations into the time domain to establish a performance baseline. While the method detailed in Chapter 5, utilize a reservoir to perform these complex operations, the current experiment intentionally omits the reservoir from the convolution stage. Instead, we manually synthesize a bank of Mel frequency filters in the time domain(as detailed in Chapter 5) and perform a direct convolution

with the raw audio signal.

This setup serves as a controlled environment to isolate the impact of the convolution process itself. By convolving the audio signal with the synthesized filter banks before feeding the resulting features into a classification reservoir, we can observe how effectively time-domain filtering preserves essential acoustic characteristics. This step is critical for validating that the filters—when expressed as temporal impulses rather than frequency bins—can still accurately capture the spectral envelope required for speech and audio recognition tasks. Ultimately, this experiment functions as a comparative

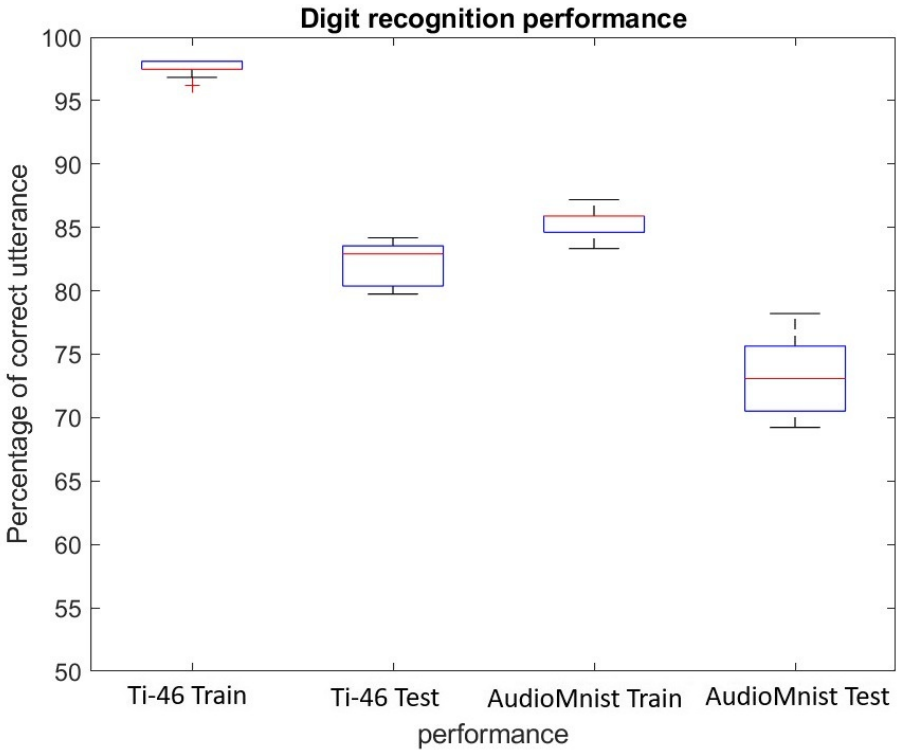


Figure B.4: Digit Recognition performance of Time MFCF

benchmark. By performing the convolution manually and analytically in this stage, we create a point of comparison for the reservoir-based approach.

The results of this study highlights how well reservoir can mimic time domain processing when the reservoir is tasked with the feature extractor, underscoring its role not just as a classifier, but as a sophisticated signal processor.

B.5. MFWC Using Reservoir

While Chapter 9 provides a comprehensive technical breakdown of Mel-Frequency Wavelet Coefficients (MFWC), this section investigates a novel implementation strategy. Specifically, it details an experimental approach where the feature extraction pipeline is replaced by a reservoir.

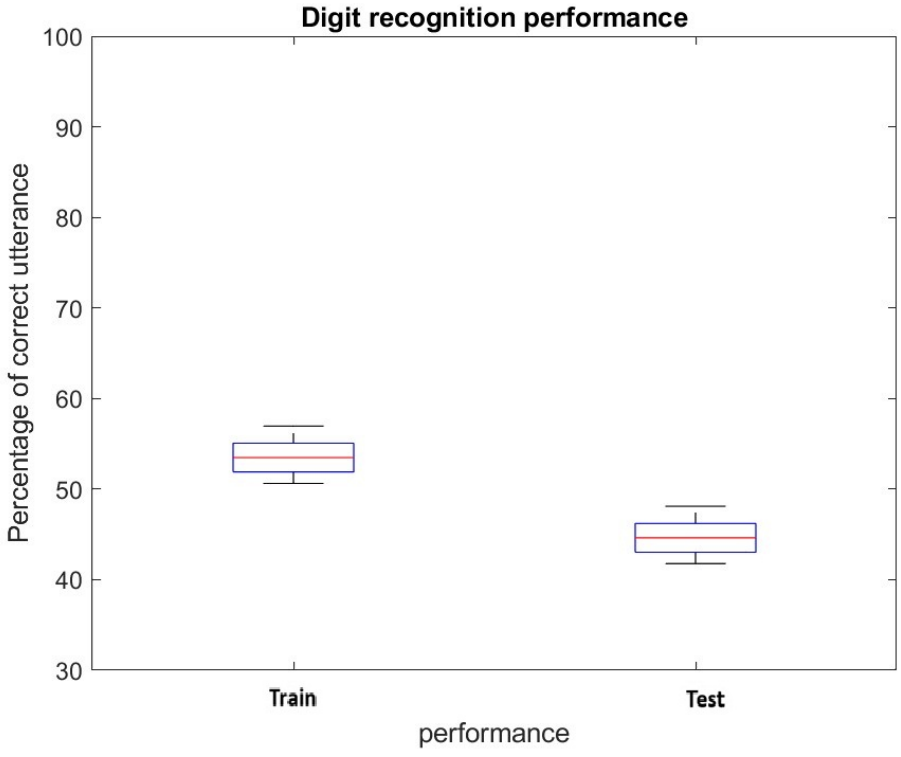


Figure B.5: Digit Recognition performance of MFWC using Reservoir using Ti-46 dataset

The core objective of this implementation is to evaluate the reservoir's

capacity to automate the extraction of MFWC-like features directly from raw data. In this architectural setup, the reservoir is tasked with emulating the behaviour of a complex filterbank. Specifically, the system must be trained to perform separate convolutions for both the sine (imaginary) and cosine (real) components of the filterbank. The final feature set is then derived by calculating the magnitude of these results through the following relation:

$$\text{TMFWC}_m(t) = \sqrt{(\text{Real}_m(t))^2 + (\text{Imag}_m(t))^2}$$

While this approach demonstrates the reservoir's potential as a sophisticated signal processor, it necessitates highly precise fine-tuning of the reservoir's internal dynamics to synchronize the dual-pathway convolutions effectively. The inherent difficulty in achieving this level of hyper-parameter optimization is likely the primary contributor to the observed degradation in performance compared to method described in chapter 9. Consequently, refining the stability and synchronization of these parallel processing paths remains a priority for future iterations of this research, where more advanced optimization strategies will be explored to bridge the performance gap.

B.6. Data Reduction

B.6.1. Wavelet Transform Based Data Reduction

We model a reservoir to do convolution between audio-signal and synthesized time domain equivalent signal of Mel Filter-Bank. While the reservoir successfully emulated the filter bank's behaviour, the primary technical obstacle

emerges in the form of extreme data dimensionality. To address this, we conduct an extensive exploration of data reduction techniques to condense the high-resolution signal into a manageable feature set without losing critical temporal information. Among the various methods tested the application of Wavelet Transforms proves to be the most effective, providing the highest fidelity for the subsequent recognition tasks.

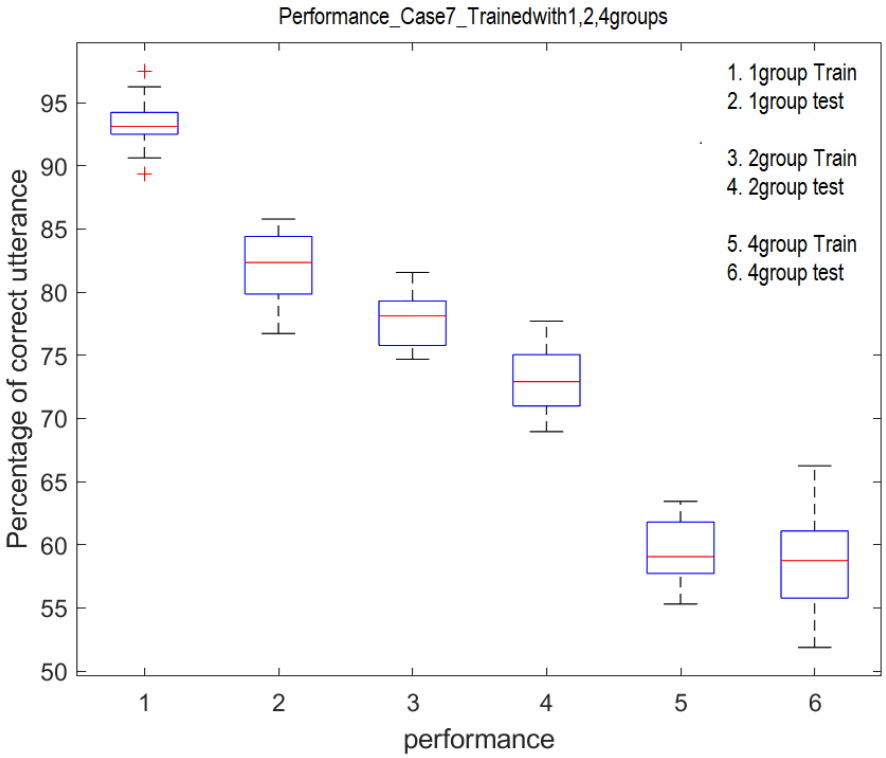


Figure B.6: Time domain MFCC where Wavelet Transform is used for data reduction (1 group =160 audio data used for training. 2 group=2*1 group and so on)

As illustrated in Figure B.6, the Wavelet-based reduction method demonstrates a strong capacity for feature preservation, though its effectiveness was closely tied to the volume of the training set. A notable observation from these results is the inverse relationship between performance and dataset

scale: the system achieved peak accuracy when the training data size is relatively small. However, as the training size increased, a marginal degradation in absolute performance was observed. Interestingly, this decrease in accuracy is accompanied by a narrowing gap between training and testing performance. This convergence suggests that while smaller datasets may allow the reservoir to over-specialize, the larger datasets—despite the slight performance dip—enhance the model's generalization capabilities and provide a more stable estimate of its true predictive power.

B.6.2. Baseline Temporal and Matrix-Based Data Reduction

While Wavelet-based reduction provides high fidelity, its implementation introduces significant computational complexity that can obscure the primary objectives of this study. To maintain focus on the reservoir's processing capabilities, we explored several lower-complexity data reduction strategies such as mean, peak to peak, max-pool/min-pool, absolute max-pool. These methods involve partitioning the input signal into discrete temporal windows, where the number of windows is equivalent to the number of MFCC output samples that we get when using the Matlab MFCC function in order to fit into the experiment framework with the second reservoir later. Out of each window we pick one data-point using one of the above said method.

We have also tried matrix method for dimension reduction. Here we first created a matrix of size equivalent to the window size. We have tried different functions to generate this matrix such as atanh, GELU, hann,hamm,sigmoid and so on. We then multiply our input signal with this window in a sliding

window concept. Thus we get one data-point from a window-length of the signal.

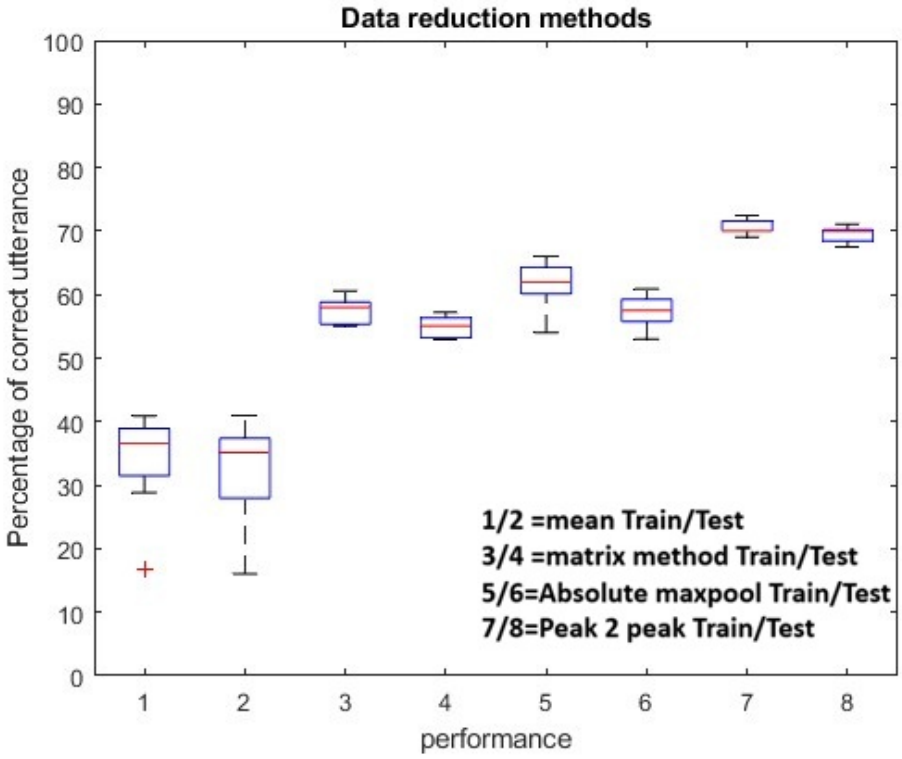


Figure B.7: Data reduction methods performance

We utilise the Audio-MNIST dataset to conduct speaker recognition experiments, the results of which serve to evaluate the performance of our dimensionality reduction methods in Figure B.7

The performance of each dimensionality reduction technique is rigorously assessed based on classification accuracy and computational overhead. Ultimately, the Peak-to-Peak method is adopted for the final end-to-end pipeline, as it provided the best trade-off between feature compression and the retention of distinctive acoustic signatures required for the task.

References

[1] Benjamin Schrauwen, David Verstraeten, and Jan Van Campenhout. An overview of reservoir computing: theory, applications and implementations. *Proceedings of the 15th European Symposium on Artificial Neural Networks*, pages 471–482, 2007.

[2] L. Appeltant, G. Vander Sande, J. Danckaert, and I. Fischer. Constructing optimized binary masks for reservoir computing with delay systems. *Scientific Reports*, 2(1):285, 2012.

[3] Mantas Lukoševičius. A practical guide to applying echo state networks. In *Lecture Notes in Computer Science*, Lecture notes in computer science, pages 659–686. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[4] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

[5] Jiang Wang, Ammar Belatreche, Liam Maguire, and T. M. McGinnity. An online supervised learning method for spiking neural networks with adaptive structure. *Neurocomputing*, 144:526–536, 2015.

[6] Lennert Appeltant, Miguel C Soriano, Guy Vander Sande, Jan Danckaert, Serge Massar, Joni Dambre, Benjamin Schrauwen, Claudio R Mirasso, and Ingo Fischer. Information processing using a single dynamical node as complex system. *Nature Communications*, 2(1):468, 2011.

[7] Tushar Sawant and S. D. Bhosale. Speech recognition using artificial neural network: A review. *International Journal of Computer Science and Information Technologies*, 1(5):430–434, 2010.

[8] Sachin Magre and G. S. Sable. A review on feature extraction and noise reduction technique. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2(2):227–231, 2014.

[9] S Russell and V Yoon. Applications of wavelet data reduction in a recommender system. *Expert Syst. Appl.*, 34(4):2316–2325, May 2008.

[10] M. Mitchell Waldrop. The chips are down for moore's law. *Nature*, 530(7589):144–147, 2016.

[11] John Backus. Can programming be liberated from the von neumann style? a functional style and its algebra of programs. *Communications of the ACM*, 21(8):613–641, 1978.

[12] Catherine D. Schuman, Shruti R. Kulkarni, Maryam Parsa, J. Parker Mitchell, Bill Kay, and Adam Prügel-Bennett. Opportunities for neuromorphic computing algorithms and applications. *Nature Computational Science*, 2(1):10–19, 2022.

[13] Carver Mead. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–1636, 1990.

[14] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100–123, 2019.

[15] Herbert Jaeger and Harald Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[16] Giacomo Indiveri, Bernabé Linares-Barranco, Tara Julia Hamilton, André Van Schaik, Ralph Etienne-Cummings, Tobi Delbruck, Shih-Chii Liu, Piotr Dudek, Philipp Häfliger, Sylvie Renaud, et al. Neuromorphic silicon neuron circuits. *Frontiers in neuroscience*, 5:73, 2011.

[17] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018.

[18] Susan Stepney, Samuel L Braunstein, John A Clark, Andy Tyrrell, Andrew Adamatzky, Robert E Smith, Tim Addis, Colin Johnson, Jon Timmis, Peter Welch, et al. Journeys in non-classical computation i:

A grand challenge for computing research. *International Journal of Parallel, Emergent and Distributed Systems*, 20(1):5–19, 2005.

[19] Andrew Adamatzky. *Unconventional computing*. Luniver Press, 2010.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017.

[21] Deepanway Ghosal, Navonil Majumder, Ambuj Mehrish, and Soujanya Poria. Text-to-audio generation using instruction-tuned LLM and latent diffusion model. *arXiv [eess.AS]*, April 2023.

[22] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *International Conference on Machine Learning (ICML)*, pages 28492–28518, 2023.

[23] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, and Neil Zeghidour. AudioLM: A language modeling approach to audio generation. *IEEE ACM Trans. Audio Speech Lang. Process.*, 31:2523–2533, 2023.

[24] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:12449–12460, 2020.

[25] David Verstraeten, Benjamin Schrauwen, Michiel D'Haene, and Dirk Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391–403, 2007.

[26] Fabian Triefenbach, Azarakhsh Jalalvand, Benjamin Schrauwen, and Jean-Pierre Martens. Phoneme recognition with large hierarchical reservoirs. *Advances in Neural Information Processing Systems*, 23, 2010.

[27] David Verstraeten, Benjamin Schrauwen, Dirk Stroobandt, and Jan Van Campenhout. Isolated word recognition with the liquid state machine: a case study. *Information Processing Letters*, 95(6):521–528, 2005.

[28] Benjamin Schrauwen, David Verstraeten, and Jan Van Campenhout. An overview of reservoir computing: theory, applications and implementations. *Proceedings of the 15th European Symposium on Artificial Neural Networks*, pages 471–482, 2007.

[29] Matthew Dale, Simon O'Keefe, Angelika Sebald, Susan Stepney, and Martin Albrecht Trefzer. Reservoir computing quality : connectivity and topology. *Nat. Comput.*, December 2020.

[30] Susan Stepney. Physical reservoir computing: a tutorial. *Nat. Comput.*, 23(4):665–685, December 2024.

[31] Matthew Dale, Julian F Miller, Susan Stepney, and Martin A Trefzer. A substrate-independent framework to characterise reservoir computers. *arXiv [cs.ET]*, October 2018.

[32] Matthew Dale, Julian F Miller, and Susan Stepney. Reservoir computing as a model for in-materio computing. In *Emergence, Complexity and Computation*, pages 533–571. Springer International Publishing, Cham, 2017.

[33] Antonio C Torrezan, John Paul Strachan, Gilberto Medeiros-Ribeiro, and R Stanley Williams. Sub-nanosecond switching of a tantalum oxide memristor. *Nanotechnology*, 22(48):485203, 2011.

[34] Enrico Picco, Alessandro Lupo, and Serge Massar. Deep photonic reservoir computer for speech recognition. *arXiv [cs.NE]*, December 2023.

[35] Kristof Vandoorne, Pauline Mechet, Thomas Van Vaerenbergh, Martin Fiers, Geert Morthier, David Verstraeten, Benjamin Schrauwen, Joni Dambre, and Peter Bienstman. Experimental demonstration of reservoir computing on a silicon photonics chip. *Nature Communications*, 5(1):3541, 2014.

[36] Sam Lilak, Walt Woods, Kelsey Scharnhorst, Christopher Dunham, Christof Teuscher, Adam Z Stieg, and James K Gimzewski. Spoken digit classification by in-materio reservoir computing with neuromorphic atomic switch networks. *Front. Nanotechnol.*, 3, May 2021.

[37] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[38] Sepp Hochreiter. Untersuchungen zu dynamischen neuronalen netzen. *Diploma thesis, Institut für Informatik, Technische Universität München*, 1991.

[39] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks. *GMD-German National Research Institute for Computer Science*, (148), 2001.

[40] Wolfgang Maass, Thomas Natschläger, and Henry Markram. Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

[41] Tomohiro Paine, Takashi Aoki, and Yasuaki Arakawa. Reservoir computing with a single delay-coupled node. *Japanese Journal of Applied Physics*, 59(6):060901, 2020.

[42] John B Butcher, David Verstraeten, Benjamin Schrauwen, Craig R Day, and Peter W Haycock. Extending reservoir computing with random static projections: a hybrid between extreme learning and rc. *ESANN*, pages 303–308, 2010.

[43] Mantas Lukosevicius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

[44] Enrico Picco, Alessandro Lupo, and Serge Massar. Deep photonic reservoir computer for speech recognition. *IEEE Trans. Neural Netw. Learn. Syst.*, 36(4):7606–7614, April 2025.

[45] M. A. Anusuya and S. K. Katti. Speech recognition by machine: A review. In *2009 International Conference on Signal and Image Processing*, pages 148–151. IEEE, 2009.

[46] Thomas Grill and Jan Schlüter. Music boundary detection using neural networks on spectrograms and self-similarity lag matrices. *Proceedings of the 23rd European Signal Processing Conference (EUSIPCO)*, pages 1296–1300, 2015.

[47] Beth Logan. Mel frequency cepstral coefficients for music modeling. *In Proceedings of the 1st International Symposium on Music Information Retrieval (ISMIR)*, 2000.

[48] Keyan He, Dihu Chen, and Tao Su. A configurable accelerator for keyword spotting based on small-footprint temporal efficient neural network. *Electronics (Basel)*, 11(16):2571, August 2022.

[49] Kabilan Palanisamy and Murugappan Selvam. An efficient architecture for mfcc feature extraction using parallel processing. In *2020 4th International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1405–1411. IEEE, 2020.

[50] Weipeng Tong, Guodong Li, Jianying Geng, and Junwei Han. Multitask reservoir computing for time series data. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[51] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks. (148), 2001.

[52] David Verstraeten, Benjamin Schrauwen, Michiel D'Haene, and Dirk Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391–403, 2007.

[53] Felix Grezes. Reservoir computing: A new paradigm for neural networks. *arXiv [cs.LG]*, April 2025.

[54] Stuart Russell and Peter Norvig. *Artificial intelligence: A modern approach, global edition*. Pearson Education, London, England, 4 edition, May 2021.

[55] Erik Brynjolfsson and Andrew McAfee. The second machine age: Work, progress, and prosperity in a time of brilliant technologies. *pp*, 306, 2014.

[56] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[57] Melanie Mitchell. *Artificial intelligence: A guide for thinking humans*. Pelican Books. Pelican, London, England, October 2019.

[58] Ziad Obermeyer, Brian Powers, Christine Vogeli, and Sendhil Mullainathan. Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464):447–453, October 2019.

[59] Foster Provost and Tom Fawcett. *Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking*. O'Reilly Media, 2013.

[60] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, 2nd edition, 2009.

[61] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction.* The MIT Press, 2nd edition, 2018.

[62] Christopher M. Bishop. *Pattern Recognition and Machine Learning.* Springer, New York, 2006.

[63] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[64] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105, 2012.

[65] S Hochreiter and J Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[66] Tian Gan, Susan Stepney, Martin A. Trefzer. Combining multiple inputs to a delay-line reservoir computer: Control of a forced van der pol oscillator system. 2023.

[67] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100–123, 2019.

[68] Herbert Jaeger. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, 148(34):13, 2010.

[69] Peter F. Dominey. Complex sensory-motor sequence learning based on recurrent state representation and reinforcement learning. *Biological Cybernetics*, 93(2):137, 2004.

[70] J. J. Steil. Backpropagation-decorrelation: online recurrent learning with o(n) complexity. *Proceedings of the 2004 IEEE International Joint Conference on Neural Networks*, 2:843–848, 2004.

[71] Mantas Lukosevicius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

[72] Matthew Dale, Jack Daniel Dewhirst, Simon Edward Marius O'Keefe, Angelika Anne-Marie Sebald, Susan Stepney, and Martin Albrecht Trefzer. The role of structure and complexity on reservoir computing quality. In *UCNC 2019, Tokyo, Japan, June 2019*, page 13, JPN, 2019.

[73] Kohei Nakajima. Physical reservoir computing—an introductory perspective. *Japanese Journal of Applied Physics*, 2020.

[74] T L Carroll. Optimizing memory in reservoir computers. *Chaos*, 32(2), February 2022.

[75] Chester Wringe. *Scaling up ESNs with Heterogeneous Reservoirs*. PhD thesis, University of York, York, 2024.

[76] Mantas Lukosevicius. A practical guide to applying echo state networks. *Neural Networks: Tricks of the Trade*, 7700:659–686, 2012.

[77] Matthew Nicholas Dale, Susan Stepney, Julian Francis Miller, and Martin Albrecht Trefzer. Reservoir computing in materio : An evaluation of configuration through evolution. In *IEEE Symposium Series on Computational Intelligence (SSCI), 2016*. IEEE, 2016.

[78] Gouhei Tanaka, Toshiyuki Yamane, Jean Benoit Héroux, Ryosho Nakane, Naoki Kanazawa, Seiji Takeda, Hidetoshi Numata, Daiju Nakano, and Akira Hirose. Recent advances in physical reservoir computing: A review. *Neural Networks*, 115:100–123, 2019.

[79] Ma Qianli, Lidan Shen, and Garrison W. Cottrell. Deep-echo state network and its application to time series prediction. *Neural Networks*, 83:89–99, 2016.

[80] Michiel Hermans and Benjamin Schrauwen. Recurrent kernel machines: Computing with infinite echo state networks. *Neural Computation*, 24(1):104–133, 2012.

[81] Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *NeuralNetworks*, 1997 Elseier Science Ltd.(9):1659–1671, 1997.

[82] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, November 2019.

[83] Yi Wan, Shaoping Wang, and Di Liu. Fault diagnosis using liquid state machine with spiking-timing-dependent plasticity learning rule. *Expert Syst. Appl.*, 271(126736):126736, May 2025.

[84] Wolfgang Maass, Prashant Joshi, and Eduardo D. Sontag. Computational aspects of feedback in neural circuits. *PLoS Computational Biology*, 3(1):e165, 2007.

[85] Wolfgang Maass and Henry Markram. On the computational power of circuits of spiking neurons. *Journal of Computer and System Sciences*, 69(4):593–616, 2004.

[86] Tian Gan, Susan Stepney, and Martin A Trefzer. Combining multiple inputs to a delay-line reservoir computer: Control of a forced van der pol oscillator system. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, June 2023.

[87] Miguel C Soriano, Silvia Ortín, Lars Keuninckx, Lennert Appeltant, Jan Danckaert, Luis Pesquera, and Guy van der Sande. Delay-based reservoir computing: noise effects in a combined analog and digital implementation. *IEEE Trans Neural Netw Learn Syst*, 26(2):388–393, February 2015.

[88] C. Merkel, N. Hasan, S. Datta, and N. Shah. Delay-based reservoir computing with a silicon photonic platform. *Journal of Lightwave Technology*, 35(17):3691–3698, 2017.

[89] John G. Proakis and Dimitris G. Manolakis. *Digital Signal Processing: Principles, Algorithms and Applications*. Pearson, 4th edition, 2007.

[90] Alan V. Oppenheim and Ronald W. Schafer. *Discrete-Time Signal Processing*. Prentice Hall, 2nd edition, 1999.

[91] J. N. Holmes. *Speech Synthesis and Recognition*. Van Nostrand Reinhold (UK), 1988.

[92] Richard F. Lyon. Human and machine hearing, 2017.

[93] Parmar Rinku, Darshak Shah, and Apurva Shah. Study of formants frequency and formants bandwidth for gujarati vowels. *International Journal of Computer Applications*, 122(4):1–4, 2015.

[94] Gaurav Sharma, Karthikeyan Umapathy, and Sridhar Krishnan. Trends in audio signal feature extraction methods. *Applied Acoustics*, 158:107020, 2020.

[95] S. Ajibola Alim, Ummi Kalsom Khairuddin, Sheroz Khan, Jafreezal Mohd Ali, Noorhuzaimi Yahya, and Radiah Mohd Hanifa. A review on audio feature extraction methods for speech recognition: Challenges and opportunities. *International Journal of Engineering & Technology*, 7(3.21):1–6, 2018.

[96] Dariusz Niewiadomy and Adam Pelikant. Implementation of mfcc vector generation in classification systems. *Proceedings of the International Multiconference on Computer Science and Information Technology*, 3:135–139, 2008.

[97] K. Sreenivasa Rao, V. Ramu Reddy, and Shyamal Maity. Language identification using spectral and prosodic features. *Springer Briefs in Speech Technology*, 2017.

[98] Sirko Molau, Michael Pitz, Ralf Schluter, and Hermann Ney. Computing mel-frequency cepstral coefficients on the power spectrum. *Proceedings of the 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1:73–76, 2001.

[99] Vaibhav Tiwari. Mfcc and its applications in speaker recognition. *International Journal on Emerging Technologies*, 1(1):19–22, 2010.

[100] Navneet Singh and R. A. Khan. Speech recognition with hidden markov model: A review. *International Journal of Advanced Research in Computer Science and Software Engineering*, 6(3):131–137, 2016.

[101] S. G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989.

[102] Li Deng and Dong Yu. Deep learning for signal and information processing. *Foundations and Trends in Signal Processing*, 7(1–2):1–199, 2013.

[103] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[104] Alessio Micheli Silvestri, Claudio Gallicchio. Local lyapunov exponents of deep echo state networks. *Neurocomputing*, 298:34–45, July 2018.

[105] Claudio Gallicchio and Alessio Micheli. Deep reservoir computing: A critical empirical study. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, pages 273–278, 2017.

[106] Rishona Daniels* Duna Wattad Ronny Ronen David Saad Shahar Kvatinsky*. On the role of preprocessing and memristor dynamics in reservoir computing for image classification. `https://arxiv.org/html/2604.21602v1`, 2026.

[107] Claudio Gallicchio and Alessio Micheli. Deep reservoir computing. In Kohei Nakajima and Ingo Fischer, editors, *Reservoir Computing: Theory, Physical Implementations, and Applications*, pages 77–95. Springer Singapore, Singapore, 2021.

[108] Gregor Holzmann and Helge Hauser. Echo state networks with filter neurons and a delay & sum readout. *Neural Networks*, 23(2):244–256, 2010.

[109] David Verstraeten, Benjamin Schrauwen, Dirk Stroobandt, and Jan Van Campenhout. Isolated word recognition with the liquid state machine: a case study. *Information Processing Letters*, 95(6):521–528, 2005.

[110] David Verstraeten, Benjamin Schrauwen, Michiel D'Haene, and Dirk Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391–403, 2010.

[111] Fernando Abreu Araujo, Mikael Riou, Jacob Torrejon, Sumito Tsunegi, Kay Yakushiji, Akio Fukushima, Hitoshi Kubota, Shinji Yuasa, Damien Querlioz, and Julie Grollier. Role of non-linear data processing on speech recognition task in the framework of reservoir computing. *Scientific Reports*, 10(1):328, 2020.

[112] Sagar Lilak, Dmitri Markovic, and Alice Mizrahi. Reservoir computing with superconductor electronics. *Nature Communications*, 12(1):3268, 2021.

[113] Muhammad Usman Ghani and George K. Atia. Reservoir computing for speech recognition. *Proceedings of the 2010 IEEE International Symposium on Circuits and Systems*, pages 4177–4180, 2010.

[114] Karel Veselỳ, Arnab Ghoshal, Lukáš Burget, and Daniel Povey. Sequence-discriminative training of deep neural networks. *Proc. Interspeech*, 2013:2345–2349, 2013.

[115] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.

[116] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv [cs.SD]*, September 2016.

[117] Lucas Deckers, Ing Jyh Tsang, Werner Van Leekwijck, and Steven Latré. Extended liquid state machines for speech recognition. *Front. Neurosci.*, 16:1023470, October 2022.

[118] Fabian Triefenbach, Azarakhsh Jalalvand, Benjamin Schrauwen, and Jean-Pierre Martens. Phoneme recognition with large hierarchical reservoirs. *Advances in Neural Information Processing Systems*, 23:2307–2315, 2010.

[119] François Duport, Ananthakrishna Bhaduri, Marc Haelterman, and Serge Massar. All-optical reservoir computing based on a single nonlinear node with delayed feedback. *Optics Express*, 20(27):28583–28595, 2012.

[120] B. J. Giron Castro, Christophe Peucheret, Darko Zibar, and Francesco Da Ros. Multi-task photonic reservoir computing: wavelength division multiplexing for parallel computing with a silicon microring resonator. *Advanced Optical Technologies*, 13(2):115–124, 2024.

[121] Graham E Rowlands, Minh-Hai Nguyen, Guilhem J Ribeill, Andrew P Wagner, Luke C G Govia, Wendson A S Barbosa, Daniel J Gauthier, and Thomas A Ohki. Reservoir computing with superconducting electronics. *arXiv [cond-mat.supr-con]*, March 2021.

[122] Li-Yue Zhang Li, Tong Yang. Spatiotemporal multiplexed photonic reservoir computing. *DOI: 10.29026/oea.2025.250159*, December 2025.

[123] Xulei Wu, Bing Dang, Jing Lin, and Yuch-Chi Yang. Spatiotemporal audio feature extraction with dynamic memristor-based time-surface neurons. *Science Advances*, 10(14):eadi3603, 2024.

[124] P. Chaudhary, S. Kothandaraman, H. Deka, K. Vaddi, and K. Seshadri. Hybrid machine learning approach for speech signal processing using reservoir computing. In *2024 International Conference on Communication and Signal Processing (ICCSP)*, pages 30–34. IEEE, 2024.

[125] D. Pierangeli, L. Di Palma, G. Scrimieri, L. Antonelli, G. Scagliusi, L. Scola, C. Betti, V. Gatti, A. Delmonte, L. Carletti, et al. Deep photonic reservoir computer for speech recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[126] Steven B. Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980.

[127] Alex Graves and N Jaitly. Towards end-to-end speech recognition with recurrent neural networks. *ICML*, 32(2):1764–1772, June 2014.

[128] Tara N Sainath, Ron J Weiss, Andrew Senior, Kevin W Wilson, and Oriol Vinyals. Learning the speech front-end with raw waveform CLDNNs. In *Interspeech 2015*, ISCA, September 2015. ISCA.

[129] Mirco Ravanelli and Yoshua Bengio. Multi-task learning for speaker verification and speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4844–4848. IEEE, 2018.

[130] Dario Amodei, Santosh Ananthanarayanan, Rishita Anubhai, Jing Bai, Caglar Gulcehre Chen, Jong Wook Chen, Mike De Langhe, Li Duan, Zhang Fu, Vicki Ghassemi, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. *International Conference on Machine Learning (ICML)*, 2016.

[131] Matthew H Tong, Adam D Bickett, Eric M Christiansen, and Garrison W Cottrell. Learning grammatical structure with echo state networks. *Neural Netw.*, 20(3):424–432, April 2007.

[132] Sören Becker, Johanna Vielhaben, Marcel Ackermann, Klaus-Robert Müller, Sebastian Lapuschkin, and Wojciech Samek. AudioMNIST: Exploring explainable artificial intelligence for audio analysis on a simple benchmark. *arXiv [cs.SD]*, July 2018.

[133] Gopalakrishnan Srinivasan, Priyadarshini Panda, and Kaushik Roy. SpiLinC: Spiking liquid-ensemble computing for unsupervised speech and image recognition. *Front. Neurosci.*, 12:524, August 2018.

[134] Guillaume Dion, Salim Mejaouri, and Julien Sylvestre. Reservoir computing with a single delay-coupled non-linear mechanical oscillator. *J. Appl. Phys.*, 124(15):152132, October 2018.

[135] C Sridhar and Aniruddha Kanhe. Performance comparison of various neural networks for speech recognition. *J. Phys.: Conf. Ser.*, 2466(1):012008, March 2023.

[136] Rinku Sebastian, Simon O'Keefe, and Martin Trefzer. Bridging biological hearing and neuromorphic computing: End-to-end time-domain audio signal processing with reservoir computing. *arXiv [cs.SD]*, March 2026.

[137] Rinku Sebastian, Simon O' Keefe, and Martin A Trefzer. Enhancing MFCC feature extraction through reservoir computing. In *Lecture Notes in Computer Science*, Lecture Notes in Computer Science, pages 294–306. Springer Nature Switzerland, Cham, 2026.

[138] Gundeep Singh, Sahil Sharma, Vijay Kumar, Manjit Kaur, Mohammed Baz, and Mehedi Masud. Spoken language identification using deep learning. *Computational Intelligence and Neuroscience*, 2021(1), January 2021.

[139] Anushka Sandesara, Shilpi Parikh, Pratyay Sapovadiya, Mrugendrasinh Rahevar. A comparative study on speech emotion recognition, November 2020.

[140] Tiancheng Deng. Effect of the number of hidden layer neurons on the accuracy of the back propagation neural network. *Highlights in Science, Engineering and Technology*, 74:462–468, 2023.

[141] H. Luan. Optimised reservoir computing for temporal signal classification in lidar. Master's thesis, University of Kent, 2024.

[142] B. J. Giron Castro, C. Peucheret, D. Zibar, and F. Da Ros. Multitask photonic reservoir computing: wavelength division multiplexing for parallel computing with a silicon microring resonator. *Advanced Optical Technologies*, 13, 2024.

[143] V. Nikolić, M. Echlin, B. Aguilar, and I. Shmulevich. Computational capabilities of a multicellular reservoir computing system. *PLOS ONE*, 18(3):e0282122, 2023.

[144] N. Soures and D. Kudithipudi. Deep liquid state machines with neural plasticity for video activity recognition. *Frontiers in Neuroscience*, 13:686, 2019.

[145] Zain Iqbal and Lorenzo Valerio. EARL: Energy-aware optimization of liquid state machines for pervasive AI. *arXiv [cs.LG]*, January 2026.

[146] Rinku Sebastian, Simon O'Keefe, and Martin Trefzer. Audio signal processing using time domain mel-frequency wavelet coefficient. *arXiv [cs.SD]*, October 2025.