Optimised reservoir computing for temporal signal classification in LiDAR

By Tianxiang Luan

MSc Supervised by Dr. Chao Wang

School of Engineering

January 2024

Abstract

Reservoir Computing has emerged as a potent machine learning tool, distinguished by its efficacy in handling complex sequential signals with minimal training complexity. This thesis delves into the realm of applying RC to train and classify distance information gleaned from FMCW LiDAR. Validation is carried out through simulation and experiments. Subsequently, the parameters of reservoir computing are optimized using genetic algorithm.

FMCW LiDAR, an optical sensing technology, is adept at gauging target distances by capturing changes in the time-domain waveform of the IF signal. Noteworthy is the spectral broadening issue arising from nonlinearities in the light source's wavelength-scanning process, leading to degraded ranging resolution. This research addressed this challenge by adapting RC-based temporal signal classification, completely avoiding computing-heavy Fourier analysis in existing systems. The approach has been validated via successful simulation experiments. These experiments showcased RC's proficiency in processing time-domain waveform signals, even when grappling with similar low-frequency waveforms.

To validate RC's real-world applicability, experimental signals were subjected to various scenarios, including sequential, jumbled, and single waveform distance signals of different lengths. Remarkably, RC demonstrated impeccable performance with 100% accuracy, attesting to its robustness across diverse signal configurations.

To quantify the quality of results, the study introduced RMSE, a metric measuring the discrepancy between test and standard results. The close alignment between test results and standard RMSE (ranging between 0.03 and 0.04) underscored RC's strong performance.

Despite RC's prowess, the importance of parameter tuning cannot be overstated. The study employed a genetic algorithm to optimize six crucial parameters in the RC model: spectral radius, lambda (regularization rate), res_units (number of nodes in RC), activation function, scaling, and connectivity. The choice of RMSE as the objective function for optimization aimed to fine-tune RC for optimal performance.

After 80 generations of iterative optimization, the observed decrease in RMSE signifies the genetic algorithm's success in identifying parameter combinations that enhance RC's performance. However, it's noteworthy that the overall degree of RMSE decrease is moderate. This suggests that RC inherently possesses a degree of robustness, capable of delivering acceptable results without exhaustive parameter optimization.

In conclusion, the synergy of RC's robust performance, combined with thoughtful parameter optimization through a genetic algorithm, positions it as a reliable and adaptable tool for processing and classifying complex sequential signals in applications such as FMCW LiDAR.

Acknowledgement

Dear Dr. Chao Wang, I am writing to express my sincere gratitude for your dedicated guidance and selfless support throughout my doctoral research. I have greatly benefited from your profound knowledge, passion for academia, and patient guidance. Under your mentorship, I not only acquired professional expertise but also developed independent thinking and problem-solving skills. Your inspiration has allowed me to delve into my research more comprehensively and deeply. I sincerely thank you for your attentive teaching and guidance!

To Dr. Yuanli Yue and Dr. Shouju Liu, I extend my heartfelt gratitude. Thanks to your collaborative efforts, our research progressed smoothly. Your assistance in steering me in the right direction when I faced confusion and your support in both my academic and personal life have been invaluable. Your cooperation has been indispensable at every step of my research.

I would also like to express my gratitude to my family and friends for their selfless care and unwavering encouragement throughout my research journey. Your support has been my strongest motivation to forge ahead.

Finally, I want to thank all those who have contributed to and supported me during my MSc research, both academically and personally. I deeply appreciate the friendship and support from each of you, enabling me to overcome challenges and move towards success.

TABLE OF CONTENTS

Presentation of pictures

List of Acronyms

Acronyms	Full Form
AI	Artificial Intelligence
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
CW	Continuous Wave
DFB	Distributed Feedback
ESN	Echo State Network
FFT	Fast Fourier Transform
FMCW	Frequency Modulated Continuous Wave
GAs	Genetic Algorithms
IF	Intermediate Frequency
LiDAR	Light Detection and Ranging
LSM	Liquid State Machine
LSTM	Long Short-Term Memory
ML	Master Laser
MLP	Multilayer Perceptron
MZI	Mach–Zehnder Interferometer
MZM	Mach–Zehnder Modulator
NMT	Neural Machine Translation
PD	Photodetector

RC	Reservoir Computing
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SL	Slave Laser
SNR	Signal-to-noise Ratio
SVM	Support Vector Machines

Chapter 1

Introduction

1. 1 Implications and challenges of the study

Reservoir Computing is an emerging machine learning algorithm that has garnered significant attention from researchers since its inception. It has demonstrated impressive performance in various domains, including time series prediction, speech recognition, natural language processing, image processing, control systems, biomedical applications, financial forecasting, and signal quality enhancement. RC exhibits wide applicability in the classification of temporal signals and offers several advantages over traditional RNNs. Notably, RC can efficiently capture long-term dependencies in temporal signals, employs a fixed reservoir that eliminates the need for sequential training, and allows parallel processing of input signals, thus enhancing computational efficiency. Furthermore, RC is adaptable to diverse types of temporal data, such as speech, biological signals, financial time series, showcasing its versatility. In addition RC's outstanding ability to model nonlinear dynamics makes it particularly valuable for problems involving complex time features.

Despite its promising potential, there remain several challenges in the current research and application of RC. One significant limitation is the relative lack of interpretability in RC models, which may restrict its usage in fields that require explanatory insights. RC also exhibits a high sensitivity to data distribution and characteristics, leading to variable performance across different datasets and application scenarios. Some applications may necessitate substantial labeled data for training, which might be challenging to obtain in certain domains. Expanding RC models to handle larger and more complex temporal signal datasets can pose significant challenges. Additionally, RC involves numerous hyperparameters, such as reservoir size and connection weights, and selecting appropriate values for these parameters can be challenging, with suboptimal choices leading to reduced performance.

When faced with challenges like parameter tuning, optimization algorithms, such as genetic algorithms, offer promising solutions. Global optimization using genetic algorithms can expedite the search for optimal hyperparameter combinations in reservoir computing.

In this study, experimental data from a FMCW LiDAR system are used as a dataset. RC is employed to classify the dataset, and genetic algorithms are used to optimize RC. Traditional FMCW LiDAR ranging systems typically employ optical heterodyne coherent detection. After coupling the signal light with a reference light, an IF signal carrying distance and velocity information is generated at the photodetector. FFT is then applied to the IF signal, resulting in a narrow power peak in the power spectrum, enabling accurate distance and velocity demodulation. However, in practice, almost all wavelength-swept light sources cannot achieve ideal linear frequency sweeping. The nonlinearities in frequency sweeping can severely distort the FFT peak, broadening the spectrum and significantly reducing LiDAR's resolution and SNR, making target recognition challenging.

The objective of this study is to leverage the strong performance of RC in time-series classification to directly train and classify time-domain waveform signals obtained from FMCW LiDAR, without applying FFT. This approach avoids the spectral broadening issues introduced by FFT when handling nonlinear frequency sweeps. RC proves effective in capturing the features of temporal signals and addressing such issues. A genetic algorithm is employed to simultaneously optimize the RC model, using RMSE as the objective function to guide the search for improved parameter settings.

This paper provides an overview of RC theory and its development, facilitating the integration of theory and practical applications. Through genetic algorithm optimization, an approach to RC parameter selection is proposed, mitigating the impact of nonlinear frequency sweeping, enabling faster and more accurate feature extraction, classification, and recognition of results. This work has implications for the classification of temporal signals and related research.

The development of RC has evolved from the progress in machine learning, artificial intelligence, artificial neural networks, and recurrent neural networks. The following sections will delve into the background and developmental history of these fields.

1.2 Background

1.2.1 Machine Learning

Machine learning is a category of algorithms and methodologies that enable computer systems to enhance their performance automatically by learning from data and patterns, constituting a subset of artificial intelligence. These algorithms and methodologies empower computers to make predictions, classifications, decisions, and recognize patterns based on historical data without the need for explicit rule-based programming. In his book on machine learning, computer scientist Tom M. Mitchell provides a formal definition: "A computer program is considered to learn from experience E concerning some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E." [1] The field of machine learning primarily centers on the design and analysis of algorithms that facilitate automatic learning in computers.

Machine learning algorithms encompass a diverse set of techniques aimed at extracting and leveraging patterns from data for making predictions about unknown data instances. Due to the substantial statistical underpinnings of learning algorithms, machine learning is intimately connected with inferential statistics, often referred to as statistical learning theory. In terms of algorithm design, machine learning theory emphasizes the development of algorithms that are both implementable and effective in minimizing error accumulation. Since many inferential challenges entail non-procedural decision-making, a portion of machine learning research is dedicated to crafting approximative algorithms that are computationally tractable.

In 1936, Alan Turing introduced the Turing Machine, a foundational concept in computer science and theoretical computing.[2] This abstract model inspired computational theory and computer science, impacting our understanding of problem solvability, computational complexity, and algorithm design.

In the 1940s and 1950s, researchers drew inspiration from the nervous systems of living organisms and initiated the development of neural networks. In 1943, Warren S. McCulloch and Walter Pitts introduced the McCulloch-Pitts model,[3], a fundamental binary logic model illustrating neuron operation in early neural networks.

During the 1950s and 1960s, scholars concentrated on employing logic and symbolic reasoning to attain the goals of artificial intelligence. In 1956, Newell and Simon introduced the Logic Theory Machine, an early AI system simulating human reasoning.[4] It used symbolic logic and Production Rules for logical problem-solving, contributing to AI research.

From the 1980s to the early 2000s, advancements in computer hardware paved the way for more sophisticated machine learning algorithms and models. This period stands as a significant milestone in the history of machine learning, setting the stage for subsequent research and practical applications. In 1986, Backpropagation, introduced by Rumelhart, Hinton, and Williams, revitalized deep learning. It enables efficient training of deep neural networks like MLP by iteratively propagating errors backward through network layers and updating weights.[5] This process improves representation learning and generalization, leading to its widespread use in image recognition, natural language processing, and more.

In 2001, statistician Leo Breiman introduced "Random Forests,"[6], a machine learning algorithm characterized by three main features: random sampling of training data, random feature selection during tree node splitting, and integrated learning through voting or averaging. Random Forests enhance model generalization and adaptability, particularly for high-dimensional data and handling missing values. In 1995, Cortes and Vapnik introduced SVMs [7], influential for classification and regression. SVMs excel in solving linear and non-linear separable problems, with strong generalization. They identify optimal hyperplanes to maximize class separation, with support vectors. SVMs evolved with Kernel Methods and impacted complex machine learning and deep learning models.

Integrated learning encompasses the fusion of outputs from multiple learners to enhance generalization performance. In 1996, Leo Breiman introduced "Bagging" (Bootstrap Aggregating) as an integrated learning method.[8] Bagging involves random sampling (bootstrap) from the original training dataset to create multiple sub-training sets, training a learner on each sub-training set, and combining their predictions through voting or averaging. Bagging mitigates model variance, particularly for high-variance learners like decision trees.

Yoav Freund and Robert E. Schapire introduced "Boosting" as an integrated learning technique.[9] Boosting entails sequentially training a series of weak learners and adjusting sample weights based on the previous learner's performance. Misclassified samples receive increased attention in subsequent learners. Boosting effectively combines weak learners into a strong learner and performs well in addressing complex problems.

In the late 1980s, neural networks faced challenges due to training difficulties and computational constraints. However, in the early 2000s, with increased computational power and large-scale datasets, deep learning experienced a resurgence, representing a significant advancement in machine learning. In 2006, Hinton and Salakhutdinov introduced "Dimensionality Reduction with Neural Networks" (DRRS), featuring autoencoders for high-dimensional data reduction.[10] Autoencoders reduce reconstruction errors, extracting essential features in their hidden layer. They are crucial for nonlinear dimensionality reduction and widely applied in image processing, speech recognition, and natural language processing.

Over recent decades, machine learning has undergone rapid development and widespread adoption, solidifying its position as a core branch of artificial intelligence. A plethora of learning algorithms and techniques, encompassing supervised learning, unsupervised learning, reinforcement learning, and deep learning, have been proposed and refined. Supervised learning relies on labeled training data to establish mappings between inputs and outputs, facilitating prediction [11] and classification [7]. Unsupervised learning [10] aims to discover latent structures and patterns within unlabeled data. Reinforcement learning [12] involves learning through interaction between an agent and its environment, optimizing cumulative rewards. Deep learning, a subset of machine learning, employs multi-layer neural networks for data learning and representation, achieving groundbreaking results in image recognition [13], natural language processing [14], and speech recognition [15]. With the continuous advancement and refinement of these methods, machine learning has made significant contributions to various domains, including natural language processing, medical diagnostics, financial forecasting, and recommendation systems.

1.2.2 Artificial Intelligence

Artificial Intelligence, commonly referred to as AI, represents a burgeoning field within technological science. It encompasses research and the development of theories, methodologies, techniques, and application systems aimed at modeling, extending, and augmenting human intelligence. AI resides as a subset of computer science, seeking to comprehend the essence of intelligence and produce intelligent machines capable of responding akin to human intelligence. According to John McCarthy, often hailed as the father of AI, it is "the science and engineering of making intelligent machines, especially intelligent computer programs." AI endeavors to imbue computers, computer-controlled robots, and software with the ability to exhibit intelligent reasoning, mirroring human cognitive processes. This is achieved through the study of how the human brain thinks, learns, decides, and problem-solves, with the outcomes serving as the foundation for the development of intelligent software and systems.

Early AI research concentrated on aspects such as reasoning, problem-solving, and symbol processing. This era yielded initial AI techniques like logical reasoning and expert systems. In 1956, the Dartmouth Conference marked AI's formal proposal. It brought scholars together to explore computer intelligence. That year, Allen Newell and Herbert A. Simon introduced the Logic Theory, a significant milestone in early AI development. [16]

The 1970s and 1980s witnessed a pivotal focus on knowledge bases and expert systems within AI research. Expert systems, computer programs grounded in the knowledge of domain experts, simulate human experts' decision-making and reasoning processes within specific domains. In 1977, Feigenbaum introduced the concept of “knowledge engineering,” which laid the theoretical foundation for expert systems and helped transition AI from theory to practical application. [17] In 1976, Edward Shortliffe introduced MYCIN, an early expert system for medical decision-making, impacting AI in medicine. [18]

During the 1990s and 2000s, AI entered a phase where knowledge representation and machine learning techniques merged, catalyzing the development of more intelligent systems. This period marked a pivotal turning point for AI, where the integration of knowledge representation and machine learning propelled the field forward, enabling AI systems to learn from data and continually enhance their performance. In 1992, Langley, Iba, and Thompson's analysis of Bayesian classifiers at an AI conference provided a foundational theoretical basis for applying Bayesian methods in knowledge representation and machine learning. [19] Bayesian classifiers are now widely used in AI research, particularly for classification and pattern recognition.

The 2010s ushered in a new era in AI with the development of deep learning technology, unleashing a wave of advancements. Deep learning harnesses multi-layer neural networks for end-to-end learning, capable of extracting intricate features from vast datasets. In 2012, Krizhevsky, Sutskever, and Hinton introduced AlexNet, a deep CNN model, revolutionizing computer vision with its breakthrough in image classification, object detection, and semantic segmentation. [13] In 2014, Ilya Sutskever, Oriol Vinyals, and Quoc V. Le introduced Sequence-to-Sequence (Seq2Seq) learning, a transformative approach that excelled in natural language processing. [20]

Seq2Seq learning addressed the challenge of processing sequences with varying input and output lengths, providing a foundation for machine translation, text summarization, and dialog generation. In 2016, at CVPR, He, Zhang, Ren, and Sun introduced Residual Learning, or ResNet, resolving deep neural network training challenges with "residual blocks." [21] ResNet significantly improved performance in computer vision tasks, inspiring subsequent deep learning models. In 2017, Ashish Vaswani and others introduced the Transformer model, showcasing substantial advancements in natural language processing and sequence modeling. [22] The Transformer model overcame limitations in processing long sequences by employing self-attention mechanisms to capture interdependencies between elements within sequences. This innovation significantly enhanced machine translation and other sequence modeling tasks.

1.2.3 ANN

Artificial Neural Network, commonly referred to as ANN, serves as a computational model inspired by the biological nervous system, designed to emulate the information transmission and processing that transpires between neurons in the human brain. It stands as a fundamental component in the domain of deep learning and finds extensive application in tasks such as image recognition, natural language processing, and speech recognition. At its core, an ANN comprises neurons, each equipped with one or more inputs. These inputs undergo multiplication by corresponding weights and subsequently pass through an activation function. The assigned weights signify the significance of each neuron in information processing, while the activation function introduces nonlinearity, empowering the neural network to tackle intricate problems. ANNs typically encompass multiple layers, including an input layer, one or more hidden layers, and an output layer. The input layer accepts raw data, the hidden layer is responsible for processing and learning patterns and features from the data, and the output layer generates the final prediction. Each layer consists of multiple neurons.

The genesis of ANN can be traced back to the 1940s and 1950s, with its roots entrenched at the intersection of neurobiology and computer science. In the year 1943, Warren Sturgis McCulloch and Walter Pitts presented a simplified model of the neuron, famously referred to as the "McCulloch-Pitts Neuron." [3] This groundbreaking work marked a pivotal moment in the field of artificial neural networks, as it laid the theoretical foundation for neural network development. The duo introduced an abstract mathematical model to elucidate the functioning of biological neurons. Their model depicted neurons receiving multiple input signals, aggregating these signals with specific weights, and generating an output through a threshold function. This model effectively simulated the excitatory and inhibitory behaviors of neurons and could be represented through logical operators.

In 1958, Frank Rosenblatt meticulously elucidated the perceptron model and conducted an extensive exploration of its applicability in modeling biological neurons and information processing. [23] The perceptron, characterized as a single-layer neural network, is regarded as a seminal and pivotal contribution to the domain of artificial neural networks. In 1986, the backpropagation algorithm was introduced by David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. [5] It's a crucial method for training multilayer neural networks, forming the basis for the resurgence of neural network research and the subsequent development of deep learning.

In the 1990s, the utilization of neural networks faced limitations due to the restricted computational resources available at the time. Nonetheless, neural networks managed to make significant strides in various domains, including pattern recognition, time series analysis, and control systems. In 1998, Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner pioneered the application of gradient-based learning methods to recognize handwritten characters using a CNN and the MNIST dataset. [24]In 1998, Zhang, Patuwo and Hu published a comprehensive review of artificial neural network approaches to time series forecasting, covering both theoretical foundations and empirical evaluations.[25] Their work highlights neural networks' adaptability in handling complex nonlinear systems, offering valuable guidance for practical applications.

In the 2010s, deep learning emerged as a dominant research avenue in machine learning, driven by heightened computational capabilities, expansive datasets, and more efficient training algorithms. The successful deployment of multi-layer neural networks, known as deep neural networks, catalyzed significant advancements in image recognition, natural language processing, and speech recognition. In 2012, Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton introduced AlexNet, a deep convolutional neural network model. [13] Their work set a breakthrough for image classification, fueled interest in deep learning, and influenced the development of more complex CNNs. In 2014, Sutskever, Vinyals, and Le introduced the Seq2Seq model, a crucial advancement in sequence-to-sequence learning using RNNs or LSTMs. [20] This work greatly influenced natural language processing and set the stage for the Transformer model, impacting models like BERT and GPT. In 2016, Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun introduced ResNet, a method addressing gradient vanishing and degradation in deep network training by employing residual blocks. [21]

In recent years, new neural network architectures and techniques have greatly advanced artificial intelligence. In 1997, Hochreiter and Schmidhuber introduced LSTMs to address short-term memory limitations in traditional RNNs. [26] In 2013, Graves, Mohamed, and Hinton proposed a Deep RNN for speech recognition, spurring advancements in the field and laying the foundation for complex sequence modeling networks like LSTM and Transformer. [27] In 2017, Ashish Vaswani et al. introduced the Transformer, a neural network architecture that greatly improved machine translation and became the base model for numerous natural language processing tasks. [22]

1.2.4 RNN

The Recurrent Neural Network, commonly abbreviated as RNN, represents a sophisticated and highly specialized artificial neural network model meticulously engineered to excel in the intricate realm of sequential data processing. It emerges as a distinguished subset within the overarching domain of ANNs and exhibits remarkable prowess in dealing with data that unfolds chronologically or in a sequential fashion. This unique capability finds its applications across a diverse spectrum of domains, including but not limited to natural language understanding, speech recognition, music composition, and the analysis of complex financial phenomena such as stock price movements.

The architectural blueprint of the RNN is profoundly inspired by the recurrent connections observed within the intricate tapestry of the biological nervous system. These biological connections endow neural networks with the capacity to perpetually propagate information within the network, thereby preserving and memorizing essential data patterns. At the core of RNN's definition lie several pivotal elements that collectively delineate its distinctive identity and functionality:

Sequential Data Processing: RNNs are designed for understanding sequential data, such as temporal sequences, text, and audio. They excel in capturing the order of data elements for improved contextual understanding.

Cyclic Structure: RNNs employ cyclic connections, allowing them to transmit information across time steps. This enables the incorporation of insights from past contexts, enhancing pattern recognition and predictive modeling.

Hidden State: RNNs maintain a hidden state at each time step, accumulating contextual information as they process the sequence. This hidden state evolves to capture temporal dependencies.

Input and Output Layers: RNNs have input and output layers. The input layer receives data from the current time step, and the output layer generates predictions or outputs. Inputs and outputs can vary in format based on the task.

Parameter Sharing: RNNs use the same parameters across different time steps, ensuring consistency in processing inputs from various time steps.

Gradient Propagation: RNNs initially faced gradient-related challenges during training, such as vanishing and exploding gradients. Advanced RNN structures like

LSTMs and GRUs have effectively addressed these issues.

In summation, the RNN stands as a formidable and indispensable neural network model meticulously tailored for the intricate domain of sequential data processing. Its distinctive hallmark, characterized by a recurrent architectural motif and the pivotal role played by hidden states, empowers the network to imbibe and retain contextual information, thereby encapsulating prior knowledge within its own structure. This remarkable flexibility has ushered RNNs into a multitude of applications across diverse domains, encompassing natural language understanding, speech recognition, and the prediction of intricate time series phenomena. However, it is crucial to remain cognizant of the inherent limitations of RNNs, including their susceptibility to information loss when confronted with extensive sequences.

The roots of RNNs extend as far back as the 1980s. In 1990, Jeffrey L. Elman introduced the Elman network, the earliest RNN architecture for modeling temporal dependencies in sequential data. This marked the beginning of RNN development, paving the way for more complex models like LSTM and GRU, and inspiring research in temporal data processing, such as natural language, audio, and time series. [28] In 1997, Michael I. Jordan's research explored sequence processing and parallel distributed processing methods. [29] He discussed challenges in handling sequential data in neural networks, introduced the Parallel Distributed Processing Model for managing sequential order, and discussed cyclic structures. His work provides insights into sequence processing, benefiting the development of RNNs and other sequence models.

In 1997, Hochreiter and Schmidhuber introduced LSTM, a groundbreaking RNN variant, to address the gradient vanishing problem in traditional RNNs. [26] LSTM effectively captures and handles long-term dependencies, becoming a vital tool in deep learning for sequence data modeling. In 2014, KyungHyun Cho and team introduced the encoder-decoder structure, a crucial advancement in NMT. [30] This structure simplifies and improves RNNs to tackle gradient issues in NMT and represents a shift from traditional statistical models in machine translation.

1.2.5 RC

Reservoir Computing, an intriguing and innovative machine learning paradigm commonly denoted as RC, stands as a formidable approach dedicated to the nuanced realm of time series data analysis and the intricate modeling of dynamic systems. This sophisticated methodology fundamentally hinges upon the creation of a specialized RNN, notable for its consistent architectural structure often referred to as the "reservoir." What truly distinguishes reservoir computing is its unique approach to weight initialization, specifically within the reservoir. Here, the internal structure, encompassing a network of interconnected neurons, is meticulously imbued with random weights, while the primary thrust of the training process converges upon adapting and fine-tuning the weights of the output layer.

Central to the concept of RC is the reservoir network itself, a foundational architectural construct that invariably constitutes an extensive ensemble of neurons replete with a labyrinth of random connections. The defining characteristic of the reservoir network is its inherent stability, with the network's structural configuration held sacrosanct throughout the training phase. The primary mission entrusted to this reservoir network is the extraction and encapsulation of critical information from input data, often comprising sequences with temporal dependencies. Subsequently, this meticulously gleaned information flows seamlessly to the output layer, poised for further analysis and prediction.

Within the expansive domain of RC, the output layer occupies a pivotal role. Its responsibility lies in the generation of predictions, classifications, or other relevant outputs. This intricate process involves a judicious amalgamation of the state information derived from the reservoir network with the specific task at hand. As a consequence, the weights governing the output layer undergo a rigorous process of training and refinement, aligning the network's predictive capabilities with the desired objectives.

A distinctive hallmark of RC is the stringent adherence to the principle of maintaining constant internal weights within the reservoir network—a salient feature that underpins its efficacy and sets it apart from conventional neural network paradigms. This unique architectural facet significantly contributes to the stability of the training process, as RC remarkably exhibits insensitivity to the initial weights.

Practical applications of RC span a wide spectrum of domains and problem types. It has proven to be a potent tool, excelling particularly when confronted with sequential data characterized by intricate temporal structures. In this regard, RC has found substantial utility in diverse areas, including but not limited to time series prediction, speech recognition, image processing, and natural language processing.

The genesis of Reservoir Computing traces its origins to the early 21st century, with Herbert Jaeger's groundbreaking proposition of the "Echo State Network" in the year 2001. [31] This seminal work laid the foundation for the reservoir computing paradigm, emphasizing the preservation of the internal structure of recurrent neural networks while exclusively training the output layer. Notably, the reservoir network within the context of Echo State Networks embraces a dynamic assembly of neurons interconnected with a degree of stochasticity, thereby imbuing the network's internal dynamics with a distinctive and advantageous character.

In the subsequent year, 2002, Maass, Natschläger, and Markram introduced the LSM, a computational model harnessing the capabilities of spiking neurons. [32] The LSM's impact reverberates significantly in the realm of neural computation, infusing fresh ideas into the fields of pattern recognition, time series analysis, and beyond.

RC then saw a spurt of growth. Researchers have used RC for weather prediction [33], image recognition [34] and natural language processing [35] with good results. In the study researchers proposed the structure of optical reservoirs [36] and achieved more research development in recent years. For example, the optical reservoir constructed by Dmitriev et al. [37] using time-delayed structures and the optical reservoir constructed by Freiberger et al. [38] using the integration method are studies conducted for optical reservoirs in recent years.

In summation, Reservoir Computing emerges as a pioneering and potent machine learning approach tailored for the intricate terrain of time series data analysis.

Its unique utilization of a stochastic reservoir network with fixed internal weights empowers it to adeptly capture and exploit temporal dependencies. This paradigm, marked by its resilience and efficacy, has witnessed continuous evolution and harnessed its capabilities across a multitude of applications, thereby offering a fresh and innovative perspective on the intricate art of processing intricate sequential data.

1.2.6 GA

While RC offers the distinct advantage of a fixed structural design, coupled with minimal training requirements that pertain solely to the output layer's weight adjustments, it is imperative to acknowledge that, like any machine learning algorithm, it relies on a set of crucial parameters to operate effectively. To determine these essential parameters, the utilization of efficient optimization algorithms is of paramount importance. Among the array of optimization techniques available, GAs stand out as a choice endowed with unique advantages and applicability, rendering them an optimal selection for optimizing RC systems.

1. Applicability to Complex Problems: GAs exhibit a remarkable versatility, particularly well-suited for tackling intricate optimization challenges. Their efficacy shines through when dealing with large-scale, high-dimensional, and nonlinear problems. Notably, they do not impose constraints based on the specific form or continuity of the problem, making them adaptable to a wide range of scenarios.

2. Global Search Capability: GAs function as global search algorithms, proficient in scouring the solution space for multiple local optima. Their capacity to explore a diverse array of solutions ensures that they have the potential to uncover the global optimal solution. This feature proves invaluable when confronted with problems boasting numerous local optima.

3. Parallelism Advantage: GAs lend themselves seamlessly to parallelization. By concurrently evaluating and evolving a set of individuals (candidate solutions), they offer a significant edge in distributed computing environments. This parallelization

capability can significantly expedite the optimization process.

4. Adaptive Nature: GAs epitomize adaptability, adjusting their parameters and evolutionary strategies organically as they progress through the optimization journey. This adaptability equips them to contend with the evolving complexities of the problem space.

5. Exploration of Solution Space: GAs employ crossover and mutation operations, injecting diversity into the solution space. This inherent diversity aids in the comprehensive exploration of various solutions, mitigating the risk of being ensnared by local optima.

6. No Dependence on Derivative Information: In contrast to certain optimization algorithms, such as gradient descent, GAs operate without a dependence on derivative information pertaining to the objective function. This attribute renders them particularly suitable for scenarios where derivatives cannot be readily computed or are unavailable.

The historical trajectory of GAs extends back to the early decades of the last century, with John Holland emerging as a pioneering figure in the late 1950s and early 1960s. His groundbreaking work, initiated at the University of Utah, laid the foundation for the concepts that underpin GAs. Holland introduced the notion of adaptive systems—systems capable of enhancing their performance through selection and evolution in response to changing environmental conditions. Within this conceptual framework, he introduced key elements such as populations, individuals, genes, crossover, and mutation. These foundational concepts later evolved into the core components of Genetic Algorithms. John Holland's 1975 book, "Adaptation in Natural and Artificial Systems," is a seminal work in evolutionary computation and Genetic Algorithms, shaping the field and laying the foundation for Genetic Algorithms. [39]

Research on GAs grew rapidly, with applications in optimization, machine learning, and control. The 1990s and 2000s saw diverse GA applications in evolutionary computation, multi-objective optimization, artificial life, and more. In 2002, Kalyanmoy Deb introduced NSGA-II, addressing multi-objective optimization challenges. [40] In 1991, Thomas S. Ray used Genetic Algorithms to explore synthetic life, simulating the evolution of life-like agents. This pioneering work contributed to artificial life research. [41]

In the 2010s, Genetic Algorithms and deep learning converged for neural network structure search and optimization. Notable work includes Erick Real and team's 2017 use of GAs to find high-performance CNN architectures, showcasing GAs' effectiveness in optimizing neural networks. [42]

In conclusion, Genetic Algorithms stand as a potent and versatile optimization tool that complements the field of Reservoir Computing and diverse problem domains. With their rich history and adaptability to intricate optimization tasks, GAs continue to be a driving force in the pursuit of efficient and effective solutions across a wide spectrum of applications.

Summary:

This chapter delves into the origins and research context of RC and GAs. It outlines the research objectives of the study. The chapter begins by providing a comprehensive overview of RC as a machine learning method, tracing its development from the early days of machine learning in the 1930s and 1940s. It further explores the evolution of artificial intelligence, artificial neural networks, and recurrent neural networks as these fields gradually matured. It culminates with the formal introduction of the RC concept at the outset of the twenty-first century. Over the subsequent two decades, RC has experienced significant development and refinement, demonstrating exceptional performance in various applications, including time series prediction, speech recognition, image processing, and natural language processing.

Additionally, this chapter offers an in-depth examination of the historical progression of GAs and underscores the rationale for employing GA to optimize RC.

Chapter 2

Overview of Reservoir Computing and Genetic Algorithm

2.1 Principles of RC

2.1.1 Principle

RC represents a distinct paradigm within the realm of machine learning and neural network methodologies, constituting a specialized variant of RNNs. RC encompasses two primary architectural configurations: the ESN and the LSM. These structures can be harnessed either independently or in combination to construct a comprehensive RC system. Diverging markedly from conventional recurrent neural networks, RC introduces a set of pivotal characteristics that set it apart. Central among these features are the incorporation of a fixed, randomly initialized hidden layer, commonly referred to as the "reservoir," alongside an output layer, typically of a linear nature. The archetypal RC model comprises three fundamental layers: an input layer, a reservoir layer, and an output layer, as elucidated in Figure 1. The reservoir layer's static, stochastically generated internal architecture is purposefully endowed with dynamic intricacies and complexities, strategically tailored to apprehend the temporal nuances and salient attributes present within the input data stream.

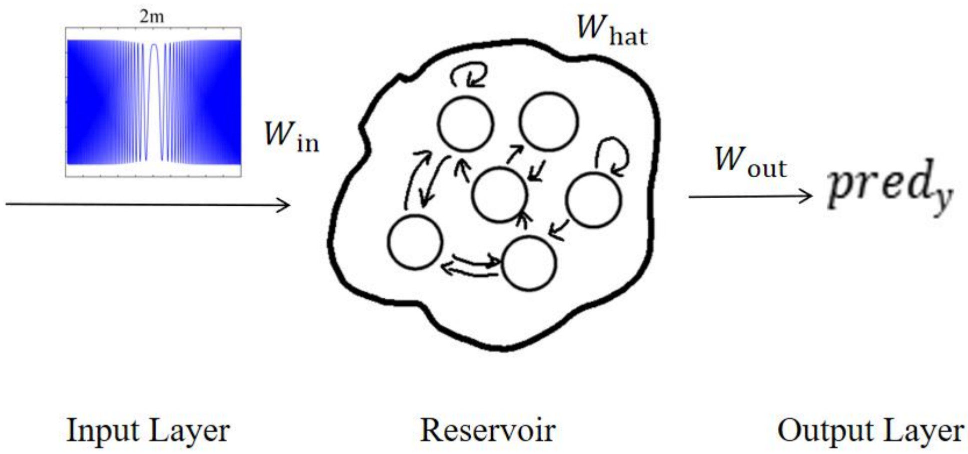


Figure 1 Structure of Reservoir Computing

The Reservoir constitutes the nucleus of the RC architecture, serving as its foundational element. It consists of a hidden layer densely populated with neurons, intimately interconnected via an intricate network of random connections. These inter-neuronal connections feature weights that are subject to stochastic initialization, persisting in a static state throughout the entirety of the training regimen. This intrinsic stochasticity generates a variety of dynamic behaviours within the Reservoir, thereby endowing it with the capacity to capture non-linear intricacies and temporal dependencies inherent in the input data.

The input data is linearly transformed using a predetermined weight matrix before being passed into the reservoir. Noteworthy is the immutability of this mapping, which remains unaltered during the course of training. As an outcome of the random connectivity and the intricate dynamic milieu intrinsic to the Reservoir, it inherently exhibits memory attributes and temporal processing prowess. These attributes render the Reservoir eminently suitable for handling temporal data and executing sequential tasks.

The insights distilled from the Reservoir traverse into a linear output layer, the weights of which are customarily refined via the training process. This output layer acts as the conduit through which the Reservoir-derived information is transmuted into the ultimate prediction or classification outcome. The training protocol for RC, in stark contrast to traditional recurrent neural networks, is characterized by its remarkable simplicity. The exclusive focus of the training regimen revolves around fine-tuning the weights of the output layer. The determination of these weight adjustments is facilitated by the ridge regression algorithm. Given the stochastic and dynamic character of the Reservoir, the need for its training or fine-tuning is obviated.

In the context of this research, the ESN structure of RC was employed. In this section, we present the mathematical model underpinning this structure. The process commences with the stochastic generation of matrices denoted as W_{in}, each having dimensions (res_units, input_features), wherein the values are randomly distributed within the range [0,1]. Here, res_units denotes the size of the reservoir, and input_features corresponds to the number of input features. Subsequently, another matrix, denoted as W_N and also possessing dimensions (res_units, res_units), is generated. However, the values within this matrix are drawn from a random distribution spanning the range [-0.5, 0.5]. Here, res_units signifies the dimensions of the reservoir layer, a quantity that warrants judicious consideration. Opting for an appropriate reservoir size is pivotal, as deviating from this balance can lead to undesirable consequences.

The reservoir weight matrix, denoted as $W_{\hbar\, at}$, can be precisely defined by the following equation:

$$W_{\hbar\, at} = \frac{\rho}{\theta_{max}} W_{in} \tag{1}$$

where ρ is the spectral radius and θ_{max} is the maximum eigenvalue of W_N.

In the subsequent training phase, we employ the ridge regression algorithm to compute the training output weight matrix, denoted as W_{out}. This weight matrix is essential for mapping the reservoir's state vector, S_j, to the desired output or prediction.

The computation of S_j, the current state vector, is achieved through the following equation:

$$S_j = \tanh\left(W_{in}X_j + W_{hat}S_{j-1}\right) \tag{2}$$

Here, X_j signifies the jth sample of the input data, S_{j-1} represents the state vector from the previous time step, W_{in} denotes the input matrix, W_{hat} corresponds to the weight matrix of the reservoir, and tanh signifies the hyperbolic tangent activation function. While only the tanh activation function is illustrated here, it's important to note that in practical applications, the sigmoid activation function can also be considered.

The subsequent step entails utilizing the ridge regression algorithm to compute the training output weight matrix, W_{out}, which plays a pivotal role in the overall functionality of the Reservoir Computing system.

$$W_{out} = \left(X^T X + \lambda \mathrm{I}\right)^{-1} X^T Y \tag{3}$$

where X is the input matrix, X^T is the transpose matrix of X, Y is the corresponding output matrix, I is the unit matrix, and λ is the regularization rate.

The calculation formula for $pred_y$, which represents the output of the ESN network, can be expressed as follows:

$$pred_y = softmax(\bar{y}) \tag{4}$$

Where $\bar{y}$ represents the output result obtained after applying the ridge regression algorithm, and its calculation formula is as follows:

$$\bar{y} = SW_{out} \tag{5}$$

Where S is the matrix of all states obtained after passing through the ESN network.Softmax is an activation function that takes a vector as input and returns the ratio of the exponent of each element to the sum of the elements. In this thesis softmax is formulated as

$$softmax(z_j) = \frac{e^{z_j}}{\sum_{i=1}^{K} e^{z_i}} \tag{6}$$

Where z_j denotes the jth element of the input vector and K denotes the length of the input vector. The function can be changed to compress each element to [0,1] and make the sum of all elements equal to 1. The final output result can be obtained from this.

Throughout this process, it becomes evident that the quantity of nodes within the RC system directly impacts computational workload, model fitting degree, and that the selection of the spectral radius holds paramount significance for ensuring RC's stability. Furthermore, there exists a multitude of choices for the activation function, and the magnitude of the regularization rate is intricately linked to both the training set's fitting degree and the model's complexity. It is thereby apparent that the meticulous determination of the various computational parameters stands as a pivotal concern for enabling the stable and efficient operation of RC.

2.1.2 Rating the accuracy

In our assessment of RC's performance, we incorporated metrics such as accuracy and RMSE. To evaluate accuracy, we initiated the process by comprehensively scanning the model's outcomes, identifying the highest value within each classification, and subsequently ascertaining the label associated with this maximum value. By calculating the number of precise classifications through a meticulous comparison between the classification result and the actual label, we derived the accuracy rate. This rate was obtained by dividing the count of accurate classifications by the overall total, as expressed by the following formula:

$$accuracy = \frac{Number\ of\ correctly\ classified\ samples}{Total\ number\ of\ samples} * 100\% \qquad (7)$$

The RMSE serves as a crucial indicator for quantifying the disparity between the acquired results and the ground truth values. In essence, a smaller RMSE signifies a higher proximity between the output results and the actual values, signifying better performance. Conversely, a larger RMSE indicates a greater divergence of the output from the true results. To establish a benchmark value for RMSE, we computed the RMSE of the training outcomes within the training dataset, which served as a reference for comparison with the RMSE of the test dataset. The formula for RMSE is articulated as follows:

$$err = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\bar{y}_i - y_i)^2} \quad (8)$$

where n is the number of samples, $\bar{y}_i$ is the output value of the ith sample, and y_i is the true value of the ith sample.

2.1.3 Overview of existing technology

In recent years, increasing attention has been directed toward the application of Reservoir Computing (RC) for time-domain signal classification, particularly through the development of physical RC systems that leverage the intrinsic properties of emerging hardware components. Notable implementations include systems based on memristors, thin-film transistors, and optical time-stretching techniques.

In a recent study conducted in 2025, Ankur Singh et al. proposed a dual-memristor RC architecture in which short-term memory was implemented within the reservoir using one memristor module, while long-term memory was embedded in the readout layer via a second memristor. This configuration achieved a classification accuracy of 98.84% on a speech digit recognition task, demonstrating the potential of memory-oriented hardware in time-series learning.[57] Similarly, in 2022, Gaurav et al. introduced a physical RC system utilizing three-terminal solid-state electrolyte ZnO/Ta_2O_5 thin-film transistors. [58] This architecture attained a classification accuracy of 94.44% on a handwritten digit recognition task using MNIST-like data. Furthermore, the system was capable of processing extended input sequences, indicating its applicability to tasks requiring long-term temporal dependency modeling.

Within the domain of optical RC, time-delay-based cyclic architectures employing optical time-stretching have emerged as a promising direction. These systems circumvent the need for electro-optical and opto-electrical conversions, enabling ultra-low latency and high-throughput processing. By generating virtual nodes across both temporal and spectral dimensions, they significantly expand the reservoir's dimensionality and enhance its information capacity, thereby improving its ability to distinguish and classify complex time-varying signals.

Between 2022 and 2024, Yue et al. systematically investigated the use of optical time-stretching for virtual node generation. Their findings confirmed that this approach substantially increases reservoir dimensionality and processing speed, while eliminating dependence on optoelectronic conversion.[59] In 2023, the proposed system achieved high performance in speech digit classification tasks. [60] Subsequently, in 2024, the integration of optical line scanning and single-pixel detection further improved both classification accuracy and speed, underscoring the suitability of optical RC for real-time time-series signal analysis.[61]

2.2 Principles of GA

Reservoir Computing, being a machine learning algorithm, necessitates the careful selection of various parameters, including the number of nodes, activation function, spectral radius, and more, to effectively carry out computations and problem-solving tasks. However, determining the optimal parameter combination for RC can be a challenging endeavor in the absence of empirical evidence. Randomly sampling parameters is not only time-consuming but also often yields suboptimal results due to the vast parameter space.

In this context, the GA emerges as a powerful tool for assisting in the selection of suitable parameter values for RC. GAs are global optimization algorithms grounded in the principles of biological evolution. They simulate the process of biological evolution, involving genetic material crossover, mutation, and natural selection, to search for optimal solutions. The evolutionary process unfolds across generations, starting with a randomly generated parent generation. Genotypes from the parent generation undergo crossover and mutation to produce offspring in the next generation. Fitness evaluation, achieved by assessing an objective function, identifies well-adapted individuals as elite offspring. These elite offspring then serve as the new parents for subsequent crossover and mutation, leading to the creation of the next generation. This iterative process continues until optimal offspring, capable of fulfilling the desired task, are obtained.

The genetic algorithm's ability to explore a wide range of parameter values and rapidly converge to promising results makes it particularly well-suited for assisting RC in parameter selection. This evolutionary approach to parameter optimization is depicted in Figure 2.

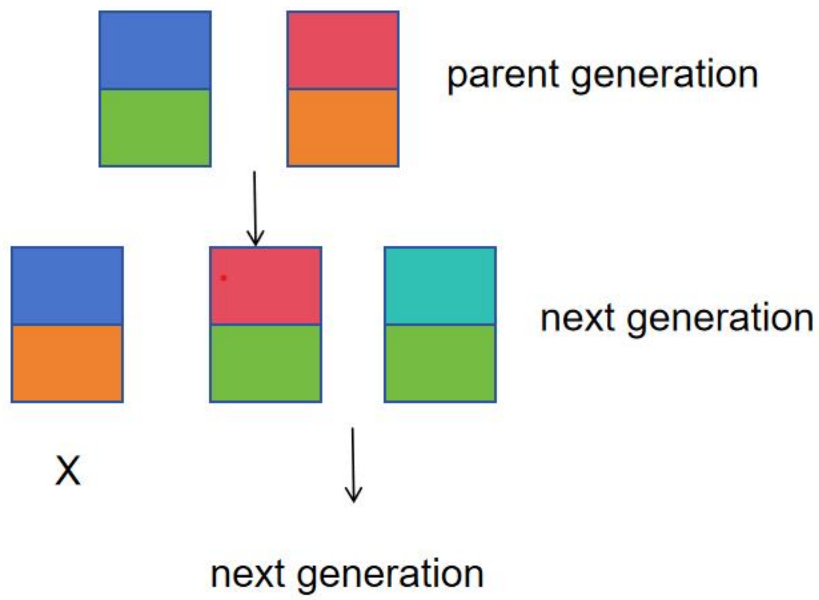


Figure 2 Schematic diagram of the genetic algorithm

The GA operates on a fundamental set of principles, which can be summarized in the following steps:

1. Initialization of Population: The process begins by forming a population of individuals. These initial solutions are randomly generated.

2. Fitness Evaluation: Each individual in the population is subjected to fitness evaluation. This evaluation results in a numerical fitness value, which quantifies an individual's performance within the problem domain.

3. Selection Operation: Based on the fitness values of individuals, a subset of individuals is selected to act as parents for the next generation. Individuals with

higher fitness possess a greater likelihood of being chosen as parents. This process aims to increase the representation of favorable genetic traits in subsequent generations.

4. Encoding DNA: The genetic information of each individual is encoded in a binary format. This encoding simplifies the processes of crossover and mutation.

5. Crossover Operation: In this step, the simulation of genetic crossover, inspired by biological genetics, takes place. Pairs of individuals are selected from the parent group, and offspring are generated by exchanging, combining, or recombining their genetic information.

6. Mutation Operation: Offspring are subjected to mutation, which involves random changes to their genetic information with a certain probability. Mutation introduces variability and novel gene combinations, thereby expanding the search space.

7. Population Update: replacing the original individuals with newly generated offspring to form a new population. The evaluation of the objective function determines which offspring are superior. The design of the objective function depends on the nature of the specific problem and is usually defined as minimizing or maximizing a particular performance metric or goal.

8. Iterative Process: Steps 2 through 7 are repeated iteratively until specific termination conditions are met, such as reaching the maximum number of iterations or achieving a satisfactory solution. With each generation, the individuals' adaptation improves, gradually leading to the discovery of better solutions.

This iterative and evolutionary approach allows genetic algorithms to effectively explore solution spaces and converge towards optimal or near-optimal solutions.

Summary

This chapter provides an in-depth look at the principles of RC, a unique paradigm in the field of machine learning and neural network methods.RC incorporates the ESN and LSM architectures, and consists of an input layer, a storage layer, and an output layer.The main feature of RC is the introduction of a fixed, randomly initialized "storage layer" with dynamic complexity, allowing it to capture the input data flow temporal nuances and properties in it. This chapter employs the ESN structure and discusses the foundations of its mathematical model, including the generation of weight matrices, the consideration of spectral radii, and ridge regression for training. The chapter also emphasizes the importance of reservoir size and activation functions. In addition, this chapter emphasizes the use of performance metrics such as accuracy and RMSE to evaluate the effectiveness of RC.

In addition, this chapter introduces GAs as a tool for parameter optimization in RC. Genetic algorithms are explained as a method to efficiently explore the parameter space and quickly converge to an optimal solution, especially in situations where parameter selection is challenging due to the large parameter space. The principles of genetic engineering are outlined, including population initialization, fitness assessment, selection, DNA coding, crossover, mutation, and population renewal. By efficiently optimizing key parameters, GA is portrayed as a powerful method for improving RC performance.

In summary, This chapter provides a comprehensive understanding of the principles of RC, ESN structure, and the application of genetic algorithms to optimize RC parameters. It lays the foundation for the subsequent chapters, which focus on empirical results and applications of RC in various scenarios.

Chapter 3

RC-based Temporal Signal Classification for High-Resolution Ranging Detection in LiDAR

3.1 FMCW LiDAR

In the research conducted in this thesis, the primary data source utilized for our Reservoir Computing model is experimental data obtained from a FMCW LiDAR system.

The FMCW LiDAR system represents an optical sensing technology designed for measuring the distance to a target. It is a specialized variant of LiDAR technology, which functions as an optical radar system used for precisely determining the distance and shape of a target through the emission of laser beams. FMCW LiDAR is distinguished by its capabilities, including robust resistance to interference, high resolution, a superior SNR, and low optical power requirements. These attributes make it a valuable tool in applications related to distance measurement, remote sensing, and three-dimensional (3D) imaging.

The fundamental operation of FMCW LiDAR systems typically involves the coherent detection method. For the purpose of range measurements, a laser source is initially employed to emit a CW laser beam. This laser beam typically maintains a constant frequency, and in our specific experiments, we utilized a frequency of 6.1 GHz. However, what sets FMCW LiDAR apart is the gradual modulation of the laser beam's frequency over time, which follows a linear slope. This rate of frequency change is termed the chirp rate. The frequency modulation results in the creation of a frequency gradient known as the modulation signal. Consequently, the laser beam's frequency undergoes continuous variation over time.

Upon striking the target, the emitted laser beam is reflected back. The returned optical signal carries essential distance-related information about the target. This signal is then combined with the locally generated laser beam, resulting in a mixed-frequency signal that contains details about both the distance and velocity of the target. The frequency of this mixed-frequency signal is directly associated with the distance to the target. This relationship is established because light waves introduce a phase difference during propagation, leading to observable frequency changes.

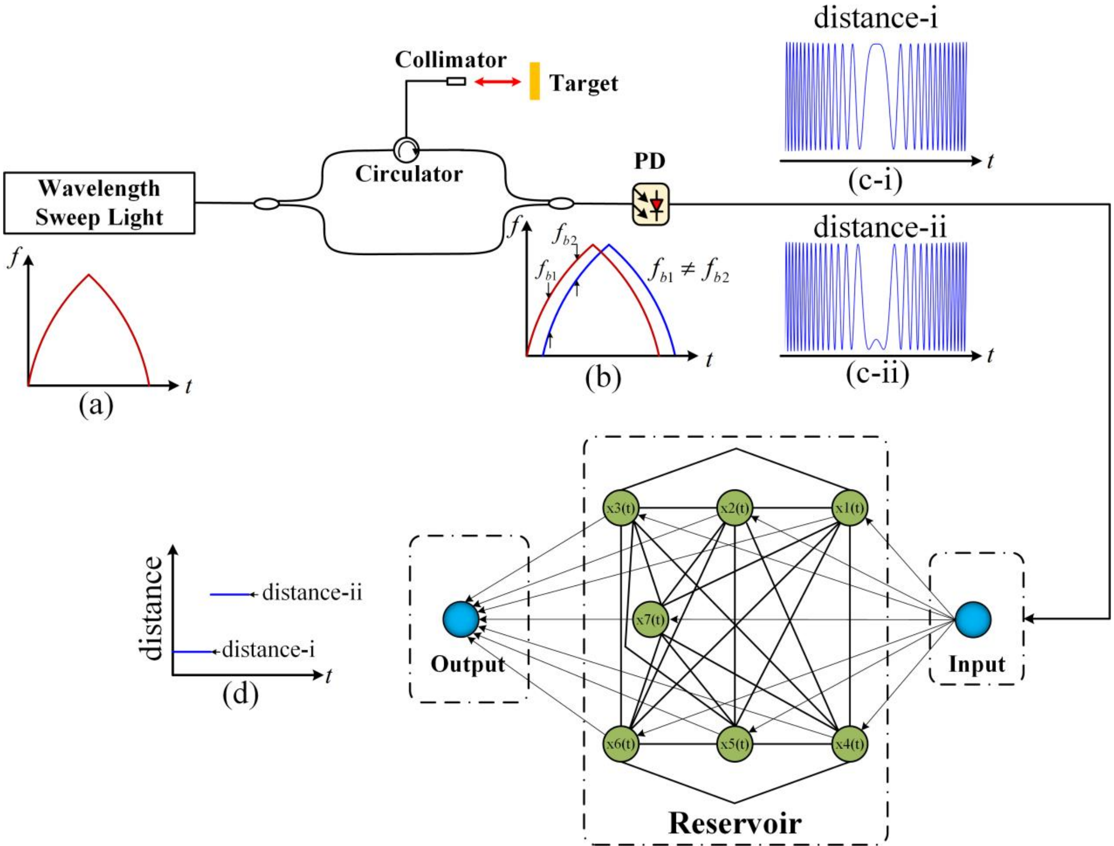


Figure 3 Schematic of the FMCW LiDAR system.

The mixed signal resulting from the interaction with the target is subjected to frequency analysis to extract its frequency components. The FFT method is employed for this purpose, facilitating the identification of a highly localized power peak within the power spectrum. This peak is instrumental in accurately demodulating information pertaining to both distance and speed.

3.2 Simulation

To illustrate this process, in our simulation experiments, we initiated the procedure by utilizing a FMCW signal featuring a nonlinear sweep range of 6 GHz. This signal served as the sweep source, generating a series of simulated time-domain waveforms. These waveforms were obtained at 1 cm intervals, corresponding to various distances ranging from 2 meters to 2.1 meters.

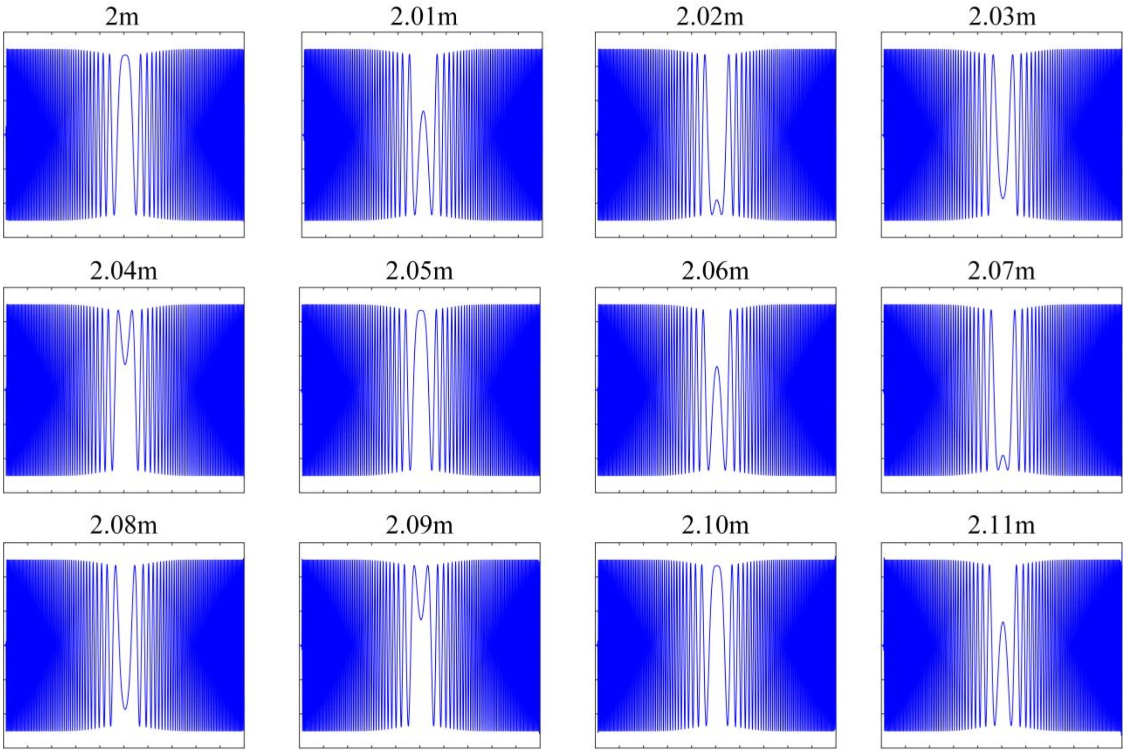


Figure 4 Waveforms of IF signals from 2m to 2.1m with an interval of 1 cm.

The figure clearly illustrates that each IF signal exhibits distinct waveforms. Notably, as the measurement distance progressively increases, the lower frequency waveforms exhibit a recurring pattern every 5 centimeters. It's important to emphasize that while this periodicity is observed, the waveforms themselves are not entirely consistent. This is due to the fact that higher frequency IF signals inherently result in denser time-domain waveforms.

Subsequently, we applied RC to process these eleven signals, and the outcomes are presented below in Figure 5.

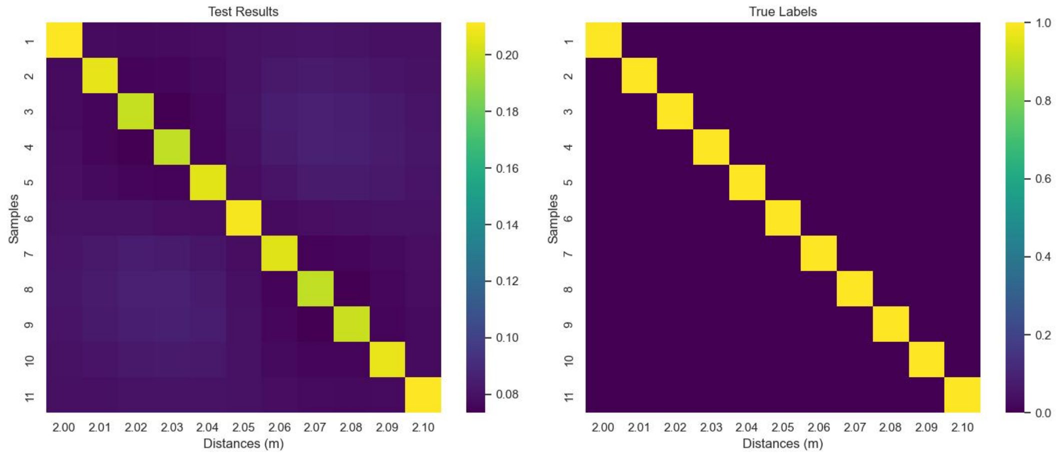


Figure 5 Results of waveform classification.

In this specific RC configuration, the reservoir size is set to 1000, the spectral radius of the reservoir weight matrix is 0.8, input scaling is adjusted to 0.2, the connection threshold between neurons or nodes is set at 0.3, the regularization rate is fixed at 500, and the tangent (tan) function is employed as the activation function. The results are depicted in the figure, where the left half represents the test results, and the right half represents the actual labels. A comparison between the two reveals a strong match, indicating that RC successfully achieved waveform classification within the range of 2 meters to 2.1 meters. Furthermore, it effectively classified waveforms with high similarity, such as those at 2 meters, 2.05 meters, and 2.1 meters, underscoring the method's robust waveform recognition capabilities.

3.3 Experiment

We then used the same parameters to train and test the data obtained from the experiments.

The experimental system mainly comprises the following functional modules. The system begins with a tunable continuous-wave ML operating at 1548.85 nm and 15.5 dBm output power, which provides the seed light. This beam is modulated in intensity via a MZM, driven by a 1 MHz triangular waveform at 2.6 Vpp, supplied by an arbitrary waveform generator (Agilent 33220A). The modulated optical signal is then directed, via an optical circulator, into a SL module based on a DFB design.

When the SL is not subjected to external optical injection, it operates at its inherent free-running frequency. However, by injecting the modulated ML signal into the SL and carefully tuning both the frequency detuning between the two lasers and the injection power, the SL is driven into a dynamic regime where its output exhibits periodic frequency oscillation. This oscillatory behavior, known for its stability and coherence, enables the generation of a time-varying wavelength output, forming the basis of the FMCW optical signal used in the system.

To extract the useful swept-frequency signal, the output from the SL is filtered using a programmable optical bandpass filter (WaveShaper 4000S) configured with a center wavelength of 1548.905 nm and a 3 dB bandwidth of 0.15 nm. This removes unwanted carrier components and residual noise. An erbium-ytterbium co-doped fiber amplifier is then applied to boost the filtered optical power, enhancing the system's overall signal fidelity.

A MZI is used to simulate changes in propagation distance, with one arm comprising a fixed 2-meter single-mode fiber and the other featuring a tunable optical delay line. Varying the delay in 50 ps increments effectively introduces path differences equivalent to 1 cm. The recombined optical signals are detected by a PD, generating an IF electrical signal, which is acquired by a high-speed oscilloscope (Tektronix DSS73304D) for subsequent waveform analysis using reservoir computing techniques.

The experimental system was designed and implemented by collaborating research groups. My primary contribution focused on the processing and analysis of the resulting experimental data, specifically through the application of RC for training and classification of time-series signals.

Seven long time-domain waveform signals containing a large number of cycles were obtained by sampling with the FMCW LiDER system, and the examples of the waveform signals we used are shown below in Figure 6.

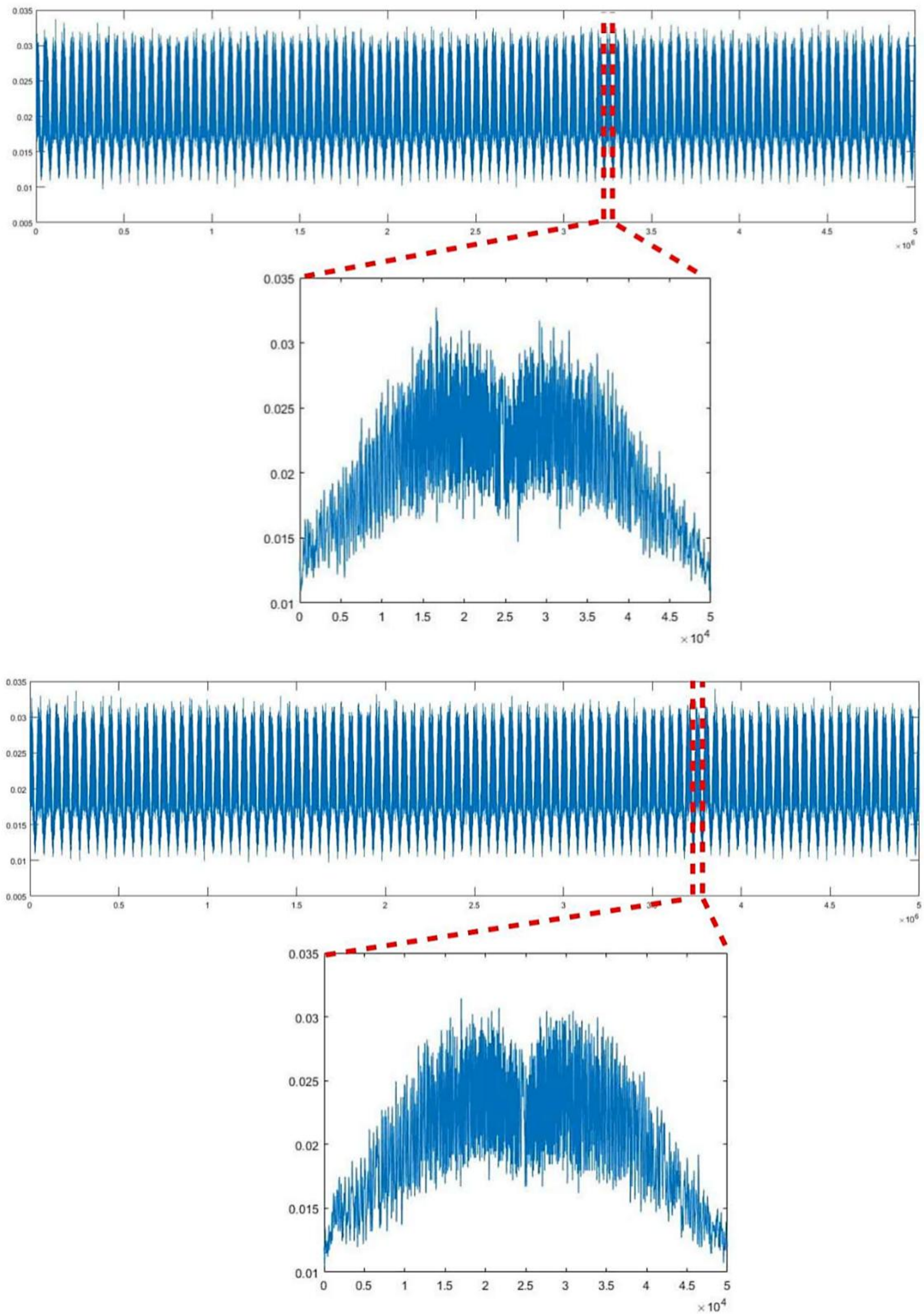


Figure 6 Waveforms measured at distances of 2m and 2.06m and individual waveforms used for training and testing.

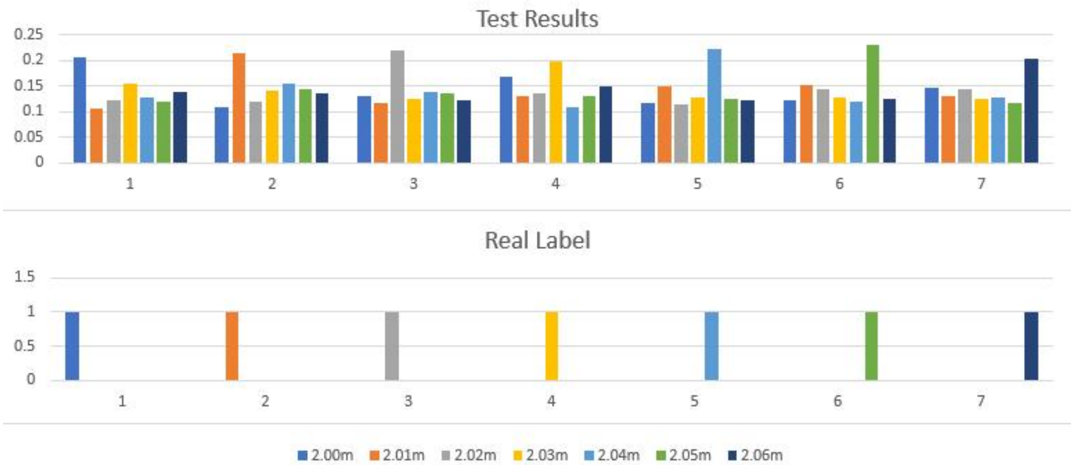
In our experimentation, we utilized real-world data with distances ranging from 2 meters to 2.06 meters. In the initial approach, we extracted 8 cycles from each dataset at varying distances, resulting in a dataset comprising 56 cycles intended for both training and testing. Employing the cross-validation technique, we randomly partitioned these 56 cycles into subsets using the KFold function, with each subset containing 4 cycles. Within each subset, one cycle was designated as the test set, while the remaining cycles were employed for training purposes. This allowed us to divide the dataset of 56 cycles into 42 cycles for training and 14 cycles for testing. We conducted four distinct sets of experiments, employing different training and testing sets for each iteration. The outcomes of these experiments are presented below in Figure 7.

Figure 7 Three sets of experiments, with test results above and real labels below, the y-axis denotes the proportion of each label.

The outcomes of the three experiments outlined previously unequivocally demonstrate the remarkable discriminative capability of the RC model in distinguishing distances ranging from 2 meters to 2.06 meters, as no errors were recorded in any of the experiments. All three experiments achieved a 100% accuracy rate, resulting in an average accuracy of 100%. Furthermore, we calculated the RMSE for these three experiments and compared it to the RMSE of the training set. The standard RMSE value may exhibit slight variations due to the differing selection of test and training sets in each group. However, for the sake of precision, we assume the value to be acceptable to six decimal places.

For the first training set, the RMSE was 0.281482, while the RMSE for the test set was 0.321265. In the second training set, the RMSE was 0.281484, and the RMSE for the test set was 0.320239. Finally, the RMSE of the third training set was 0.281484, resulting in a test set RMSE of 0.318830. It is evident that the RMSE of the test set in all three experiments deviates by only 0.04 from the standard value, which is an exceptional result.

In our second approach, we leveraged the 56 cycles collected during the first method as the training set. Subsequently, we randomly selected a cycle distinct from the training set for each distance segment, thereby creating a total of 8 test cycles. Additionally, we repeated this procedure with different data and random order to generate another test set. The experimental results shown in Figure 8 demonstrate the performance of the RC model under these testing conditions.



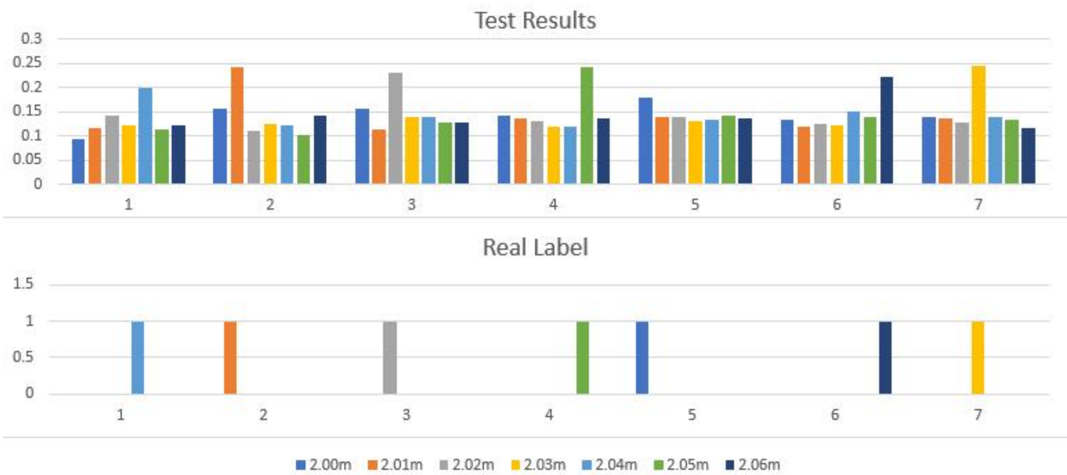


Figure 8 Sequential testing set data and shuffled testing set data, with test results above and real labels below, the y-axis denotes the proportion of each label.

The outcomes presented above demonstrate that RC excels in categorizing various distance information, whether presented in a sequential or random order. The overall accuracy remains at a remarkable 100%. Specifically, the standard RMSE registers at 0.281431, with the test data sorted sequentially yielding a result of 0.321188. On the other hand, the test data sorted randomly produces a result of 0.317738. Notably, the difference between the obtained results and the standard value is approximately 0.04, underscoring the consistent high performance of RC across diverse scenarios.

In the third approach, we maintain the previous training set and extract 8 cycles from distances of 2 meters and 2.06 meters, creating two test sets to evaluate the training effectiveness of RC with varying distance data. Furthermore, we select 4 cycles from 2.01 meters as the test set to assess the training performance of RC when provided with limited experimental data. The summary of the experimental results is presented in Figure 9.

Figure 9 Eight test cycles conducted at distances of 2.00m and 2.06m, and four test cycles conducted at a distance of 2.01m, with test results above and real labels below, the y-axis denotes the proportion of each label.

Based on the findings outlined above, it becomes evident that RC exhibits remarkable accuracy when confronted with single-distance signals. It can accurately discern distances of 2 meters and 2.06 meters with 100% precision. Furthermore, even when presented with a smaller input signal of 2.01 meters, RC maintains a perfect accuracy rate of 100%. These results underscore RC's ability to consistently deliver precise outcomes when dealing with single-distance signal inputs, regardless of their number.

In the experiment conducted at a distance of 2.00 meters, the standardized RMSE is 0.281431, while the RMSE of the test result is 0.324359. Similarly, in the experiment at a distance of 2.06 meters, the standardized RMSE remains at 0.281431, with the RMSE of the test result measuring 0.317049. For the experiment carried out at a distance of 2.01 meters, the standardized RMSE is 0.281431, and the test result yields an RMSE of 0.281431. Lastly, in the experiment performed at a distance of 2.01 meters, the standard RMSE is 0.281431, while the RMSE of the test result is 0.311488. It is noteworthy that the difference between the test result and the standard RMSE falls within the range of 0.03 to 0.04, indicating a very close alignment between the test result and the standard result. This underscores the consistently high performance of RC.

To summarize, the results affirm that RC excels in accurately classifying waveforms at varying distances. It consistently produces highly accurate results, whether dealing with ordered signals, disordered signals, single signals, or a smaller number of signals.

Summary

In this chapter, the experimental section was pivotal to our research, aimed at validating the performance of the Reservoir Computing model in handling data from a FMCW LiDAR system. FMCW LiDAR represents a highly precise optical sensing technology used for distance measurement and 3D imaging, favored for its outstanding SNR, interference resistance, and low power consumption.

Through a series of meticulously designed experiments, we conducted a thorough evaluation of the RC model in various testing scenarios. The results consistently demonstrated that the RC model achieved 100% accuracy in all experiments, successfully accomplishing waveform classification. This not only underscores the model's consistent performance in handling diverse signal inputs but also provides a robust foundation for its extensive potential in LiDAR applications.

Additionally, this chapter covered performance assessments under different testing conditions, including sequentially ordered signals and randomly ordered signals. In all circumstances, the RC model exhibited outstanding and consistent performance, with RMSE values deviating only slightly from standard values. These experimental results highlight the remarkable performance and reliability of the RC model, regardless of the diversity of signal inputs and testing conditions.

In summary, the experimental results in this chapter provide strong support for the RC model's high accuracy and robustness in handling FMCW LiDAR data. These findings hold significant implications not only for the outcomes of this research but also for providing a strong foundation for future applications in the field of laser radar. Subsequent research will focus on applying the RC model in real-case scenarios to explore its potential applications in different domains.

Chapter 4

Optimization of RC parameters using Generic Algorithm

4.1 Implementation of Generic Algorithm

The GA employed in this investigation was tailored to optimize the integrated parameters within the Reservoir Computing model. The principal objective is to devise a global optimization algorithm capable of effectively fine-tuning the pivotal parameters of the RC. In Chapter 2, we delineated the operational process of the RC, wherein numerous key parameters assume a fundamental role in ensuring the proper functionality of the RC.

Among these parameters, the size of the Reservoir holds particular significance. We advocate for a moderate reservoir size. If the reservoir is undersized, it may give rise to drawbacks such as insufficient capacity to capture the complexity and dynamics of input data, inefficiency in retaining and transferring crucial information, resulting in information loss, limited generalization capability of the RC model, challenges in handling unseen data, and suboptimal performance with novel data. Conversely, an excessively large reservoir presents drawbacks such as prolonged training time, heightened hardware resource requirements, overfitting to training data leading to compromised generalization performance on new data, and increased memory demands for storing connection weights and neuronal state information. Hence, judiciously determining the reservoir size is imperative.

Within ESN, the spectral radius assumes a pivotal role in ensuring system stability. Striking a balance is crucial, aiming for a spectral radius that is sizable enough for optimal performance yet not excessively large to maintain stability. This equilibrium is achieved by adjusting the ratio between the weight matrix's size and the spectral radius.

The regularization rate is intricately linked to RC fitting. Its purpose is to govern model complexity and mitigate overfitting. A higher regularization rate intensifies the penalization of the model by the regularization term, diminishing model complexity, lowering the fit to the training set, yet enhancing generalization to the test set. Conversely, a lower regularization rate increases model complexity, improving the fit to the training set but diminishing generalization to the test set.

Chapter 2 highlighted tanh and sigmoid as common activation functions, with inconclusive evidence on their relative advantages. Therefore, the choice of activation function emerges as a crucial key parameter.

Furthermore, the generation of input matrices W_{IN} and W_N during program execution involves several parameters, impacting the results. Scaling, a factor in W_{IN} generation during program execution, adjusts the relationship between input data and program weights.

In the context of RC, the reservoir is randomly generated. However, a crucial parameter must be defined in the program to regulate the proportion of weighted links in the reservoir. Connectivity serves as this parameter, indicating the proportion of weighted links involved in generating W_N.

Therefore, our optimization approach using GAs encompasses six pivotal parameters: spectral radius, lambda (regularization rate), res_units (number of nodes in RC), activation function, scaling, and connectivity. These parameters are amalgamated, with each group constituting an individual in the GA population denoted by S=(spectral radius,λ,res_units,Act,Scaling,Connectivity).

The specified value ranges for the six parameters subjected to optimization in this paper are as follows:

Parameter	Range
Spectral radius	0.5-1
Lambda (Regularization rate)	0-900
Res_units (Number of nodes in RC)	500-1200
Activation Function	tanh (0) or sigmoid (1)
Scaling	0.1-0.5
Connectivity	0.1-0.5

In the context of GAs, an objective function is employed to assess the superiority of offspring within the population, determining which individuals should be retained or discarded. Typically, this involves optimizing a specific metric, often aiming to minimize or maximize its value. In Reservoir Computing, RMSE serves as one of the metrics to gauge the accuracy of classification results. In our GA optimization, the objective function aims to minimize the RMSE, indicating that smaller RMSE values correspond to improved similarity between training and real results.

Following the objective function, each generation undergoes selection, eliminating the group of individuals with the highest RMSE. The remaining individuals constitute the new parent generation, engaging in crossover and mutation operations. This process iterates for 80 generations, obtaining RMSE values for each individual in the new child generation. The maximum value is eliminated, and the minimum value is retained, completing the operation.

The computational setup involves a 13th Gen Intel(R) Core(TM) i7-13650HX 2.60 GHz CPU and an NVIDIA GeForce RTX 4060 Laptop GPU. The initial population comprises 10 randomly generated individuals, with a crossover rate of 0.95 and a mutation rate of 0.05. The iteration termination condition is set to 80 iterations, signifying that the genetic algorithm concludes its iterations and outputs results after 80 cycles. The pseudo-code for this genetic algorithm procedure is as follows:

Initialization: A population is initially created, consisting of 10 individuals and denoted by S. Each individual contains six parameters, namely S = (spectral radius, lambda, res_units, Act, Scaling, Connectivity). Each individual is used to construct an RC network, and its objective function is evaluated.

Crossover and Mutation: New population sequences are generated through crossover and mutation operations. Each new individual creates a fresh RC network, and its objective function is assessed.

Selection of Offspring: Individuals meeting the objective function criteria are selected as good offspring. These successful offspring become the new parents for generating subsequent population sequences.

Termination: The algorithm concludes when the genetic algorithm satisfies the stopping condition. In this study, the termination condition is set to 80 generations of iterations.

The schematic diagram of the optimization process is shown in Figure 10 below.

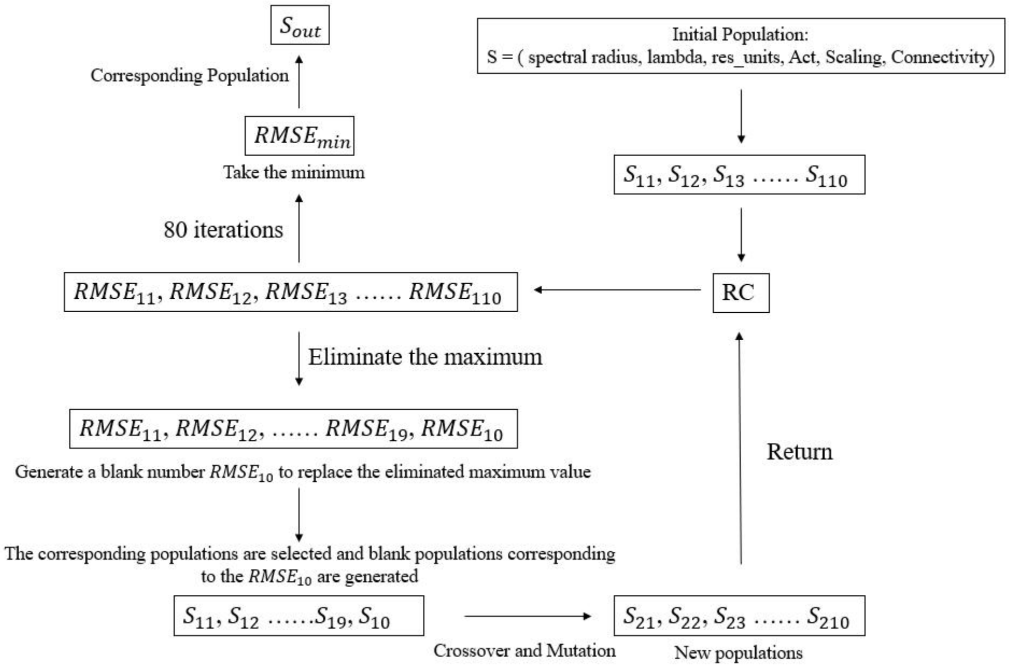


Figure 10 Optimization process

4.2 Optimisation Results

Figure 11 depicts the RMSE trend observed during the optimization process.

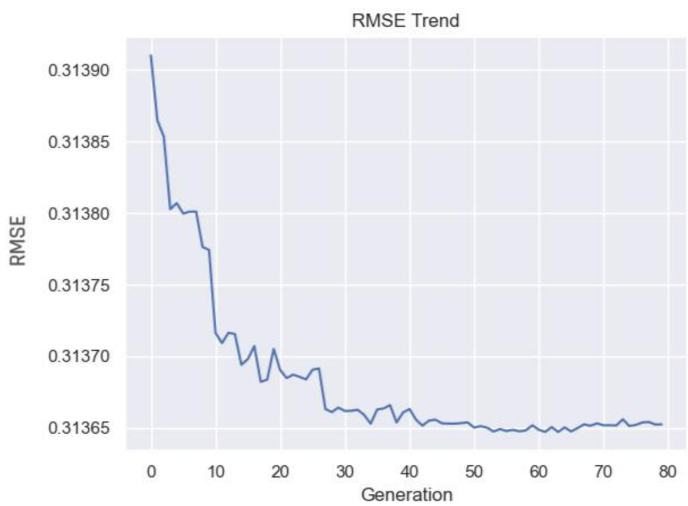


Figure 11 Decreasing curve of RMSE during 80 iterations of optimization

The results demonstrate that the genetic algorithm has the capacity to attain an optimal combination of parameters within the scope of the RC optimization task. After 80 iterations, it is noticeable that the RMSE of the test results versus the actual labels has decreased from 0.31391 to 0.31365 and has reached a stable state. This suggests that the minimum achievable RMSE within this range is approximately 0.31365. The optimized parameters, recorded to five decimal places, are as follows:

Parameter	Result
Spectral radius	0.63
Lambda	11.43950
Res_units	1196
Activation Function	tanh
Scaling	0.28807
Connectivity	0.25171

An observation of the RMSE trend curve reveals that the genetic algorithm reached around 0.3165 after roughly 50 iterations, and subsequently, the RMSE value stabilized until the 80th iteration when the algorithm concluded. It's worth noting that this optimization process was limited to 80 generations. If the termination condition were set to achieving a specific fitness value, it might be possible to reduce the RMSE value further with additional iterations. However, we believe that pushing the RMSE even lower is less likely to yield substantial improvements. Even after only 80 iterations, the RMSE is already approaching a plateau, suggesting that further iterations are unlikely to produce significant reductions in RMSE.

Moreover, it's interesting to explore alternative optimization goals, such as minimizing the runtime of the RC while maintaining accurate and discernible training results. Achieving better results with minimal time and energy consumption would be a valuable avenue for further research.

Summary

In this chapter, we delved into the optimization of the RC model's critical parameters using a GA. The objective was to fine-tune all six key parameters that play a pivotal role in RC's performance. We established the significance of parameters such as spectral radius, regularization rate (lambda), number of nodes in RC (res_units), activation function, scaling factor, and connectivity in shaping the efficacy of the RC model.

The GA framework allowed us to systematically explore a range of parameter values, seeking to minimize the RMSE as the objective function. A smaller RMSE signifies closer alignment between training results and actual values, indicating superior model performance. The specified ranges for the six parameters guided the optimization process, with the optimal values determined as follows: spectral radius (0.63), lambda (11.43950), res_units (1196), activation function (tanh for softmax), scaling (0.28807), and connectivity (0.25171).

The optimization process, conducted over 80 iterations, revealed a clear trend in RMSE reduction, culminating in a stable state. The minimum attainable RMSE within this range was approximately 0.31365, providing a strong foundation for the RC model's performance. It is worth noting that further iterations may yield diminishing returns, as the RMSE was already approaching a plateau after 80 iterations.

Looking ahead, there are intriguing prospects for exploring alternative optimization goals, such as minimizing the runtime of the RC model while maintaining high-performance standards. Balancing computational efficiency with accurate training results remains a compelling avenue for future research in the field of Reservoir Computing.

Chapter 5

Conclusions

5.1 Conclusion

The research detailed in this manuscript focuses on the complex process surrounding the classification of time series signals, employing the RC methodology and optimizing it using GAs.RC is an exceptionally powerful framework that has demonstrated excellence in efficiently dealing with a wide range of scenarios encountered in the analysis of one-dimensional time series signals.

The primary focus of the investigation detailed in this manuscript revolves around the intricate processes of classifying and optimizing time series signals, employing the cutting-edge methodology of Reservoir Computing. Demonstrating remarkable prowess, RC stands out as an exceptionally robust framework, showcasing superior capabilities in effectively addressing diverse scenarios encountered in the analysis of one-dimensional time series signals.

The empirical findings underscore the proficiency of RC in simulated scenarios, where the model exhibits precise identification and accurate classification of all simulated signals following a rigorous training regimen. In experimental contexts, RC demonstrates commendable performance by achieving 100% accuracy in the classification of various instances encompassing sequential, unordered, and single signals. This attests to the versatility of RC across a spectrum of real-world scenarios.

Further enhancing the adaptability and efficiency of RC, the conducted experiments integrate GA for parameter optimization. Notably, the optimization process reveals a convergence to the nadir of the optimization curve after 80 generations of iterative refinement. This culmination results in the derivation of an optimal set of parameter combinations, meticulously fine-tuned to bolster the operational efficiency of the RC model.

The outcomes of these experiments provide compelling evidence that the deployed GA not only facilitates the optimization of RC parameters but also significantly contributes to the amelioration of the overall efficiency and effectiveness of the RC model.

The outcomes of these experiments provide compelling evidence that the deployed GA not only facilitates the optimization of RC parameters but also significantly contributes to the amelioration of the overall efficiency and effectiveness of the RC model. This combination of techniques not only expands our understanding of RC's capabilities, but also highlights the potential for tailored optimization strategies that enhance RC's utility in a variety of signal processing domains.

5.2 Future Work

This study reveals a wide range of potential research directions. Firstly, it emphasizes the classification of time series signals using RC. While this thesis uses one-dimensional time-series signals from the FMCW LiDAR system, other optical test systems that generate one-dimensional time-series signals, such as fiber-optic sensing, offer similar potential for RC application. Furthermore, optical RC research has gained traction, where optical devices are used to construct RC reservoirs at a physical level. This approach allows the direct input of waveform signals from optical measurement systems into the RC training process, reducing computation in the computer, energy consumption, and simplifying result analysis.

In addition, RC also shows good ability in multi-signal classification. Analyzing fiber optic imaging and similar multi-dimensional time series signals can simplify the complexity of result analysis. Additionally, RC can tackle the integration of diverse data types, such as combining sensor, image, and text data for applications like emotion recognition systems and smart driving solutions. This entails integrating various data types into RC models to support complex decision-making.

The use of optics to construct physical reservoirs for RC has been identified as a promising avenue. However, physical reservoirs differ significantly from computer program-based RC, and tuning parameters that are critical in RC programs can be challenging in physical reservoirs. GAs, as global optimization tools, play a crucial role in determining tuning methods and degrees of tuning through simulation, offering a cost-effective and efficient approach to system development.

In conclusion, the future outlook of this study outlines a broad array of potential research directions and applications in related fields. It acknowledges the relevance of current technological trends and highlights opportunities for innovation. The paper serves as a reference for ongoing exploration of physical reservoir structures and methods, aiming to contribute to future scientific and technological advancements.

Reference

[1] T. M. Mitchell, Machine Learning. McGraw-Hill, 1997.

[2] A. M. Turing, "On Computable Numbers, with an Application to the Entscheidungsproblem," *Proceedings of the London Mathematical Society*, vol. 2, no. 42, pp. 230-265, 1936.

[3] W. S. McCulloch and W. Pitts, "A Logical Calculus of Ideas Immanent in Nervous Activity," *Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[4] A. Newell and H. A. Simon, "The Logic Theorist – A Case Study in Heuristic Problem Solving," *Journal of the ACM*, vol. 10, no. 2, pp. 151–256, 1956.

[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533-536, 1986.

[6] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[7] C. Cortes and V. N. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[8] L. Breiman, "Bagging Predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[9] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proceedings of the Thirteenth International Conference on Machine Learning (ICML '96)*, pp. 148–156, 1996.

[10] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[11] Y. LeCun, L. Bottou, G. B. Orr, and K. R. Müller, "Efficient BackProp," *Neural Networks: Tricks of the Trade*, pp. 9-48, 2012.

[12] R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction," *MIT Press*, 2018.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.

[14] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, "Recurrent

Neural Network Based Language Model," *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[15] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, pp. 369–376, 2006.

[16] A. Newell and H. A. Simon, "Report on a general problem-solving program," *RAND Corporation, Santa Monica, CA, Tech*. Rep. P-296, 1957.

[17] E. A. Feigenbaum, "The art of artificial intelligence: Themes and case studies of knowledge engineering," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, pp. 1014–1029, 1977.

[18] E. Shortliffe, "MYCIN: A Rule-based Computer Program for Advising Physicians Regarding Antimicrobial Therapy Selection," *Proceedings of the American Federation of Information Processing Societies*, 1974.

[19] P. Langley, W. Iba, and K. Thompson, "An analysis of Bayesian classifiers," in *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, 1992, pp. 223-228.

[20] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 3104-3112.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770-778.

[22] A. Vaswani et al., "Attention is All You Need," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 5998-6008.

[23] F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain," *Psychological Review*, vol. 65, no. 6, pp. 386-408, 1958.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324,

1998.

[25] G. Zhang, B. E. Patuwo, M. Y. Hu, “Forecasting with artificial neural networks: The state of the art,” *International Journal of Forecasting*, vol. 14, no. 1, pp. 35–62, 1998.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735-1780, 1997.

[27] A. Graves, A. R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6645-6649, 2013.

[28] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179-211, 1990.

[29] M. I. Jordan, "Serial order: A parallel distributed processing approach," *Advances in Psychology*, vol. 121, pp. 471-495, 1997.

[30] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[31] H. Jaeger, "The 'Echo State' Approach to Analysing and Training Recurrent Neural Networks," *GMD—German National Research Institute for Computer Science*, 2001.

[32] W. Maass, T. Natschläger, H. Markram, "Real-time computing without stable states: a new framework for neural computation based on perturbations." *Neural Comput*. 2002

[33] A. A. Ferreira and T. B. Ludermir, "Using Reservoir Computing for Forecasting Time Series: Brazilian Case Study," in *Proceedings of the 8th International Conference on Hybrid Intelligent Systems (HIS 2008)*, vol. I, pp. 602-607, 2008.

[34] Z. Tong and G. Tanaka, "Reservoir Computing with Untrained Convolutional Neural Networks for Image Recognition," *2018 24th International Conference on Pattern Recognition (ICPR)*, Beijing, China, pp. 1289-1294, 2018 .

[35] N. Schaetti, "Behaviors of Reservoir Computing Models for Textual Documents Classification," *2019 International Joint Conference on Neural Networks (IJCNN)*,

Budapest, Hungary, pp. 1-7, 2019.

[36] G. Traverso, M. Allmen, M. Meucci, L. Arecchi, and R. Ramponi, "Photonic information processing beyond Turing: an optoelectronic implementation of reservoir computing," *Applied Physics B*, vol. 90, no. 3-4, pp. 367-375, 2008.

[37] P. Dmitriev *et al*., "Analysis of time-delay photonic reservoirs for neuromorphic computing," *2022 Photonics North (PN)*, Niagara Falls, ON, Canada, pp. 1-1, 2022.

[38] M. Freiberger, S. Sackesyn, C. Ma, A. Katumba, P. Bienstman and J. Dambre, "Improving Time Series Recognition and Prediction With Networks and Ensembles of Passive Photonic Reservoirs," in *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 26, no. 1, pp. 1-11, Jan.-Feb. 2020.

[39] J. H. Holland, "Adaptation in Natural and Artificial Systems," *University of Michigan Press*, 1975.

[40] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, 2002.

[41] T. S. Ray, "An approach to the synthesis of life," *Artificial Life*, vol. 1, no. 1-2, pp. 179-209, 1991.

[42] E. Real, A. Aggarwal, Y. Huang, and J. Ha, "Regularized Evolution for Image Classifier Architecture Search," *arXiv preprint arXiv:1802.01548*, 2017.

[43] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable Architecture Search," *arXiv preprint arXiv:1806.09055*, 2018.

[44] A. A. Ferreira and T. B. Ludermir, "Genetic algorithm for reservoir computing optimization," in *2009 International Joint Conference on Neural Networks*, Atlanta, GA, USA, 2009, pp. 811-815, doi: 10.1109/IJCNN.2009.5178654.

[45] G. Jian, J. Sun, and M. Cong, "Research on an FM/cw ladar system using a 64×64 InGaAs metal-semiconductor-metal self-mixing focal plane array of detectors," *Applied Optics*, vol. 56, no. 10, pp. 2858-2862, 2017.

[46] R. J. Roy, M. Lebsock, L. Millán, et al., "Validation of a G-band differential absorption cloud radar for humidity remote sensing," *Journal of Atmospheric and Oceanic Technology*, vol. 37, no. 6, pp. 1085-1102, 2020.

[47] C. Li, F. Zhang, and X. Qu, "High-Resolution Frequency-Modulated Continuous-Wave LiDAR Using Multiple Laser Sources Simultaneously Scanning," *Journal of Lightwave Technology*, vol. 41, no. 1, pp. 367-373, 2023.
[48] J. Lee, Y. J. Kim, K. Lee, et al., "Time-of-flight measurement with femtosecond light pulses," *Nature Photonics*, vol. 4, no. 10, pp. 716-720, 2010.
[49] D. F. Pierrottet, F. Amzajerdian, L. B. Petway, et al., "Linear FMCW laser radar for precision range and vector velocity measurements," *MRS Proceedings*, vol. 1076, no. K04-06, 2008.
[50] L. Qiao, D. Sun, X. Zhang, et al., "Linearity Requirement for a Linear Frequency Modulation Lidar," *Optical Review*, vol. 6, no. 2, pp. 160-162, 1999.
[51] Z. Guo, G. Han, J. Yan, et al., "Ultimate Spatial Resolution Realisation in Optical Frequency Domain Reflectometry with Equal Frequency Resampling," *Sensors*, vol. 21, no. 14, pp. 4632, 2021.
[52] J. Xing, Y. Zhang, F. Wang, et al., "A Method Based on Time-Scale Factor for Correcting the Nonlinear Frequency Sweeping in an OFDR System," *IEEE Photonics Journal*, vol. 11, no. 3, pp. 7101808, 2019.
[53] K. Iiyama, M. Yasuda, and S. Takamiya, "Extended-Range High-Resolution FMCW Reflectometry by Means of Electronically Frequency-Multiplied Sampling Signal Generated from Auxiliary Interferometer," *Ieice Transactions on Electronics*, vol. E89C, no. 6, pp. 823-829, 2006.
[54] S. Jiang, B. Liu, and S. Wang, "A Dispersion Compensation Method Based on Resampling of Modulated Signal for FMCW Lidar," *Sensors*, vol. 21, no. 1, pp. 249, 2021.
[55] S. Vergnole, D. Lévesque, and G. Lamouche, "Experimental validation of an optimized signal processing method to handle non-linearity in swept-source optical coherence tomography," *Optics Express*, vol. 18, no. 10, pp. 10446-10461, 2010.
[56] Y. Dong, Z. Zhu, X. Tian, et al., "Frequency-Modulated Continuous-Wave LIDAR and 3D Imaging by Using Linear Frequency Modulation Based on Injection Locking," *Journal of Lightwave Technology*, vol. 39, no. 8, pp. 2275-2280, 2021.
[57]A. Singh, S. Choi, G. Wang, M. Daimari, B.G. Lee, "Analysis and fully memristor-based reservoir computing for temporal data classification," *Neural Networks*, vol. 182, art. 106925, 2025.
[58]A. Gaurav, X. Song, S. Manhas, A. Gilra, E. Vasilaki, P. Roy, M. M. De Souza, "Reservoir Computing for Temporal Data Classification Using a Dynamic Solid

Electrolyte ZnO Thin Film Transistor,” *Frontiers in Electronics*, vol. 3, art. 869013, 2022.
[59]Y. Yue, S. Liu, Y. Zhai and C. Wang, "Experimental Implementation of An All-Optical Reservoir Computer Using Photonic Time Stretch and Spectral Mixing," *2022 IEEE International Topical Meeting on Microwave Photonics (MWP)*, pp. 1-3, 2022,
[60]Y. Yue., S. Liu., W. Xu., C. Wang., “Experimental Investigation of Time Stretch Based Reservoir Computing with an Optical Input Mask,” *EPJ Web of Conferences*, vol. 287, art. 13010, 2023,
[61]Y. Yue, S. Liu, and C. Wang. "Reservoir computing assisted single-pixel high-throughput object classification." *Optical Sensing and Detection VIII*. Vol. 12999. SPIE, 2024.

List of Publication

[1] **T. Luan**, R. Zhang, Y. Yue, S. Liu, A. Zhang, C. Wang, “A Frequency Modulated Continuous Wave LiDAR System Based on Reservoir Computing,” 2023 IEEE International Topical Meeting on Microwave Photonics (MWP), Oct. 2023.

[2] Y. Yue, S. Liu, R. Zhang, **T. Luan**, A. Zhang, and C. Wang, “Experimental Investigation of an Optical Reservoir Computer with MZI-based Optical Mask,” 2023 IEEE International Topical Meeting on Microwave Photonics (MWP), Oct. 2023.