A Survey of Audio Processing Fault Tolerant Scalability Microservices Natural Language Processing Fault Detection and Distributed Systems

www.surveyx.cn

Abstract

The convergence of audio processing, fault tolerance, scalability, microservices, natural language processing (NLP), fault detection, and distributed systems represents a transformative frontier in contemporary computing. This survey delineates the intricate interdependencies and synergistic potential of these domains, foundational for the development of resilient and efficient computational systems. Audio processing, pivotal in speech technologies and music research, leverages meta-learning methodologies and neuromorphic computing for enhanced data analysis. Fault tolerance ensures system reliability amidst component failures, while microservices architectures facilitate scalability and efficient workload management. NLP complements these technologies by enabling advanced human-machine interactions, with multimodal systems improving emotion recognition accuracy. Fault detection mechanisms are critical for maintaining the integrity and performance of distributed systems. The survey highlights the necessity for robust frameworks that integrate these concepts, addressing challenges such as energy consumption, real-time processing, and system adaptability. Innovative approaches like DEEPSPECTRUMLITE and the Medusa framework exemplify advancements in scalability and fault tolerance. Future research directions include optimizing encryption processes, expanding datasets, and enhancing self-stabilizing properties in distributed systems. By examining these diverse yet interconnected motivations, the survey advances technology in audio processing, fault tolerance, scalability, and beyond, reflecting their relevance to current technological advancements.

1 Introduction

1.1 Scope and Significance

The convergence of audio processing, fault tolerance, scalability, microservices, natural language processing (NLP), fault detection, and distributed systems signifies a transformative frontier in contemporary computing. This survey delineates the intricate interdependencies and synergistic potential of these domains, foundational for developing resilient and efficient computational systems. Audio processing is pivotal for enhancing speech technologies and music research through sophisticated deep learning applications [1]. Meta-learning methodologies play a crucial role in integrating diverse approaches to improve audio data analysis [2]. The advancement of invariant speech recognition, essential for cognitive neuroscience and artificial intelligence, necessitates integrating audio processing with neural networks [3]. Additionally, configuring Spiking Neural Networks (SNNs) for temporal signal processing presents accessibility challenges for machine learning engineers [4]. High-performance computing in neuromorphic audio processing is set to revolutionize the audio processing landscape [5], while integrating audio processing with energy-efficient solutions for embedded devices addresses the limitations of deep neural networks [6].

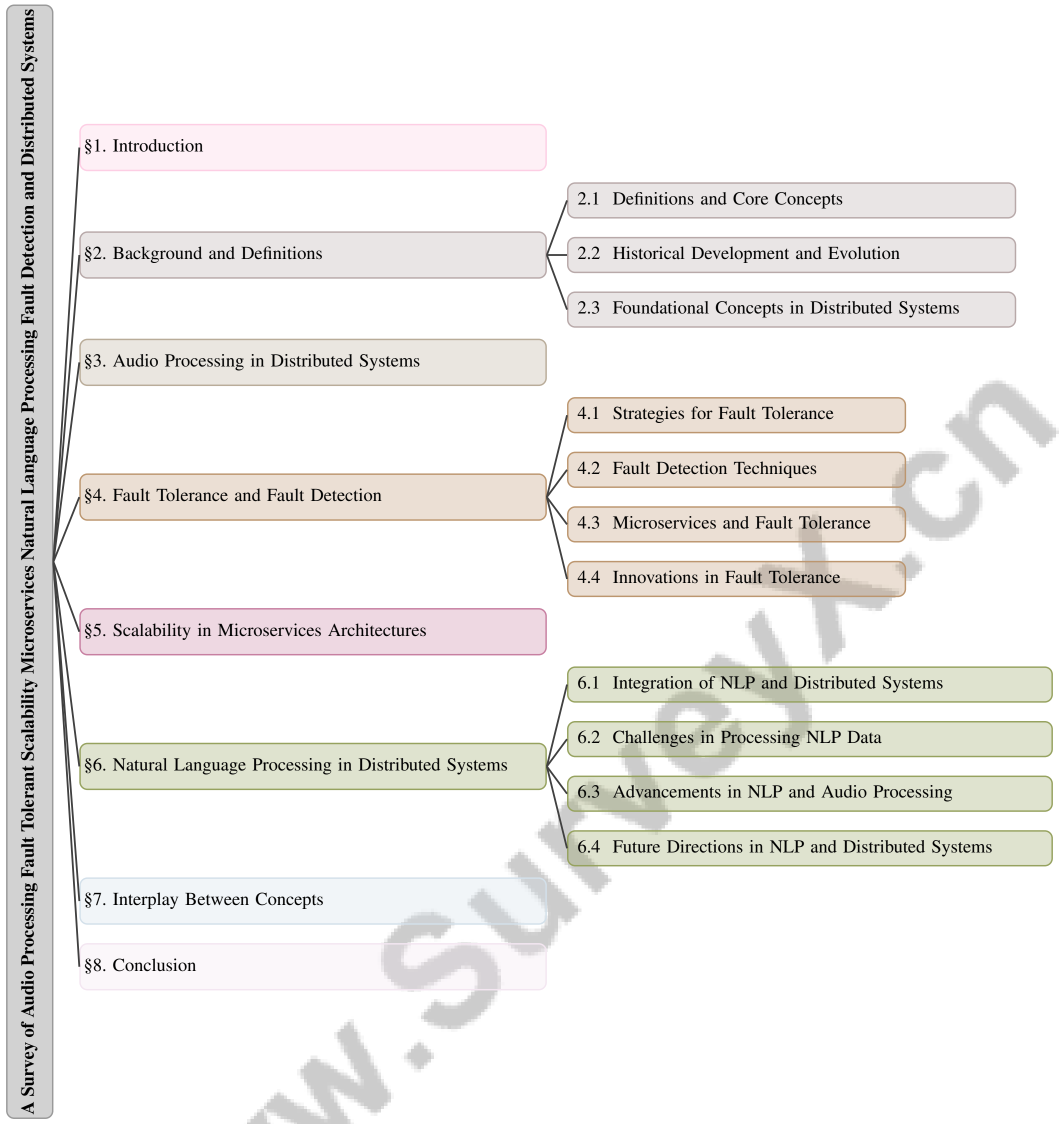
A Survey of Audio Processing Fault Tolerant Scalability Microservices Natural Language Processing Fault Detection and Distributed Systems
§1. Introduction
§2. Background and Definitions
2.1 Definitions and Core Concepts
2.2 Historical Development and Evolution
2.3 Foundational Concepts in Distributed Systems
§3. Audio Processing in Distributed Systems
§4. Fault Tolerance and Fault Detection
4.1 Strategies for Fault Tolerance
4.2 Fault Detection Techniques
4.3 Microservices and Fault Tolerance
4.4 Innovations in Fault Tolerance
§5. Scalability in Microservices Architectures
§6. Natural Language Processing in Distributed Systems
6.1 Integration of NLP and Distributed Systems
6.2 Challenges in Processing NLP Data
6.3 Advancements in NLP and Audio Processing
6.4 Future Directions in NLP and Distributed Systems
§7. Interplay Between Concepts
§8. Conclusion

Figure 1: chapter structure

Fault tolerance is critical for ensuring system reliability amidst component failures. Simplifying programming for reliable cloud services and integrating fault tolerance with scalability in distributed systems is emphasized in [7]. Deriving formal specifications for dependable systems enhances fault tolerance [8]. Scalability, particularly in microservices architectures, enables efficient management of increased workloads and data traffic [9].

NLP enhances these technologies by enabling advanced human-machine interactions. The necessity for a multi-modal system that integrates audio and video data to improve emotion recognition accuracy under challenging acoustic conditions is highlighted in [10]. Fault detection mechanisms are essential for identifying and diagnosing errors, maintaining the integrity and performance of distributed systems.

Integrating microservices architecture patterns with efficient media query processing is vital for advancing computing research and development. This integration fosters innovations that tackle contemporary technological challenges while enhancing scalability and maintainability. Utilizing a curated dataset of 20 open-source microservices projects facilitates a deeper understanding of architectural dependencies and patterns, while developing analytic frameworks for optimizing device energy consumption and cloud billing in IoT applications addresses critical resource management issues [11, 12]. This survey considers the broader implications of these integrations, emphasizing their significance in modern technological advancements.

1.2 Motivation and Relevance

This survey is motivated by the pressing need to address challenges and opportunities within advanced computing technologies. A significant impetus is the constrained flexibility of audio models, which require labeled data for training and are limited to predefined categories, as addressed by CLAP [13]. The exploration of meta-learning approaches in audio and speech processing aims to fill the gap where extensive surveys are lacking, providing insights into methodologies and future research directions [2].

In neuromorphic systems, inefficiencies in training due to reliance on external software lead to increased energy consumption and time delays, particularly in real-time audio processing applications [5]. This survey addresses these inefficiencies, contributing to advancements in neuromorphic audio processing. Furthermore, the need for robust, user-friendly libraries in audio processing frameworks is underscored by advancements in Torchaudio, facilitating research and development in this area [14].

The survey investigates the reliability of audio similarity metrics as proxies for audio quality, focusing on discrepancies between these metrics and human perception, which is crucial for enhancing audio processing systems [15]. In fault tolerance and distributed systems, challenges in programming reliable cloud services resilient to failures and network issues are explored [7], complemented by structured methods for specifying fault-tolerant systems functioning in real-world environments [16].

The necessity for lightweight frameworks, such as DEEPSPECTRUMLITE, enabling effective audio processing on embedded devices without extensive data uploads, aligns with current technological advancements [6]. Additionally, the demand for interpretable AI systems that can be trusted in critical applications is a significant motivation, addressing the need for effectively non-blocking consensus procedures [17].

By exploring interconnected motivations behind audio signal processing, this survey aims to enhance technological advancements in audio processing algorithms, fault tolerance, and scalability. It emphasizes the critical role of feature extraction techniques across various domains—temporal, frequency, cepstral, wavelet, and time-frequency—in improving machine learning performance in audio applications. Furthermore, the significance of meta-learning approaches in optimizing model performance with limited annotated data reflects their relevance to contemporary developments in audio technology [2, 18].

1.3 Structure of the Survey

This survey is meticulously structured to provide a comprehensive examination of the integration of audio processing, fault tolerance, scalability, microservices, NLP, fault detection, and distributed systems. The paper begins with an **Introduction**, articulating the scope, significance, motivation, and relevance of the survey, setting the stage for the subsequent detailed exploration of each concept.

The second section, **Background and Definitions**, offers foundational understanding of core concepts, including their historical development and evolution, crucial for readers unfamiliar with these technologies.

In the third section, **Audio Processing in Distributed Systems**, the role of audio processing within distributed systems is explored, detailing challenges, innovative techniques, and real-world applications.

The fourth section, **Fault Tolerance and Fault Detection**, examines strategies and technologies employed to achieve fault tolerance and detect faults within distributed systems, focusing on their application in audio processing and microservices architectures.

The fifth section, **Scalability in Microservices Architectures**, discusses scalability, particularly in microservices, providing examples of scalable audio processing and NLP applications.

In the sixth section, **Natural Language Processing in Distributed Systems**, the integration of NLP with distributed systems is analyzed, highlighting challenges, advancements, and future directions.

The seventh section, **Interplay Between Concepts**, analyzes how these technologies complement and influence each other, supported by case studies and practical implementations.

In the **Conclusion**, the study encapsulates essential findings related to audio signal feature extraction, evaluates the current landscape of research in audio processing and machine learning integration, and outlines prospective future research avenues and unresolved challenges, particularly in enhancing the efficacy of feature extraction techniques for applications in fields such as healthcare and speech recognition [19, 18]. This structured approach ensures a coherent narrative guiding the reader through the complexities of modern computing technologies.The following sections are organized as shown in Figure 1.

2 Background and Definitions

2.1 Definitions and Core Concepts

The integration of audio processing, fault tolerance, scalability, microservices, NLP, fault detection, and distributed systems encompasses diverse computing paradigms. This section defines core concepts essential for understanding the survey's scope.

Audio processing involves analyzing and manipulating sound signals, pivotal in applications like speech recognition and music information retrieval. It extends to chatbot systems managing audio queries, thus enhancing interactive platforms [20]. Advanced applications leverage deep learning for speech and audio processing, increasingly integrated with embedded devices to boost energy efficiency [6]. Bioacoustic applications face challenges due to the scarcity of labeled data for training DNN models, necessitating innovative feature extraction and training approaches [21].

Fault tolerance ensures system functionality amid component failures, critical in distributed systems where consensus among processes is vital. The Byzantine fault-tolerance problem is significant in distributed multi-agent machine learning, where agents may share incorrect stochastic gradients [22]. The Reliable State Machines (RSMs) framework is a programming model for developing fault-tolerant cloud services [7]. The absence of structured methods for deriving formal specifications in software systems contributes to misunderstandings and inefficiencies, underscoring the need for formal methods to achieve fault tolerance [8].

Scalability refers to a system's capacity to manage increased workloads or growth, a fundamental trait of microservices architectures defined by their loosely coupled nature. This architecture facilitates scalable system designs, enabling independent deployment and scaling of services [9]. Understanding architectural patterns and inter-service dependencies is crucial for analyzing microservices-based systems.

NLP enables computers to comprehend and interact with human language, often integrating with audio processing to enhance applications like multi-modal emotion recognition, merging audio and video modalities to improve accuracy [10].

Fault detection identifies and diagnoses errors to uphold distributed systems' integrity and performance. Distributed fault-tolerant algorithms are crucial in wireless sensor networks, where middleware manages communication and coordination [23]. Ensuring consistent access to shared objects is complicated by the lack of support for cooperative concurrency [17].

Distributed systems consist of components across networked computers that communicate and coordinate actions via message passing. Achieving distributed agreement in large-scale systems necessitates efficient resource utilization and recovery methods, such as checkpointing, to enhance reliability [24]. The topological properties of distributed systems are crucial for understanding their operation, with combinatorial topology and distributed algorithms playing a significant role [25]. Incorporating runtime adaptation into session types is critical for managing disciplined communication while allowing dynamic reconfiguration of processes [26].

Defining these key terms establishes a comprehensive foundation for understanding the interplay among these advanced computing paradigms.

AI-generated, for reference only.

2.2 Historical Development and Evolution

The historical trajectory of audio processing, fault tolerance, scalability, microservices, NLP, fault detection, and distributed systems reflects technological advancements and evolving challenges. Audio processing has evolved from traditional signal processing to machine learning-based approaches, addressing the complexity of audio data and the scarcity of annotated datasets, necessitating models capable of generalizing across tasks and environments [2]. The shift from static feature extraction methods to dynamic approaches marks a significant advancement, overcoming earlier limitations [21].

In fault tolerance, the transition from basic redundancy techniques to sophisticated consensus algorithms has been pivotal. The Paxos algorithm introduced complexities that spurred the development of comprehensible alternatives like Raft, addressing challenges in constructing fault-tolerant, consistent distributed systems [27]. The consensus problem remains central in distributed computing, with complexity gaps persisting [28]. Byzantine faults, where agents disrupt processes arbitrarily, continue to drive new fault-tolerant strategies [22].

Scalability within microservices architectures is critical in contemporary software design. The shift towards microservices underscores the necessity for architectural patterns that support scalable and resilient designs, facilitating independent scaling and deployment of services [11]. This evolution enhances the capacity to handle increased workloads effectively.

NLP has transitioned from rule-based systems to sophisticated machine learning models, highlighting the limitations of earlier methods and the potential of AI-driven solutions for enhanced language fluency and comprehension [29].

The evolution of fault detection in distributed systems has advanced from basic error-checking mechanisms to complex algorithms capable of detecting multiple events. The historical focus on AI solutions or intricate image/video recognition underscores the necessity for simpler, cost-effective alternatives [23]. Additionally, the development of session types has been crucial for ensuring communication protocols, yet prior methods lacked a formal framework for reasoning about dynamic reconfiguration in adaptable systems [26].

The historical evolution of these concepts illustrates a continuous cycle of innovation driven by emerging challenges and new technological capabilities. The fragmentation and lack of standardization in existing benchmarks complicate performance comparisons and coherent research directions, indicating a need for more unified approaches [30].

2.3 Foundational Concepts in Distributed Systems

Distributed systems consist of interconnected components that communicate and coordinate actions through message passing, enabling task execution across multiple nodes. This paradigm is fundamental to modern computing, supporting applications requiring high availability, fault tolerance, and scalability. The architecture of distributed systems is characterized by decentralized resources and processes, enhancing system resilience and performance while facilitating fault tolerance through independent, modular services, exemplified in microservices architectures. This design effectively manages failures and load variations, allowing for the implementation of advanced consensus algorithms like Paxos and Raft, ensuring reliability and consistency across components [31, 27, 32, 11, 33].

A critical aspect is the consensus mechanism, guaranteeing that all nodes agree on a common state, even amidst faults. Traditional consensus algorithms, such as Paxos and Raft, have been instrumental in achieving fault tolerance and consistency but often encounter performance and complexity challenges. The CAANS framework exemplifies hardware-accelerated solutions, utilizing programmable network devices to enhance consensus performance, surpassing traditional software implementations [34].

The reliability of distributed systems is bolstered through fault-tolerant designs employing strategies like redundancy and replication. These include "artificial redundancy," minimizing costs while maintaining diversity, and partial replication techniques enabling swift recovery by substituting failed processes with replicas. Algorithms for consensus and other primitives are optimized for challenging conditions, ensuring robust performance [32, 35, 36, 37, 38]. These designs mitigate component failures' impact, ensuring continuous operation and data integrity. Middleware solutions manage communication and coordination among components, facilitating seamless integration and interaction within the system.

Scalability is essential, enabling systems to accommodate increasing workloads through dynamic resource allocation and independent scaling of components. The microservices architecture, decomposing applications into modular, independent services, exemplifies a scalable design adhering to distributed system principles, allowing effective maintenance and scalability through tailored patterns [11, 12, 31, 39].

The foundational concepts of distributed systems, including peer-to-peer frameworks and microservices architectures, provide a robust infrastructure for developing efficient, scalable, and resilient applications. These systems leverage adaptive service organization, dynamic group membership, and fault tolerance to optimize resource utilization and maintain high availability under varying load conditions. By integrating architectural patterns and technologies like JXTA and PKI, developers can create modular services enhancing inter-service communication and overall system reliability [11, 31]. These systems underpin the survey topics, offering insights into integrating audio processing, fault tolerance, scalability, microservices, NLP, fault detection, and distributed computing.

In the realm of distributed systems, the processing of audio data presents a unique set of challenges and opportunities. To elucidate these complexities, Figure 2 provides a comprehensive overview of the hierarchical structure inherent in audio processing. This figure categorizes the various challenges faced, including performance issues, algorithmic difficulties, and the intricacies of noise evaluation. Furthermore, it highlights innovative techniques that have emerged to address these challenges, such as efficiency enhancements, interpretability, filtering, and meta-learning. The figure also illustrates real-world applications, detailing the algorithms, frameworks, systems, techniques, optimization strategies, and tools that are instrumental in advancing audio processing technologies. By integrating this visual representation, we can better appreciate the multifaceted nature of audio processing in distributed systems and the innovative approaches being employed to overcome its inherent challenges.

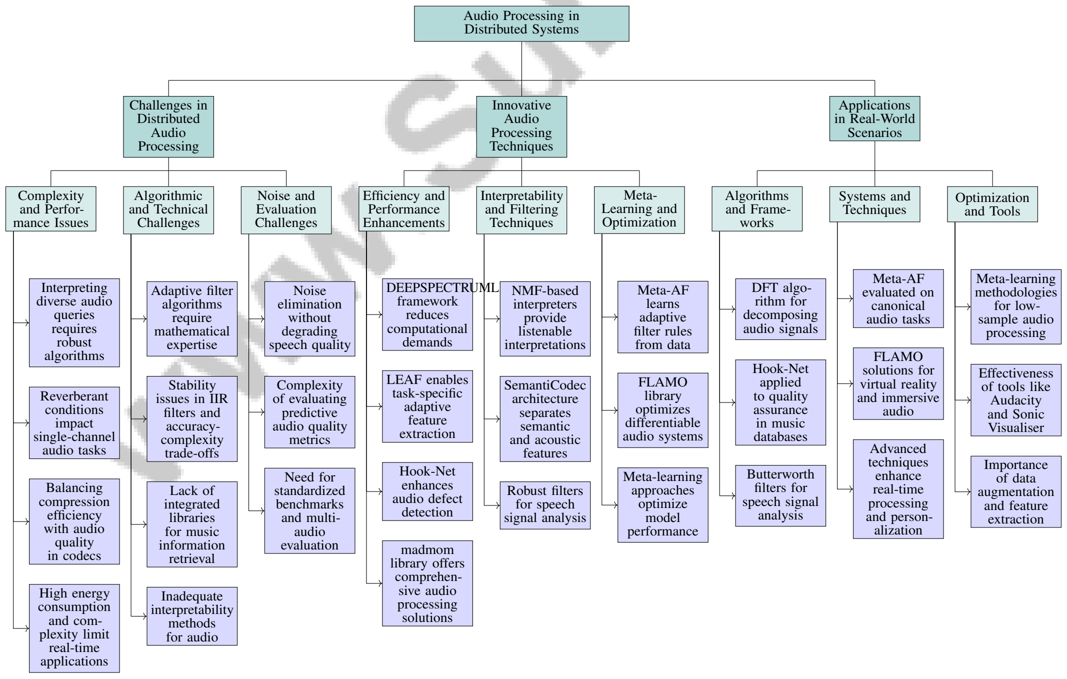
Audio Processing in Distributed Systems
Challenges in Distributed Audio Processing
Innovative Audio Processing Techniques
Applications in Real-World Scenarios
Complexity and Performance Issues
Algorithmic and Technical Challenges
Noise and Evaluation Challenges
Efficiency and Performance Enhancements
Interpretability and Filtering Techniques
Meta-Learning and Optimization
Algorithms and Frameworks
Systems and Techniques
Optimization and Tools
Interpreting diverse audio queries requires robust algorithms
Reverberant conditions impact single-channel audio tasks
Balancing compression efficiency with audio quality in codecs
High energy consumption and complexity limit real-time applications
Adaptive filter algorithms require mathematical expertise
Stability issues in IIR filters and accuracy-complexity trade-offs
Lack of integrated libraries for music information retrieval
Inadequate interpretability methods for audio
Noise elimination without degrading speech quality
Complexity of evaluating predictive audio quality metrics
Need for standardized benchmarks and multi-audio evaluation
DEEPSPECTRUMLITE framework reduces computational demands
LEAF enables task-specific adaptive feature extraction
Hook-Net enhances audio defect detection
madmom library offers comprehensive audio processing solutions
NMF-based interpreters provide listenable interpretations
SemantiCodec architecture separates semantic and acoustic features
Robust filters for speech signal analysis
Meta-AF learns adaptive filter rules from data
FLAMO library optimizes differentiable audio systems
Meta-learning approaches optimize model performance
DFT algorithm for decomposing audio signals
Hook-Net applied to quality assurance in music databases
Butterworth filters for speech signal analysis
Meta-AF evaluated on canonical audio tasks
FLAMO solutions for virtual reality and immersive audio
Advanced techniques enhance real-time processing and personalization
Meta-learning methodologies for low-sample audio processing
Effectiveness of tools like Audacity and Sonic Visualiser
Importance of data augmentation and feature extraction

Figure 2: This figure illustrates the hierarchical structure of audio processing in distributed systems, categorizing the challenges, innovative techniques, and real-world applications. It highlights the complexity and performance issues, algorithmic challenges, and noise evaluation difficulties faced in distributed audio processing. Innovative techniques such as efficiency enhancements, interpretability, filtering, and meta-learning are detailed. Additionally, the figure outlines applications in real-world scenarios, showcasing algorithms, frameworks, systems, techniques, optimization, and tools used to advance audio processing technologies.

3 Audio Processing in Distributed Systems

3.1 Challenges in Distributed Audio Processing

Distributed audio processing faces significant challenges due to the complexity of audio signals and distributed system architectures. A key issue is effectively interpreting and processing diverse audio queries, which demands robust algorithms [20]. Reverberant conditions degrade audio signals, impacting systems that rely on single-channel audio for tasks like emotion recognition [10]. Current audio codecs often fail to balance compression efficiency with audio quality, crucial for distributed systems [40]. High energy consumption and complexity further limit real-time applications on resource-constrained devices, necessitating more efficient models [6].

As illustrated in Figure 3, the key challenges in distributed audio processing can be categorized into three main areas: audio signal complexity, system limitations, and evaluation and interpretation. Each category highlights specific issues such as diverse audio queries, codec efficiency, energy consumption, adaptive filters, noise elimination, and interpretability, emphasizing the need for innovative solutions in these areas. Adaptive filter algorithms, essential for processing in nonstationary environments, require intensive mathematical expertise and are time-consuming to develop [41]. Stability issues in IIR filters, coupled with accuracy-complexity trade-offs, often lead to vanishing gradients and high memory usage [42]. In music information retrieval, the lack of integrated libraries for low-level processing and high-level feature analysis hinders comprehensive solution development [43]. Existing interpretability methods are inadequate for audio, highlighting the need for listenable interpretations that improve user understanding [44].

Noise elimination without speech quality degradation remains a persistent challenge [45]. Evaluating predictive audio quality metrics across different implementations is complex, underscoring the need for standardized benchmarks [46]. Addressing these challenges is vital for advancing distributed audio processing, especially in multi-audio scenarios requiring efficient, simultaneous stream processing. Innovations such as multi-audio evaluation benchmarks and multi-audio large language models (MALLMs) highlight the need for improved efficiency and effectiveness. Techniques like meta-learning and synthetic data generation offer potential enhancements in system performance, aligning closer to human auditory perception and expanding real-world applicability [47, 2].

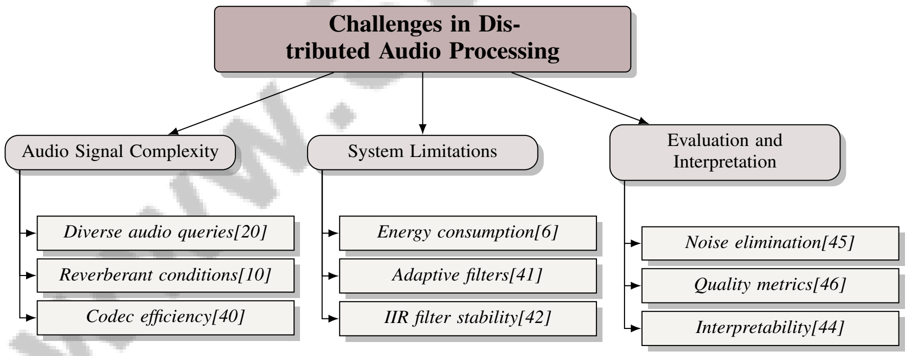

Figure 3: This figure illustrates the key challenges in distributed audio processing, categorized into audio signal complexity, system limitations, and evaluation and interpretation. Each category highlights specific issues such as diverse audio queries, codec efficiency, energy consumption, adaptive filters, noise elimination, and interpretability, emphasizing the need for innovative solutions in these areas.

3.2 Innovative Audio Processing Techniques

Recent advancements in distributed audio processing have introduced techniques that significantly enhance system efficiency, accuracy, and adaptability. The DEEPSPECTRUMLITE framework, for instance, reduces computational demands by generating Mel-spectrograms on-the-fly and using a lightweight multilayer perceptron (MLP) for classification [6]. LEAF processes audio through learnable layers, enabling task-specific adaptive feature extraction and improved performance in dynamic environments [48]. Hook-Net, a convolutional neural network, enhances audio defect

AI-generated, for reference only.

detection by focusing on context sensitivity [49]. The madmom library integrates low-level feature extraction with high-level analysis, offering a comprehensive audio processing solution [43].

In audio interpretability, non-negative matrix factorization (NMF) based interpreters provide listenable interpretations linked to high-level audio objects, enhancing manipulation within distributed systems [44]. The SemantiCodec architecture separates semantic and acoustic features, improving semantic richness and reconstruction quality [40]. Filters like Butterworth, Chebyshev Type I, and Elliptical are robust tools for speech signal analysis [45].

Meta-AF eliminates the need for exhaustive manual tuning by learning adaptive filter rules from data, enabling real-time operation across various tasks, which is particularly valuable in distributed audio processing [41]. The FLAMO library's ability to implement and optimize differentiable audio systems using frequency-sampling techniques enhances flexibility and performance [42]. These innovative techniques address challenges related to efficiency, accuracy, and adaptability, including meta-learning approaches that optimize model performance with minimal annotated samples and advanced digital audio processing tools for notated and non-notated music analysis. Comprehensive methodologies like data augmentation, feature extraction, and task selection strategies provide practical solutions for real-world audio processing [50, 2].

3.3 Applications in Real-World Scenarios

Distributed audio processing systems have broad applications across various real-world scenarios, showcasing their versatility in managing complex audio tasks. The Discrete Fourier Transform (DFT) algorithm is essential for decomposing audio signals into frequencies, facilitating analyses crucial for audio synthesis and noise reduction [51]. Hook-Net has been effectively applied to quality assurance in large music databases, detecting and localizing audio defects using the Free Music Archive dataset [49].

Butterworth filters, with their optimal attenuation-phase response balance, are suitable for speech signal analysis in telecommunications and audio enhancement [45]. Meta-AF has been evaluated across canonical audio tasks like system identification and acoustic echo cancellation, showing superior adaptability and performance over traditional baselines [41]. Frameworks like FLAMO offer open-source solutions for implementing differentiable audio systems, with case studies demonstrating improvements in response smoothness and optimization effectiveness, crucial for virtual reality and immersive audio experiences [42].

The application of advanced audio processing techniques within distributed systems enhances real-time processing, user personalization, and system adaptability. These techniques enable detailed information extraction from diverse audio sources, facilitating the analysis of non-notated music and performance data. Meta-learning methodologies optimize model performance with minimal annotated samples, addressing low-sample audio processing challenges. This comprehensive approach emphasizes the effectiveness of tools like Audacity and Sonic Visualiser, highlighting the importance of data augmentation, feature extraction, and task selection strategies in advancing audio processing technologies [50, 2].

4 Fault Tolerance and Fault Detection

In distributed systems, fault tolerance is crucial for ensuring reliability and continuous operation amidst challenges such as hardware failures, network disruptions, and malicious attacks. Table 5 offers a detailed classification of fault tolerance strategies, fault detection techniques, and microservices approaches, illustrating the diverse methodologies employed to bolster the robustness of distributed systems. This section explores strategies to enhance fault tolerance, focusing on methodologies and frameworks essential for robust distributed systems.

4.1 Strategies for Fault Tolerance

Fault tolerance is critical for maintaining reliability in distributed systems, particularly against failures and malicious activities. Replication techniques are foundational, providing data redundancy across multiple nodes to enhance availability and resilience against node failures [7]. The HPCNeuroNet

Category	Feature	Method
Strategies for Fault Tolerance	Efficiency and Resource Optimization	HPCN[5]
	System Reliability and Management	RSM[7], DFTM[23]
	Fault Detection and Mitigation	ALCA[52]
	Replication and Redundancy Strategies	5MR[24]
Fault Detection Techniques	Advanced Techniques	MDR[53], PSCA[35], DDTSE[54], SIA[55], PC[28], VF[56]
Microservices and Fault Tolerance	Distributed System Resilience	FTC[57], CLFT[58], HP[59], HT[32], PBR[60]
	State and Recovery Techniques	WRK[61], RR[62], FT-GAIA[63], N/A[31]
	Modular Deployment Strategies	J2K[64]
Innovations in Fault Tolerance	Interpretability and Testing	L2I[44], NTRX[65]
	Predictive and Reasoning Strategies	LFTS[8], RBM[66]
	Redundancy and Consensus	AFT[36], BFT[67], FLAQR[68]
	Resource Efficiency	CGE[22]

Table 1: This table provides a comprehensive overview of various strategies and techniques for enhancing fault tolerance in distributed systems. It categorizes methods into four main areas: strategies for fault tolerance, fault detection techniques, microservices and fault tolerance, and innovations in fault tolerance, highlighting specific features and methods employed within each category. The table serves as a reference for understanding the diverse approaches and innovations that contribute to the robustness and reliability of distributed systems.

architecture exemplifies energy-efficient designs suitable for real-time processing in distributed audio environments [5].

Addressing Byzantine failures, where components behave arbitrarily, is challenging. The Comparative Gradient Elimination (CGE) method mitigates incorrect stochastic gradients by filtering out those with the largest Euclidean norms in distributed optimization [22]. The ParameterizedConsensus algorithm enhances communication and time complexity, improving consensus efficiency [28]. The FLAQR framework introduces Flow-Limited Authorization for Quorum Replication, offering high-level abstractions for consensus and replication management while enforcing security policies [68].

Automated fault detection in audio processing differentiates between intentional effects and defects. The ALCA method improves generalization and classification accuracy while reducing power consumption [52]. The Listen to Interpret (L2I) method uses non-negative matrix factorization (NMF) for accurate interpretations of high-level audio objects [44].

Cloud environments require robust fault-tolerant designs to manage crash faults and limitations of frameworks like MapReduce, which struggle with arbitrary faults or malicious attacks [53]. Reliable State Machines (RSMs) provide structured state management and automatic failure handling, enhancing fault tolerance [7]. Middleware solutions, such as those proposed by [23], effectively monitor crowd density using low-cost components, achieving high reliability without resource-intensive methods.

In edge computing, efficient fault-tolerance mechanisms prevent resource contention and system overload under non-stationary workloads. The method proposed by [24], employing 5-modular redundancy and a voting mechanism, exemplifies robust reliability strategies in such settings.

The EFTOS Voting Farm addresses fault tolerance in distributed software systems, highlighting drawbacks of N-modular redundancy (NMR), which can create single points of failure [56]. The Layered Fault Tolerant Specification (LFTS) method enhances fault tolerance by distinguishing normal from abnormal behaviors through layered abstraction [8].

These strategies highlight the necessity for adaptive and resilient fault tolerance mechanisms tailored to the unique challenges of distributed systems. Leveraging concepts like artificial redundancy and component-based middleware ensures reliable service delivery amid diverse and dynamic conditions, enhancing system dependability and minimizing the impact of evolving fault models [36, 60, 32].

4.2 Fault Detection Techniques

Fault detection is essential in distributed systems to maintain reliability and ensure rapid recovery in high-availability environments. Traditional consensus solutions often falter with flaky channels that lose messages, necessitating advanced fault detection mechanisms capable of functioning under unreliable communication conditions [35]. In cloud environments, Medusa efficiently schedules job replicas, minimizing replication costs and significantly improving performance [53].

Method Name	Communication Reliability	System Resilience	Operational Context
PSCA[35]	Flaky Channels	Process Crashes	Distributed Systems
MDR[53]	Flaky Channels	Advanced Fault Detection	Multi-cloud Environments
DDTSE[54]	Fault-tolerant Mechanism	Fault Tolerance	Wireless Sensor Networks
SIA[55]	Asynchronous System	Byzantine Fault-tolerant	Point-to-point Communication
CGE[22]	-	Advanced Fault Detection	Google Cloud Platform
PC[28]	Flaky Channels	Malicious Attacks	Distributed Systems
RBM[66]	-	System Resilience	Industrial Systems
VF[56]	Flaky Channels	Advanced Fault Detection	Message Passing Environments
RR[62]	Communication Breakdown	Fault Tolerance Capabilities	Robotic Applications

Table 2: Overview of fault detection methods in distributed systems, highlighting their communication reliability, system resilience, and operational context. The table compares various methods such as PSCA, MDR, and DDTSE, providing insights into their applicability across different environments, including multi-cloud and wireless sensor networks.

Wireless sensor networks face challenges in detecting simultaneous events across distinct regions, requiring robust distributed detection techniques that maintain accuracy [54]. The shortcomings of existing causal ordering algorithms under Byzantine faults complicate fault detection, as ensuring causal message delivery is challenging when processes act maliciously [55].

The CGE method ensures fault tolerance by enabling non-faulty agents to compute optimal learning parameters despite the presence of faulty agents [22], complemented by the ParameterizedConsensus algorithm that balances communication and time complexity [28].

Industrial computing systems face operational risks due to increased critical failure rates toward the end of their useful lifetimes. Effective fault detection mechanisms, such as Reliability Based Monitoring (RBM), utilize hierarchical reliability models for predictive diagnostics and maintenance scheduling [66]. The EFTOS Voting Farm emphasizes the limitations of traditional fault tolerance methods, which can lead to single points of failure [56].

In distributed optimization, maintaining accurate information about faulty agents' cost functions is challenging, necessitating advanced fault detection techniques that can operate with incomplete or inaccurate data [69]. The loss of metadata about nodes in distributed systems hampers recovery efforts after master failures, highlighting the need for improved fault detection and recovery protocols [62].

To effectively tackle challenges faced by distributed systems, it is imperative to develop advanced fault detection techniques that ensure reliability and performance across various contexts, particularly given the complexities of non-malicious arbitrary faults. Leveraging concepts like artificial redundancy and active replication can enhance fault tolerance while minimizing resource overhead, thereby improving system resilience against benign faults and potential communication issues or malicious attacks [69, 36, 32].

Figure 4 illustrates the categorization of fault detection techniques across different domains, highlighting key methods and algorithms in distributed systems, wireless sensor networks, and industrial systems. Each category is associated with specific approaches that address the unique challenges of fault tolerance in their respective environments. Additionally, Table 2 presents a comprehensive comparison of fault detection methods, detailing their communication reliability, system resilience, and operational contexts, which are crucial for understanding their effectiveness in maintaining reliability in distributed systems.

4.3 Microservices and Fault Tolerance

Microservices architectures significantly enhance fault tolerance in distributed systems by promoting modularity and resilience through loosely coupled service designs. This architecture allows for independent deployment, scaling, and updating of services, crucial for maintaining robustness amid failures. The Workrs framework exemplifies this by providing fault tolerance and scalability in computation offloading, ensuring job completion despite faults [61]. By distributing fault management across multiple layers, microservices architectures enhance overall system reliability [58].

Table 3 presents a comprehensive comparison of different methods utilized to enhance fault tolerance in microservices architectures, detailing their architectural designs, fault management techniques, and adaptability and scalability features. The integration of fault-tolerant techniques in microservices

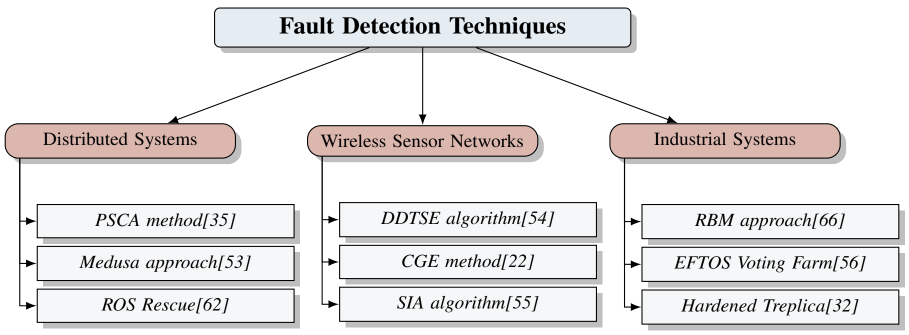

Figure 4: This figure illustrates the categorization of fault detection techniques across different domains, highlighting key methods and algorithms in distributed systems, wireless sensor networks, and industrial systems. Each category is associated with specific approaches that address the unique challenges of fault tolerance in their respective environments.

Method Name	Architectural Design	Fault Management Techniques	Adaptability and Scalability
WRK[61]	Docker Checkpointing Integration	Fault Tolerant Orchestration	Fault Tolerant Service
CLFT[58]	-	Cross-layer Techniques	-
FTC[57]	Microservices Architecture	State Replication	Vertical Scaling
J2K[64]	Distributed Kubernetes Deployment	Comprehensive Error Handling	Dynamic Pod Deployment
HT[32]	-	Distributed Validation Mechanisms	-
HP[59]	-	Consistency Validation Techniques	-
PBR[60]	Component-based Middleware	Primary-Backup Replication	Dynamic Reconfiguration
RR[62]	-	State Replication	-
FT-GAIA[63]	Functional Replication	Functional Replication	Not Applicable
N/A[31]	Peer Group Structure	Adaptive Redundancy	Dynamic Role Interchange

Table 3: Comparison of various methods for implementing fault tolerance in microservices architectures, highlighting their architectural design, fault management techniques, and adaptability and scalability. The table provides an overview of different approaches, such as Docker checkpointing integration, microservices architecture, and distributed Kubernetes deployment, and their corresponding fault management strategies.

is further illustrated by the FTC method, which piggybacks state updates onto packets, allowing middleboxes to replicate without dedicated servers, thus enhancing fault tolerance with minimal infrastructure overhead [57]. Deploying microservices in containerized environments like Kubernetes facilitates dynamic resource allocation, further improving resilience. The Jup2Kub framework exemplifies this by converting Jupyter notebooks into self-contained units within Kubernetes pods, enhancing modularity and scalability in scientific workflows [64].

Fault tolerance in microservices is reinforced through decentralized validation techniques that ensure replicas do not propagate erroneous states. By employing fault detection and distributed validation, systems can correct non-malicious arbitrary faults, maintaining consistency and reliability [32]. The hardening of consensus algorithms, such as Paxos, through consistency validation techniques further illustrates the integration of fault-tolerant strategies within microservices environments [59].

Additionally, the ability to dynamically manipulate software architecture allows microservices to transition between fault tolerance mechanisms without system downtime, ensuring continuous operation [60]. For example, the ROS Rescue mechanism logs the metadata of the ROS master to persistent storage, enabling recovery from failures and enhancing fault tolerance in robotic systems [62]. Similarly, FT-GAIA, a fault-tolerant extension of the GAIA/ART‘IS middleware, replicates simulation entities to tolerate failures, showcasing the adaptability of microservices across various contexts [63].

Game theory models interactions between fault-tolerant implementations and nominal models, providing a framework for assessing relative fault tolerance, as demonstrated by methods leveraging artificial redundancy. The CONSCIENTIA framework utilizes peer services to enhance service availability through adaptive redundancy and self-organization, further illustrating microservices' role in maintaining continuous service operation [31].

AI-generated, for reference only.

Method Name	Fault Tolerance Strategies	Technological Innovations	Application Domains
AFT[36]	Artificial Redundancy	Artificial Redundancy Approach	Web Services
L2I[44]	Not Mentioned	Spectral Pattern Dictionary	Audio Processing
BFT[67]	Consensus Protocols	Serverlessbft Protocol	Edge Applications
RBM[66]	Hierarchical Reliability Model	Predictive Maintenance Strategy	Complex Systems
LFTS[8]	Layered Specifications	Rely/guarantee Reasoning	Transportation Automotive Systems
FLAQR[68]	Quorum Replication Protocols	Flow-Limited Authorization	Distributed Systems
CGE[22]	Comparative Gradient Elimination	Adaptive Threshold	Distributed Learning
NTRX[65]	Netrix Framework	Randomized Exploration Engine	Distributed Systems

Table 4: Overview of recent advancements in fault tolerance methods, highlighting their respective fault tolerance strategies, technological innovations, and application domains. The table provides a comparative analysis of various methodologies, emphasizing their contributions to enhancing resilience and efficiency in distributed systems.

4.4 Innovations in Fault Tolerance

Recent advancements in fault tolerance technologies have introduced innovative solutions that enhance the resilience and efficiency of distributed systems. Table 4 presents a comprehensive comparison of recent innovations in fault tolerance technologies, detailing their strategies, technological advancements, and application domains to illustrate the progress and impact on distributed systems. The artificial fault tolerance (AFT) approach leverages universal redundancy to minimize costs associated with traditional fault tolerance methods. By incorporating diverse components, AFT enhances reliability while reducing replication expenses, offering an economical strategy for maintaining system integrity [36].

In audio processing, the Listen to Interpret (L2I) system provides high-fidelity interpretations that effectively bridge classifier decisions with high-level audio objects, enhancing interpretability and robustness in audio-related fault tolerance solutions [44].

The SERVERLESSBFT framework represents a significant advancement in managing transactional flows in a Byzantine fault-tolerant manner, addressing challenges faced by existing edge computing solutions [67]. The Runtime-Based Monitoring (RBM) approach enhances reliability through predictive maintenance strategies that leverage real-time diagnostics data, improving system health management and reducing failure likelihood [66].

Integrating Layered Fault Tolerant Specification (LFTS) with rely/guarantee reasoning introduces a structured methodology for enhancing fault tolerance and system reliability, providing a formal mechanism for managing system behaviors under fault conditions [8]. The FLAQR framework exemplifies the extension of the Flow Limited Authorization Model with availability policies, enhancing fault tolerance capabilities by ensuring secure and reliable consensus [68].

The CGE method offers a computationally simpler alternative for achieving fault tolerance in distributed learning environments, particularly against Byzantine faults, making it a competitive choice for distributed learning applications [22]. The Netrix framework facilitates guided and effective testing of distributed protocol implementations, significantly contributing to the robustness of fault tolerance strategies [65].

These innovations reflect the ongoing evolution of fault tolerance technologies, addressing the complex challenges posed by modern distributed systems. They pave the way for more resilient, efficient, and scalable computing environments, with potential applications across diverse domains. Future research should focus on optimizing and applying these advanced audio processing models, particularly in enhancing performance and usability in real-world scenarios. Investigations into specialized models like Automatic Speech Recognition, Speaker Diarization, and Music Identification, along with improved pre-processing techniques, will be crucial for simplifying tasks and reducing computational demands, thereby making these technologies more accessible for deployment in automated voice assistant applications. Additionally, exploring different signal decomposition methods, such as Fourier and Wavelet transforms, will inform best practices for model training and evaluation in audio processing tasks [20, 70].

Feature	Strategies for Fault Tolerance	Fault Detection Techniques	Microservices and Fault Tolerance
Fault Tolerance Strategy	Replication Techniques	Advanced Detection Mechanisms	Modular Service Design
Technological Advancement	Hpcneuronet Architecture	Medusa Scheduling	Workrs Framework
Application Domain	Distributed Audio	Cloud Environments	Microservices Architectures

Table 5: This table provides a comprehensive comparison of various strategies and technological advancements in fault tolerance, fault detection, and microservices architectures. It highlights the methodologies applied to enhance the resilience and reliability of distributed systems across different application domains, including distributed audio, cloud environments, and microservices architectures.

5 Scalability in Microservices Architectures

5.1 Conceptual Foundations of Scalability in Microservices

Scalability is a core attribute of microservices architectures, enabling them to efficiently handle increased workloads through modular design. This scalability is achieved by optimizing resource utilization and enhancing resilience via independent services, ensuring fault tolerance and availability. Architectural patterns are crucial for scaling to meet performance constraints, focusing on capacity planning and fault tolerance strategies that reduce resource replication costs while maintaining reliability [11, 37, 57]. Distributed detection methods enhance scalability by processing local data and aggregating results from neighboring components, reducing reliance on centralized processing and improving responsiveness [54].

The HPCNeuroNet architecture exemplifies scalability advancements through neuromorphic computing, optimizing microservices for resource-intensive applications [5]. Meanwhile, the Reliable State Machines (RSM) framework simplifies programming and enhances performance, facilitating scalable application development [7]. In edge computing, the DeepFT model predicts system states to optimize task scheduling and migration, enhancing scalability [71]. The Medusa framework supports scalable execution in multi-cloud setups by facilitating MapReduce jobs without modifying the framework [53]. Additionally, Runtime-Based Monitoring (RBM) introduces a hierarchical model for dynamic maintenance scheduling, improving system availability [66].

The CONSCIENTIA framework organizes services into peer groups, allowing dynamic role interchange based on network conditions, enhancing resource utilization and service continuity [31]. These foundational concepts underscore scalability's critical role in microservices, enabling effective management of diverse computing demands. Recent studies highlight the importance of tailored architectural patterns and curated datasets to understand inter-service dependencies and scalability in practical applications [11, 31].

Figure 5 illustrates the key components and frameworks that contribute to scalability in microservices, categorizing them into architectural patterns, frameworks and models, and scalable execution strategies. This visual representation reinforces the discussion on scalability by providing a structured overview of the elements that play a pivotal role in enhancing the performance and adaptability of microservices architectures.

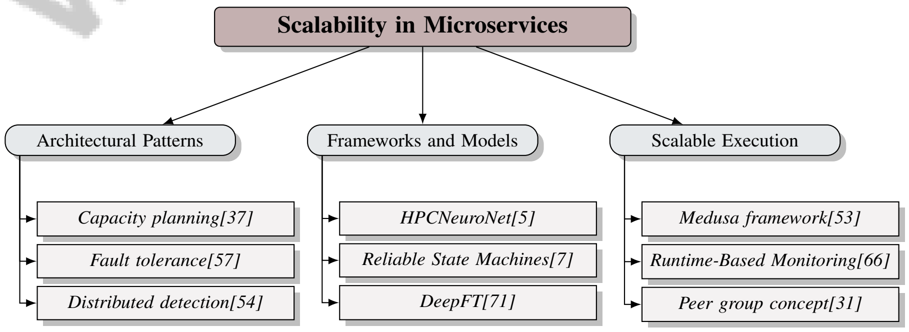

Figure 5: This figure illustrates the key components and frameworks that contribute to scalability in microservices, categorizing them into architectural patterns, frameworks and models, and scalable execution strategies.

5.2 Scalable Audio Processing in Microservices

Scalable audio processing within microservices significantly improves handling complex audio tasks. DEEPSPECTRUMLITE exemplifies this by enabling on-device processing, minimizing data transfers, and enhancing efficiency in real-time applications [6]. The Rockpool library facilitates deploying Spiking Neural Network (SNN) architectures to neuromorphic hardware, optimizing temporal signal processing in scalable contexts [4]. A dataset of 370 audio files segmented into various classes supports effective audio recognition, showcasing scalable applications within microservices frameworks [72].

Innovations such as compressing diverse audio types into fewer than a hundred tokens per second while preserving quality are vital for efficient data transmission within microservices architectures [40]. These advancements demonstrate the capability to manage intricate audio tasks across domains, including meta-learning approaches that enhance model performance with minimal samples and multi-audio processing techniques in audio large language models (ALLMs) [50, 47, 2].

5.3 Scalable Natural Language Processing (NLP) in Microservices

The integration of scalable natural language processing (NLP) within microservices architectures enables robust applications for diverse linguistic tasks. The CLAP framework, evaluated across 16 downstream tasks in eight domains, exemplifies scalability and adaptability in various NLP contexts [13]. Reinforcement learning frameworks automate architecture search, enhancing scalability by discovering high-performing models with minimal human intervention, thus streamlining development and reducing computational overhead [73]. Evaluating models based on accuracy and F1 scores underscores the importance of robust metrics for assessing scalability and effectiveness in NLP applications [30].

Recent advancements in scalable NLP applications within microservices highlight their role in managing complex linguistic tasks efficiently. Leveraging large language models (LLMs) and integrating specialized audio processing systems optimize resource utilization across distributed networks, supporting applications from audio query handling to real-time inference and fine-tuning [20, 11, 74, 19, 39].

5.4 Innovations and Algorithms Enhancing Scalability

Recent innovations in microservices scalability introduce techniques and algorithms that significantly enhance system efficiency. The PETALS method adapts to device availability and network conditions, minimizing communication overhead and maximizing throughput [74]. The Adaptive Resource Allocation Framework (ARAF) employs machine learning to improve resource allocation decisions based on historical data, optimizing scalability by ensuring efficient workload management [75]. In fault tolerance, the Medusa framework replicates jobs and compares outputs, enhancing reliability without excessive resource expenditure [53].

The Sender-Inhibition Algorithm ensures reliable communication in microservices architectures by adhering to strict delivery rules based on message acknowledgments, reducing message loss [55]. The WCP protocol maintains near-constant communication workload per node while tolerating Byzantine failures, representing a significant advancement in scalability for distributed systems [76]. The DPDP method utilizes parallel processing for efficient handling of large datasets, enhancing real-time analytics performance [9]. Future research in audio processing should focus on robust meta-learning models adaptable to various conditions and advanced sampling techniques, further enhancing scalability within microservices frameworks [2].

Additionally, integrating Vedic multipliers and carry-save adders in FPGA implementations optimizes speed and efficiency in specialized applications, such as ECG signal denoising, highlighting the potential for hardware-based optimizations to improve scalability [24]. Advancements in microservices architectures, alongside innovative algorithms, emphasize scalability's essential role in efficiently managing increased workloads while maintaining performance and reliability across diverse computing environments. Curated datasets of microservices projects facilitate understanding of architectural patterns and inter-service dependencies, supporting ongoing research in the field [11, 53].

6 Natural Language Processing in Distributed Systems

6.1 Integration of NLP and Distributed Systems

Integrating natural language processing (NLP) with distributed systems marks a significant advancement in handling linguistic data within decentralized architectures. This integration leverages the scalability and fault tolerance of distributed systems to improve NLP application efficiency and reliability. As illustrated in Figure 6, the advancements in this field can be categorized into three main areas: enhancements in NLP efficiency, the development of distributed NLP applications, and directions for future research. This figure highlights significant contributions from various studies, emphasizing the multifaceted nature of the integration.

Deploying deep neural networks has notably enhanced speech recognition by decoding auditory representations, critical for increasing accuracy in distributed NLP contexts [3]. In distributed audio processing, Spiking Neural Networks (SNNs) via the Rockpool library demonstrate low-power applications, crucial for real-time audio tasks such as voice-activated chatbots [4, 20].

The Reliable State Machines (RSMs) framework ensures reliable NLP operations by enabling exactly-once processing, preserving input consistency across distributed nodes [7]. Platforms like Jup2Kub enhance deployment efficiency and fault tolerance in scientific workflows, facilitating seamless NLP task execution in distributed environments [64]. Integrating audio and video modalities enhances emotion recognition, showcasing the synergy between audio processing and NLP for interpreting emotional expressions [10].

In smart buildings, distributed audio processing techniques improve occupancy estimation, highlighting NLP's role in creating intelligent environments through distributed data utilization [77]. The integration of Deep Active Feature Learning (DAFL) into active learning frameworks augments audio data processing by enhancing adaptive feature extraction, boosting NLP capabilities [21]. The UNISTORE system exemplifies varying transaction consistency based on dependencies, ensuring reliable NLP process commitments [78].

Integrating NLP in distributed systems fosters robust, scalable applications capable of managing complex linguistic tasks. Future research should focus on improving automatic audio processing tool accuracy through deep learning to advance NLP capabilities within distributed architectures [50].

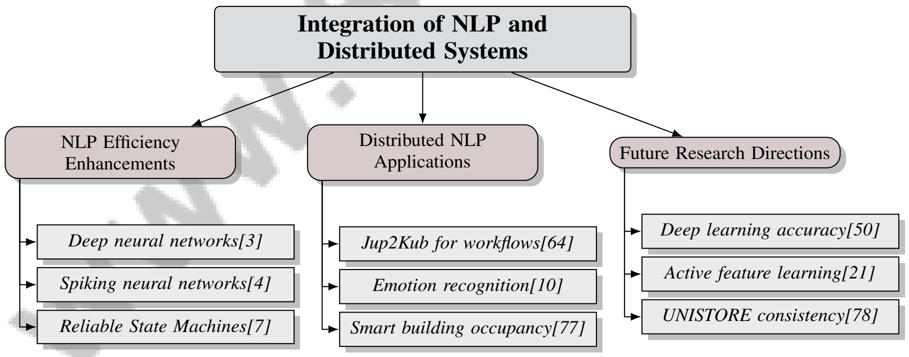
Integration of NLP and Distributed Systems
NLP Efficiency Enhancements
Distributed NLP Applications
Future Research Directions
Deep neural networks[3]
Spiking neural networks[4]
Reliable State Machines[7]
Jup2Kub for workflows[64]
Emotion recognition[10]
Smart building occupancy[77]
Deep learning accuracy[50]
Active feature learning[21]
UNISTORE consistency[78]

Figure 6: This figure illustrates the integration of NLP and distributed systems, categorizing advancements into NLP efficiency enhancements, distributed NLP applications, and future research directions, highlighting significant contributions from various studies.

6.2 Challenges in Processing NLP Data

Processing NLP data in distributed systems presents challenges due to linguistic input intricacies and system decentralization. Leader-based coordination inefficiencies can create bottlenecks, increasing latency and hindering large-scale NLP data processing [79]. Robust fault recovery mechanisms are essential, as traditional methods often require significant recomputation, delaying processing [80].

Data privacy is critical in distributed NLP systems, necessitating robust privacy-preserving techniques [81]. NLP systems' adaptability to real-world scenarios is often limited by performance in variable noise conditions, affecting tasks like voice activity detection [82]. Processing low-frequency sounds, such as voiceless consonants, poses challenges, potentially leading to speech recognition errors [83]. Dataset limitations, particularly those excluding relevant audio events, can impede NLP models' effectiveness [84].

Incorporating unlabeled data into NLP models can enhance performance in tasks like speaker recognition, but challenges arise regarding data quality and model reliability [85, 86]. Existing transaction models struggle with concurrent activity cooperation, complicating NLP data processing in distributed systems and diminishing application efficiency [87]. Complex acoustics or noise interference present significant challenges, impacting audio processing task accuracy [88].

Balancing integrity and availability in systems with heterogeneous trust relationships is challenging, as current methods often fail to effectively manage these trade-offs [68]. Developing advanced algorithms and models capable of adapting to diverse conditions is necessary for robust NLP data processing in distributed systems.

6.3 Advancements in NLP and Audio Processing

Recent advancements in NLP and audio processing have significantly improved systems' capabilities in interpreting and generating language and audio signals. Deep learning architectures, such as CNNs and LSTMs, have enhanced tasks like speech recognition, music retrieval, and sound detection by leveraging extensive datasets and sophisticated feature representations [1, 89, 70]. Non-negative Matrix Factorization (NMF)-based interpreters enhance audio classification network transparency and interpretability, improving user understanding of network decisions [44].

Innovations like SemantiCodec integrate semantic-rich audio representations, enhancing audio processing efficiency in distributed systems [40]. The Meta-AF framework improves adaptive filter performance, offering a robust alternative to traditional methods by learning adaptive rules directly from data [41]. The AQP framework provides a platform for evaluating audio quality metrics, ensuring high performance and reliability in audio processing systems [46].

These advancements highlight the dynamic interplay between NLP and audio processing, driving innovations that enhance systems' abilities to understand and generate language and audio signals. Future research should explore advanced deep learning techniques alongside domain-specific insights to improve NLP and audio processing applications, addressing challenges and unlocking cross-domain advancements [1, 20, 90].

6.4 Future Directions in NLP and Distributed Systems

The future of NLP in distributed systems involves significant advancements to enhance performance, scalability, and fault tolerance. Optimizing consensus protocols, like those in SERVERLESSBFT, is crucial for reducing latency and improving throughput in edge computing environments [67]. Integrating asynchronous models and failure scenarios into topological frameworks can improve NLP system robustness and adaptability [91].

Applying reinforcement learning algorithms to NLP tasks offers prospects for developing adaptable models capable of managing diverse linguistic tasks in distributed environments [73]. Enhancing libraries like Torchaudio with advanced models and exploring cross-modal applications could significantly improve NLP and audio processing systems [14].

Developing expressive transaction protocols in high-fault environments, informed by Byzantine fault-tolerant methods, is another critical area for exploration [92]. Enhancing frameworks like Workrs to improve APIs and implement security measures is essential for robust NLP applications [61]. Exploring resource allocation methods, such as ARAF, across domains could optimize resource utilization and ensure NLP application scalability [75].

Enhancing the Comparative Gradient Elimination (CGE) method for improved fault tolerance and investigating its applicability in heterogeneous data settings could provide valuable insights into NLP system robustness [22]. Addressing these future directions will advance NLP integration in distributed systems, leading to more resilient, efficient, and scalable applications.

7 Interplay Between Concepts

7.1 Case Studies and Practical Implementations

Exploring the interconnections between audio processing, fault tolerance, scalability, microservices, NLP, fault detection, and distributed systems reveals a complex web of interactions. Various case studies demonstrate how these elements integrate to enhance audio and speech processing. Meta-learning methodologies, for instance, optimize audio tasks with limited annotated samples, while Fourier and Wavelet transforms improve machine learning performance in voice recognition [70, 2]. The connected consensus problem, reducible to approximate agreement on graphs, underlines the role of distributed algorithms in achieving consensus, crucial for reliability and coordination in complex networks [93].

In human-robot interaction, real-time processing of ego speech exemplifies the integration of audio processing with fault detection to enhance interaction quality [83]. Distributed algorithms in failure-free networks provide foundational insights into the dynamics of distributed systems [91]. The ETM method in semi-supervised learning outperforms conventional loss functions, particularly in one-shot learning, highlighting the efficiency of advanced algorithms in distributed systems [85].

Algorithms for multicast scenarios, especially in Byzantine fault-tolerant causal ordering, illustrate the synergy between fault tolerance and scalability, ensuring robust mechanisms for dynamic groups [55]. Clustering processes into groups using local consensus and gossip protocols demonstrates effective communication complexity and fault tolerance integration, ensuring agreement among non-faulty processes [28]. The FLAQR framework, with its tailored security policies, underscores the critical role of security in consensus and replication, ensuring efficient operations [68].

As illustrated in Figure 7, the hierarchical categorization of key concepts in computing systems emphasizes the relationships among audio processing, distributed systems, and fault tolerance. Each category is supported by specific methodologies and case studies, demonstrating their integration and application in enhancing system performance and reliability. These case studies collectively underscore the intricate relationships among computing concepts, showcasing their synergistic integration to enhance robustness, efficiency, and scalability. Microservices-based architectures highlight the importance of architectural patterns in system maintenance and scaling, while peer-to-peer frameworks demonstrate the value of decentralized resource management for fault tolerance and availability. Distributed inference for large language models reveals efficient resource pooling across geodistributed devices, and comprehensive audio query systems enhance query resolution through specialized models. Media query processing in IoT contexts balances device energy consumption and cloud billing, offering a holistic view of optimizing modern computing infrastructures [31, 20, 11, 12, 74].

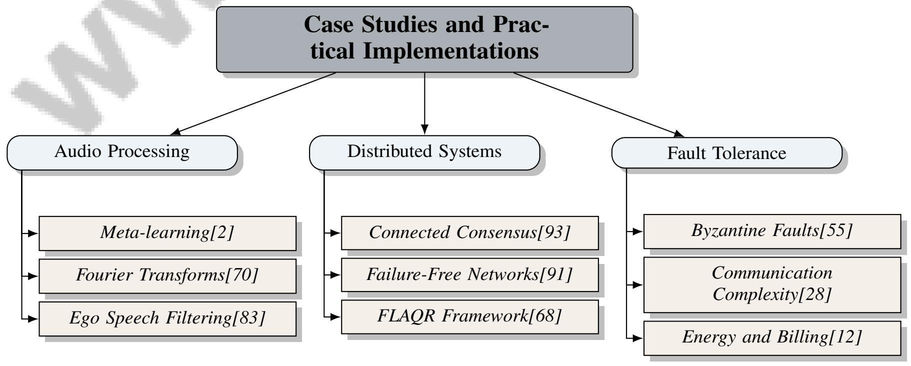

Figure 7: This figure illustrates the hierarchical categorization of key concepts in computing systems, focusing on audio processing, distributed systems, and fault tolerance. Each category is supported by specific methodologies and case studies, demonstrating their integration and application in enhancing system performance and reliability.

7.2 Energy Consumption and System Performance

The integration of audio processing, fault tolerance, scalability, microservices, NLP, fault detection, and distributed systems profoundly influences energy consumption and system performance. Integrating audio/visual recognition within IoT services requires careful resource utilization analysis, optimizing device energy use and cloud costs. This is crucial for efficient operations as devices transmit media queries to cloud services for content matching. Advanced audio processing techniques for smart building occupancy estimation further enhance HVAC energy efficiency, leading to cost savings and improved service quality [12, 77, 39].

The Adaptive Resource Allocation Framework (ARAF) exemplifies real-time resource optimization, enhancing system performance by dynamically adjusting resources based on demand, reducing energy consumption [75]. Basil's leaderless architecture supports energy-efficient fault-tolerant systems by enabling high concurrency and single round-trip transaction commitments, reducing overhead and enhancing performance [92].

Byzantine fault-tolerant causal ordering mechanisms impact energy consumption by ensuring robust fault detection and recovery, minimizing recomputation and resource-intensive error correction, conserving energy, and maintaining system efficiency [55]. Scalable microservices architectures improve system performance by enabling independent scaling and deployment of modular services, enhancing fault tolerance and capacity planning. This modularity allows for efficient resource management, ensuring performance constraints are met without excessive infrastructure costs [11, 37]. Dynamic resource allocation optimizes individual services for performance and energy efficiency, aligning energy consumption with system needs and further enhancing overall performance.

8 Conclusion

8.1 Innovative Approaches and Future Directions

The integration of audio processing, fault tolerance, scalability, microservices, natural language processing, fault detection, and distributed systems is driving substantial innovation and opening new research pathways. In audio processing, meta-learning models that adapt to the spectral-temporal characteristics of audio data are crucial for enhancing robustness and performance. These models, combined with advanced data augmentation techniques, are poised to significantly improve audio processing systems. Further exploration of robust phase estimation techniques for time-domain signal reconstruction will also advance this field.

In the realm of emotion recognition, systems that effectively identify emotions from audio and video under challenging conditions are paving the way for novel research opportunities. Future work could focus on optimizing acoustic encoders and expanding semantic codebooks to enhance audio reconstruction quality, thereby integrating audio processing with fault tolerance.

Incorporating Deep Active Learning Features into edge-AI frameworks for autonomous learning and continuous improvement in field applications presents another promising direction. This integration could greatly enhance the adaptability and efficiency of distributed systems. Additionally, optimizing libraries for real-time applications and expanding their scope to cover a broader range of audio processing tasks are vital future research areas.

In neuromorphic computing, Spiking Neural Networks hold significant promise for low-power audio processing applications. Future research should aim to optimize relevant libraries and explore additional applications to maximize energy efficiency and performance. The capabilities of architectures designed for real-time processing also offer promising avenues for developing energy-efficient audio processing solutions.

For distributed systems, enhancing protocols under dynamic failure models and integrating reliable leader-based designs are essential for improving system performance. Future endeavors should delve into these areas to bolster robustness and efficiency. Refining frameworks to support complex distributed systems can provide more robust solutions for ensuring system reliability.

The need for scalable and resilient cloud computing solutions is underscored by advancements in execution time and fault tolerance. Concurrently, scalable, self-managing frameworks leveraging peer-to-peer networks suggest future directions for system development.

In fault detection, future research should focus on validating monitoring techniques in industrial settings and developing integrated tools for their implementation. Additionally, optimizing resource allocation algorithms and exploring their applicability across domains like IoT and big data analytics are promising research trajectories.

Enhancing model interpretability without compromising performance remains a significant challenge. Innovative approaches warrant further exploration to improve transparency and usability in complex systems. Future developments will focus on expanding available metrics and datasets, improving error checking, and integrating machine learning models to enhance audio quality assessments.

These innovative approaches and future directions illustrate the dynamic integration of advanced computing concepts. As research progresses, these efforts will contribute to the development of cutting-edge technological solutions addressing contemporary computing challenges.

8.2 Future Directions and Open Challenges

The convergence of audio processing, fault tolerance, scalability, microservices, natural language processing, fault detection, and distributed systems presents numerous open challenges and future research opportunities. A primary challenge involves optimizing encryption processes in audio processing and machine learning to minimize computational demands while ensuring robust privacy protections. This optimization is crucial for maintaining data security without compromising system performance.

In microservices, future research should focus on expanding datasets to include more diverse projects and enhancing analytical tools to better evaluate microservices developed with various frameworks. These initiatives are vital for improving adaptability and performance across different applications, addressing existing limitations in resource availability and computational efficiency.

The reliance on statistical models for query production, particularly in media query processing for the Internet of Things, highlights an open challenge in integrating energy consumption with cloud billing. Future work should enhance the accuracy and reliability of these models to improve system efficiency. Moreover, exploring advanced techniques in serverless computing environments and integrating semi-supervised learning methods could enhance generalizability and resource efficiency.

In distributed systems, the limitations of existing frameworks in supporting consistent client migration during data center failures highlight the need for future enhancements to bolster system resilience and reliability. Additionally, optimizing algorithms for specific applications and enhancing self-stabilizing properties in fault-tolerant systems are critical areas for future exploration. Combining fault tolerance with privacy preservation and developing more efficient gradient filters are promising research directions.

Developing robust audio processing models capable of accurately classifying speaker fluency levels presents another challenge. Future research should focus on expanding audio datasets and refining model architectures to enhance classification accuracy. Furthermore, addressing the potential trade-off between interpretability and model complexity remains an open challenge.

In the context of fault tolerance, optimizing protocols and exploring additional mechanisms to reduce latency in high contention scenarios are vital research directions. These efforts will contribute to more efficient and resilient distributed systems capable of navigating dynamic environments.

The application of fault-tolerance metrics to other types of fault tolerance, such as failsafe and non-masking fault tolerance, presents a ripe area for exploration. Defining such metrics will broaden the applicability of fault-tolerance frameworks and enhance system reliability. Additionally, future research could focus on optimizing the performance of peer group frameworks under varying network conditions and enhancing security measures.

Finally, expanding datasets for AI-enabled sound pattern recognition, improving model robustness, and exploring real-world applications for enhanced patient monitoring are crucial for advancing audio processing and its integration with distributed systems. Addressing these open challenges and pursuing these future research directions will be instrumental in advancing the integration of these complex computing paradigms, paving the way for more robust, efficient, and scalable technological solutions.

References

[1] Hendrik Purwins, Bo Li, Tuomas Virtanen, Jan Schlüter, Shuo-Yiin Chang, and Tara Sainath. Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219, 2019.

[2] Athul Raimon, Shubha Masti, Shyam K Sateesh, Siyani Vengatagiri, and Bhaskarjyoti Das. Meta-learning in audio and speech processing: An end to end comprehensive review, 2024.

[3] Cory Stephenson, Jenelle Feather, Suchismita Padhy, Oguz Elibol, Hanlin Tang, Josh McDermott, and SueYeon Chung. Untangling in invariant speech recognition, 2020.

[4] Hannah Bos and Dylan Muir. Sub-mw neuromorphic snn audio processing applications with rockpool and xylo, 2022.

[5] Murat Isik, Hiruna Vishwamith, Kayode Inadagbo, and I. Can Dikmen. Hpcneuronet: Advancing neuromorphic audio signal processing with transformer-enhanced spiking neural networks, 2023.

[6] Shahin Amiriparian, Tobias Hübner, Maurice Gerczuk, Sandra Ottl, and Björn W. Schuller. Deepspectrumlite: A power-efficient transfer learning framework for embedded speech and audio processing from decentralised data, 2021.

[7] Suvam Mukherjee, Nitin John Raj, Krishnan Govindraj, Pantazis Deligiannis, Chandramouleswaran Ravichandran, Akash Lal, Aseem Rastogi, and Raja Krishnaswamy. Reliable state machines: A framework for programming reliable cloud services, 2019.

[8] Manuel Mazzara. Deriving specifications of dependable systems: toward a method, 2010.

[9] Riyansha Singh, Parinita Nema, and Vinod K Kurmi. Towards robust few-shot class incremental learning in audio classification using contrastive representation, 2024.

[10] Ohad Cohen, Gershon Hazan, and Sharon Gannot. Multi-microphone and multi-modal emotion recognition in reverberant environment, 2024.

[11] Mohammad Imranur, Rahman, Sebastiano Panichella, and Davide Taibi. A curated dataset of microservices-based systems, 2019.

[12] Francesco Renna, Joseph Doyle, Vasileios Giotsas, and Yiannis Andreopoulos. Media query processing for the internet-of-things: Coupling of device energy consumption and cloud infrastructure billing, 2016.

[13] Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. Clap learning audio concepts from natural language supervision. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[14] Jeff Hwang, Moto Hira, Caroline Chen, Xiaohui Zhang, Zhaoheng Ni, Guangzhi Sun, Pingchuan Ma, Ruizhe Huang, Vineel Pratap, Yuekai Zhang, Anurag Kumar, Chin-Yun Yu, Chuang Zhu, Chunxi Liu, Jacob Kahn, Mirco Ravanelli, Peng Sun, Shinji Watanabe, Yangyang Shi, Yumeng Tao, Robin Scheibler, Samuele Cornell, Sean Kim, and Stavros Petridis. Torchaudio 2.1: Advancing speech recognition, self-supervised learning, and audio processing components for pytorch, 2023.

[15] Pranay Manocha, Zeyu Jin, and Adam Finkelstein. Audio similarity is unreliable as a proxy for audio quality, 2022.

[16] Manuel Mazzara. On methods for the formal specification of fault tolerant systems, 2012.

[17] Robert Constable. Effectively nonblocking consensus procedures can execute forever - a constructive version of flp, 2011.

[18] Garima Sharma, Kartikeyan Umapathy, and Sridhar Krishnan. Trends in audio signal feature extraction methods. *Applied Acoustics*, 158:107020, 2020.

[19] Raphael Lenain, Jack Weston, Abhishek Shivkumar, and Emil Fristed. Surfboard: Audio feature extraction for modern machine learning, 2020.

[20] Vakada Naveen, Arvind Krishna Sridhar, Yinyi Guo, and Erik Visser. Comprehensive audio query handling system with integrated expert models and contextual understanding, 2024.

[21] Md Mohaimenuzzaman, Christoph Bergmeir, and Bernd Meyer. Deep active audio feature learning in resource-constrained environments, 2024.

[22] Nirupam Gupta, Shuo Liu, and Nitin H. Vaidya. Byzantine fault-tolerant distributed machine learning using stochastic gradient descent (sgd) and norm-based comparative gradient elimination (cge), 2021.

[23] Alexandros Gazis and Eleftheria Katsiri. Crowd tracking and monitoring middleware via map-reduce, 2022.

[24] Sathvik Reddy O and Sakthivel SM. Fault tolerant fpga implementation on redundancy techniques and ecg denoising, 2023.

[25] Andrei Damien, Cezara Dragoi, Alexandru Militaru, and Josef Widder. Reducing asynchrony to synchronized rounds, 2019.

[26] Cinzia Di Giusto and Jorge A. Pérez. Session types with runtime adaptation: Overview and examples, 2013.

[27] Heidi Howard and Richard Mortier. Paxos vs raft: Have we reached consensus on distributed consensus?, 2020.

[28] MohammadTaghi HajiAghayi, Dariusz R. Kowalski, and Jan Olkowski. Improved communication complexity of fault-tolerant consensus, 2022.

[29] Alan Preciado-Grijalva and Ramon F. Brena. Speaker fluency level classification using machine learning techniques, 2018.

[30] Yupei Li, Qiyang Sun, Hanqian Li, Lucia Specia, and Björn W. Schuller. Detecting machine-generated music with explainability – a challenge and early benchmarks, 2024.

[31] Muhammad Asif Jan, Fahd Ali Zahid, Mohammad Moazam Fraz, and Arshad Ali. Exploiting peer group concept for adaptive and highly available services, 2003.

[32] Rodrigo R. Barbieri, Enrique S. dos Santos, and Gustavo M. D. Vieira. Decentralized validation for non-malicious arbitrary fault tolerance in paxos, 2020.

[33] Wenbing Zhao. Proactive service migration for long-running byzantine fault tolerant systems, 2008.

[34] Huynh Tu Dang, Pietro Bressana, Han Wang, Ki Suh Lee, Hakim Weatherspoon, Marco Canini, Fernando Pedone, and Robert Soulé. Network hardware-accelerated consensus, 2016.

[35] Alejandro Naser-Pastoriza, Gregory Chockler, and Alexey Gotsman. Fault-tolerant computing with unreliable channels (extended version), 2023.

[36] Ali Shoker. Exploiting universal redundancy, 2016.

[37] Rasha Faqeh, Andrè Martin, Valerio Schiavoni, Pramod Bhatotia, Pascal Felber, and Christof Fetzer. Pcraft: Capacity planning for dependable stateless services, 2022.

[38] Sarthak Joshi and Sathish Vadhiyar. Partreper-mpi: Combining fault tolerance and performance for mpi applications, 2023.

[39] Francesco Renna, Joseph Doyle, Vasileios Giotsas, and Yiannis Andreopoulos. Query processing for the internet-of-things: Coupling of device energy consumption and cloud infrastructure billing, 2016.

[40] Haohe Liu, Xuenan Xu, Yi Yuan, Mengyue Wu, Wenwu Wang, and Mark D. Plumbley. Semanticodec: An ultra low bitrate semantic audio codec for general sound, 2024.

[41] Jonah Casebeer, Nicholas J. Bryan, and Paris Smaragdis. Meta-af: Meta-learning for adaptive filters, 2022.

[42] Gloria Dal Santo, Gian Marco De Bortoli, Karolina Prawda, Sebastian J. Schlecht, and Vesa Välimäki. Flamo: An open-source library for frequency-domain differentiable audio processing, 2024.

[43] Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. madmom: a new python audio and music signal processing library, 2016.

[44] Jayneel Parekh, Sanjeel Parekh, Pavlo Mozharovskyi, Florence d'Alché Buc, and Gaël Richard. Listen to interpret: Post-hoc interpretability for audio networks with nmf, 2022.

[45] Prajoy Podder, Md. Mehedi Hasan, Md. Rafiqul Islam, and Mursalin Sayeed. Design and implementation of butterworth, chebyshev-i and elliptic filter for speech signal analysis, 2020.

[46] Jack Geraghty, Jiazheng Li, Alessandro Ragano, and Andrew Hines. Aqp: An open modular python platform for objective speech and audio quality metrics, 2022.

[47] Yiming Chen, Xianghu Yue, Xiaoxue Gao, Chen Zhang, Luis Fernando D'Haro, Robby T. Tan, and Haizhou Li. Beyond single-audio: Advancing multi-audio processing in audio large language models, 2024.

[48] Mark Anderson, Tomi Kinnunen, and Naomi Harte. Learnable frontends that do not learn: Quantifying sensitivity to filterbank initialisation, 2023.

[49] Daniel Wolff, Rémi Mignot, and Axel Roebel. Audio defect detection in music with deep networks, 2022.

[50] Johanna Devaney. Digital audio processing tools for music corpus studies, 2021.

[51] Omkar Deshpande, Kharanshu Solanki, Sree Pujitha Suribhatla, Sanya Zaveri, and Luv Ghodasara. Simulating the dft algorithm for audio processing, 2021.

[52] Soufiyan Bahadi, Eric Plourde, and Jean Rouat. Efficient sparse coding with the adaptive locally competitive algorithm for speech classification, 2024.

[53] Pedro A. R. S. Costa, Xiao Bai, Fernando M. V. Ramos, and Miguel Correia. Medusa: An efficient cloud fault-tolerant mapreduce, 2015.

[54] Mina Moradi Kordmahalleh, Mohammad Gorji Sefidmazgi, Jafar Ghaisari, and Javad Askari. A fault-tolerant distributed detection of two simultaneous events in wireless sensor networks, 2016.

[55] Anshuman Misra and Ajay Kshemkalyani. Byzantine fault tolerant causal ordering, 2021.

[56] Vincenzo De Florio, G. Deconinck, and Rudy Lauwereins. Software tool combining fault masking with user-defined recovery strategies, 2014.

[57] Milad Ghaznavi, Elaheh Jalalpour, Bernard Wong, Raouf Boutaba, and Ali Jose Mashtizadeh. Fault tolerance for service function chains, 2020.

[58] Lauri Vihman, Maarja Kruusmaa, and Jaan Raik. Overview of fault tolerant techniques in underwater sensor networks, 2019.

[59] Rodrigo R. Barbieri and Gustavo M. D. Vieira. Hardened paxos through consistency validation, 2017.

[60] Miruna Stoicescu, Jean-Charles Fabre, and Matthieu Roy. Experimenting with component-based middleware for adaptive fault tolerant computing, 2012.

[61] Alexander Droob, Daniel Morratz, Frederik Langkilde Jakobsen, Jacob Carstensen, Magnus Mathiesen, Rune Bohnstedt, Michele Albano, Sergio Moreschini, and Davide Taibi. Workrs: Fault tolerant horizontal computation offloading, 2023.

[62] Pushyami Kaveti and Hanumant Singh. Ros rescue : Fault tolerance system for robot operating system, 2019.

[63] Gabriele D'Angelo, Stefano Ferretti, Moreno Marzolla, and Lorenzo Armaroli. Fault-tolerant adaptive parallel and distributed simulation, 2016.

[64] Jinli Duan and Shasha Dennis. Jup2kub: algorithms and a system to translate a jupyter notebook pipeline to a fault tolerant distributed kubernetes deployment, 2023.

[65] Cezara Dragoi, Constantin Enea, Srinidhi Nagendra, and Mandayam Srivas. A domain specific language for testing consensus implementations, 2023.

[66] Alessandro Fantechi, Gloria Gori, and Marco Papini. Runtime reliability monitoring for complex fault-tolerance policies, 2022.

[67] Suyash Gupta, Sajjad Rahnama, Erik Linsenmayer, Faisal Nawab, and Mohammad Sadoghi. Reliable transactions in serverless-edge architecture, 2022.

[68] Priyanka Mondal, Maximilian Algehed, and Owen Arden. Applying consensus and replication securely with flaqr, 2022.

[69] Shuo Liu. A survey on fault-tolerance in distributed optimization and machine learning, 2021.

[70] Radan Ganchev. Voice signal processing for machine learning. the case of speaker isolation, 2024.

[71] Deepft: Fault-tolerant edge computing using a self-supervised deep surrogate model.

[72] Nikos D. Fakotakis, Stavros Nousias, Gerasimos Arvanitis, Evangelia I. Zacharaki, and Konstantinos Moustakas. Ai-enabled sound pattern recognition on asthma medication adherence: Evaluation with the rda benchmark suite, 2023.

[73] Roswitha Bammer, Monika Dörfler, and Pavol Harar. Gabor frames and deep scattering networks in audio processing, 2019.

[74] Alexander Borzunov, Max Ryabinin, Artem Chumachenko, Dmitry Baranchuk, Tim Dettmers, Younes Belkada, Pavel Samygin, and Colin Raffel. Distributed inference and fine-tuning of large language models over the internet, 2023.

[75] Surender Kumar, R. K. Chauhan, and Parveen Kumar. A low overhead minimum process global snapshop collection algorithm for mobile distributed system, 2010.

[76] Philipp Schneider. Byzantine fault tolerant protocols with near-constant work per node without signatures, 2025.

[77] Qian Huang, Zhenhao Ge, and Chao Lu. Occupancy estimation in smart buildings using audio-processing techniques, 2016.

[78] Manuel Bravo, Alexey Gotsman, Borja de Régil, and Hengfeng Wei. Unistore: A fault-tolerant marriage of causal and strong consistency (extended version), 2021.

[79] Marius Poke and Colin W. Glass. Formal specification and safety proof of a leaderless concurrent atomic broadcast algorithm, 2017.

[80] Brendan Harding, Markus Hegland, Jay Larson, and James Southern. Scalable and fault tolerant computation with the sparse grid combination technique, 2014.

[81] H. S. Aravinda, H. D. Maheshappa, and Ranjan Moodithaya. Implementation of the six channel redundancy to achieve fault tolerance in testing of satellites, 2010.

[82] Joshua Ball. Voice activity detection (vad) in noisy environments, 2023.

[83] Yue Li, Florian A. Kunneman, and Koen V. Hindriks. A near-real-time processing ego speech filtering pipeline designed for speech interruption during human-robot interaction, 2024.

[84] Rajat Hebbar, Digbalay Bose, Krishna Somandepalli, Veena Vijai, and Shrikanth Narayanan. A dataset for audio-visual sound event detection in movies, 2023.

[85] Swapnil Bhosale, Rupayan Chakraborty, and Sunil Kumar Kopparapu. Semi supervised learning for few-shot audio classification by episodic triplet mining, 2021.

[86] Siavash Shams, Sukru Samet Dindar, Xilin Jiang, and Nima Mesgarani. Ssamba: Self-supervised audio representation learning with mamba state space model, 2025.

[87] Jie Xu, Brian Randell, Alexander Romanovsky, Robert J. Stroud, and Avelino F. Zorzo. Supporting and controlling complex concurrency in fault- tolerant distributed systems, 2021.

[88] Faxian Cao, Yongqiang Cheng, Adil Mehmood Khan, and Zhijing Yang. Are microphone signals alone sufficient for self-positioning?, 2023.

[89] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

[90] Tycho Max Sylvester Tax, Jose Luis Diez Antich, Hendrik Purwins, and Lars Maaløe. Utilizing domain knowledge in end-to-end audio processing, 2017.

[91] Armando Castañeda, Pierre Fraigniaud, Ami Paz, Sergio Rajsbaum, Matthieu Roy, and Corentin Travers. A topological perspective on distributed network algorithms, 2020.

[92] Florian Suri-Payer, Matthew Burke, Zheng Wang, Yunhao Zhang, Lorenzo Alvisi, and Natacha Crooks. Basil: Breaking up bft with acid (transactions), 2021.

[93] Hagit Attiya and Jennifer L. Welch. Multi-valued connected consensus: A new perspective on crusader agreement and adopt-commit, 2023.