Acoustic-Based Bushfire Detection: Comparative Evaluation and Resource-Efficient Approaches for Different Machine Learning Methods

A thesis submitted in part fulfilment of the degree of
Bachelor of Engineering (Honours)

by
Rui Jiang
U7228214

Supervisor: Ms. Qiangyang Zhuang
Examiner: Dr. Xiangyun (Sean) Zhou

Australian National University

College of Engineering and Computer Science
The Australian National University

May 2025

This thesis contains no material which has been accepted for the award of any other degree or diploma in any university. To the best of the author's knowledge, it contains no material previously published or written by another person, except where due reference is made in the text.

Rui Jiang
30 May 2025

©Rui Jiang

Acknowledgements

This has conceivably been the most conceptually perspicuous phase of my university experience, frankly an even split of torment and rapture, although it has been an emotional rollercoaster. Undeniably, it has represented the most developmentally positive academic period for me.

Many distinguished people possess my deep appreciation. I deeply value each of them. Initially and principally, for my cherished companion, Daisy, who buoyed me through many impending collapses and fostered as well as strengthened me steadfastly as I bordered upon the precipice of scholastic despondency.

I sincerely extend gratitude toward my mate Ricky Guo since he miraculously envisions practical resolutions whenever I'm swamped by predicaments. He lent assistance, ergo the consequences seem quite outstanding.

Miss Qingyang Zhuang, my supervisor, possesses my deep gratitude. Notwithstanding that I am cognisant I have not consistently fulfilled anticipations, she strengthened me greatly. Dr Xiangyun Zhou, being my co-supervisor, garners my equivalent appreciation as my scholarly verifier, adroitly curtailing the extravagant, unrealistic ideas of mine which at times menaced to transmute this undertaking into an unmanageable leviathan. Both supervisors, through their applied acumen and vocational skill, have consistently grounded me.

Rebbecca furnished explicit instruction in my journal therefore I appreciate her. This specific instruction proved to be quite important for effectively maintaining my trajectory. My mates have emboldened me to confront unavoidable setbacks. I also recognise them, for furnishing me with fortitude toward confronting potential hazards.

My mates as well as rellies in Melbourne, Sydney, as well as Adelaide possess my deep gratitude. Their backing has strengthened my fortitude throughout this study's formidable tribulations. My well-regarded mates aged 8 as well as 17 years warrant particular acknowledgement, for a kick-off. Their skill for elevating my morale throughout the bleakest junctures of this escapade remained absolutely outstanding. They exist as firm mates, perpetually rendering fortitude. My folks, in continuing to furnish that emotional succour from afar, despite their not viewing me these past eight years, garner my deep gratefulness. I remain astonished by the manner their affection has exceeded corporeal limitations. I must also acknowledge my aunt residing in Adelaide, since she has beneficently taken on my stand-in "mom" position and offers the affection, solicitude, plus gentle reprimands solely a mother archetype can. Her presence in this location renders this unfamiliar country related to one's domicile.

Since the prescribed stipulations are fulfilled, allow me to proceed into more unorthodox territory. I have to recognise my cat since she is wonderfully fluffy, unusually well-behaved, and has rendered vital emotional support via curling up next to me throughout many testing sessions. Every anxiety-reduction scheme diminishes when weighed against her being. Her existence has been deeply restorative.

This investigation has been genuinely energised via coffee, Monster energy drinks, alongside a range of other caffeinated beverages. I'd happily pen a more wide-ranging paean, assuming sponsorship prospects were procurable, but unfortunately, such luck is beyond me. I believe the intellect who

uncovered caffeine, in any event, warrants a Nobel Prize regarding Human Productivity, it should be declared.

Although I'm unable to endorse this method, wines and whisky have furnished outstanding stress reduction, I must mention. Booze has a proclivity to spark inspired concepts, yet temperance unfailingly causes one to scratch their noggin in bafflement at yesterday's "genius" concepts.

I appreciate my colleagues. They generously jest at my expense plus they exercise restraint so they don't overwhelm me with labour which induces despondency. They warrant accolades by reason of their temperance.

My cherished important other has stated that she only obtained a solitary sentence of recognition presently. I am dedicating this whole additional paragraph with a view to rectifying this terrible omission. Without her presence, I might have gave in to my inclination for averting meaningful matters via clever dawdling.

Abstract

Conventional bushfire detection systems rely on costly infrastructure with limited coverage, such as infrared towers and satellites, which require substantial power for remote deployment and perform poorly under smoky conditions. We propose an acoustic-based detection system that leverages the characteristic "crackling" sounds produced by fires. Using a curated dataset of 3,259 audio clips (1,697 fire, 1,562 non-fire) containing ambient environmental sounds, we develop and compare three neural network architectures using 64-bin Mel spectrograms as input. Our Convolutional Recurrent Neural Network (CRNN) integrates bidirectional GRU layers with attention mechanisms for enhanced temporal modeling. Compared to spiking neural network alternatives, the CRNN achieves 97.39% accuracy and 98.05% recall on held-out test data, demonstrating superior threshold robustness. External validation on 89 novel clips from unseen environments confirms strong generalization capability (93.3% accuracy, 98.2% recall). Despite having 48× more parameters than spiking networks, the CRNN delivers 19× faster inference (0.04ms per clip) while maintaining a compact 3.03MB footprint. The system meets the <5% miss-rate requirement and shows potential for distributed deployment, though actual energy consumption on edge devices requires further validation. This work demonstrates that acoustic-only approaches can provide a practical alternative to traditional vision-based methods for early bushfire detection.

Contents

List of Figures

List of Tables

Nomenclature

AED	Audio-Event Detection
CNN	Convolutional Neural Network
CRNN	Convolutional–Recurrent Neural Network (CNN + GRU)
CSNN	Convolutional Spiking Neural Network
GRU	Gated Recurrent Unit
LIF	Leaky Integrate-and-Fire (spiking neuron)
LSTM	Long Short-Term Memory
PANN	Pre-trained Audio Neural Network
RNN	Recurrent Neural Network
SNN	Spiking Neural Network
dB	Decibel (log-amplitude unit)
FFT	Fast Fourier Transform
MFCC	Mel-Frequency Cepstral Coefficient
Mel	Mel Spectrogram (64-bin)
SNR	Signal-to-Noise Ratio
STFT	Short-Time Fourier Transform
AP	Average Precision (area under PR)
AUC	Area Under the Curve
AUROC	Area Under the ROC Curve
mAP	mean Average Precision (multi-class)
PR	Precision–Recall (curve)
ROC	Receiver-Operating-Characteristic (curve)
Adam	Adaptive Moment Estimation (optimizer)
MSE	Mean Square Error (loss function)
ReLU	Rectified Linear Unit (activation)

A100	NVIDIA A100 GPU (40 GB)
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture (NVIDIA)
FP16	16-bit Floating-Point (half precision)
FLOPs	Floating-Point Operations (per second)
GPU	Graphics Processing Unit
NVML	NVIDIA Management Library
SRAM	Static Random-Access Memory
TPU	Tensor Processing Unit (Google Edge)
VRAM	Video RAM (GPU memory)
ASA	Australian Space Agency
DCASE	*Detection&Classificationof AcousticScenes&Events*
IoT	Internet of Things
PyTorch	Python deep-learning framework
B	Batch size (clips)
C	Channels / feature maps
H	Height (Mel bins)
W	Width (time frames)
T	Discrete time steps (SNN, T=15)
L	Layers (network depth)
h	Hidden-state size (GRU units)
k	Kernel size (convolution)
m	Time steps after pooling
n	Batch size (alt.)
β	Membrane decay factor
θ	Firing threshold (LIF)
σ	Sigmoid activation
$\odot$	Element-wise multiplication
$U[t]$	Membrane potential at t
$X[t]$	Input current at t
h_t	Hidden state at t (RNN/GRU)
x_t	Input at t
r_t	Reset gate (GRU)
z_t	Update gate (GRU)
α_t	Attention weight at t

Introduction

Bushfires represent a significant hazard worldwide, posing threats to economic stability, environmental health, and societal well-being. For instance, the 2025 bushfire in California, United States, resulted in estimated economic losses ranging from $95 billion to $164 billion [16]. In Australia, bushfires also pose a major risk; the 2020 Canberra fires alone burned nearly 90,000 hectares (approximately 40% of the Australian Capital Territory) [1]. Although traditional approaches such as image- or thermal-based methods are widely adopted, they can be expensive, require substantial infrastructure, and may perform poorly under low-visibility conditions. According to an ASA satellite-based fire detection project, early-stage detection remains a challenge [2], highlighting limitations in remote or resource-constrained settings. Recent research efforts have proposed leveraging the "crackling" sounds produced by fire as an alternative means of detection. Such an acoustic-based method may offer faster response times when visual cues are obstructed by smoke or terrain, while also reducing infrastructure costs. Building on this premise, the aim of this project is to develop a sound-based, machine-learning-driven system for early bushfire detection that can address the limitations of current methods and more effectively safeguard communities. The primary goal of this project is to develop and validate a robust, acoustic-signal-based machine-learning model that can be feasibly run on personal devices. To achieve this objective, the following specific aims have been identified:

1. **Model Investigation and Comparison:** Evaluate multiple machine-learning architectures (e.g., PANN, CSNN, CRNN) for the purpose of bushfire detection.

2. **Data Acquisition and Preprocessing:** Collect real-world audio recordings that reflect diverse environmental conditions (e.g., wind, wildlife) and preprocess them appropriately to ensure reliable training inputs.

3. **Model Evaluation:** Assess each model's performance using metrics such as accuracy, F1-score, and resource consumption, among others.

This thesis primarily focuses on the algorithmic and experimental aspects of acoustic-based detection and does not address hardware sensor design or large-scale field deployment. All sound data used in this work are publicly available. References to "bushfire" sound data throughout this thesis encompass real recordings of fires, as well as public database samples, without targeting specific fire subtypes or particular environmental conditions. In this thesis, we focus on acoustic data originating from *actual* bushfire recordings, rather than relying solely on controlled laboratory sound files or generic online audio repositories. Building upon prior research into spiking neural architectures [14] and Pretrained Audio Neural Networks (PANNs) [12, 19], we aim to finalize a model that optimally balances detection performance and computational complexity for resource-constrained environments. A central contribution of this work is the reduction of false positives when detecting genuine bushfire events in real-world conditions. By developing a system capable of running on personal devices, we

effectively lessen the required computing power, thereby decreasing operational costs and broadening the method's applicability. In addition, we demonstrate the advantages of employing an RNN-based architecture for audio classification under noisy environments. This includes showcasing how the recurrent layers can more effectively capture temporal patterns and mitigate the challenges associated with unpredictable, real-life acoustic scenarios. Through these explorations, our thesis expands upon current understanding of lightweight, robust, and cost-effective bushfire detection systems.

The structre of this thesis is as follow:

1. **Chapter 2 (Literature Review)**: Outlines existing bushfire detection approaches, from conventional visual/thermal systems to newer acoustic-based methods. Additionally, it evaluates machine-learning techniques such as CNNs, spiking neural networks, and recurrent architectures for audio event classification, highlighting their respective strengths and drawbacks.

2. **Chapter 3 (Methodology)**: Describes the steps taken to gather real-world fire and non-fire audio, including preprocessing and feature extraction. It also details the ML models tested—initially Pretrained Audio Neural Networks (PANNs), then a reproduced/improved Convolutional Spiking Neural Network (CSNN), and ultimately a Convolutional Recurrent Neural Network (CRNN) as the final approach.

3. **Chapter 4 (Results and Discussion)**: Presents experimental outcomes, comparing each model's performance in terms of accuracy, resource consumption, and robustness to environmental noise. It also discusses key findings and potential reasons for observed successes or shortcomings.

4. **Chapter 5 (Conclusion and Future Work)**: Summarises the main contributions of this research in advancing acoustic-based bushfire detection. This chapter reflects on outstanding limitations, provides suggestions for future investigations, such as additional data collection or alternative architectures, and underscores the broader implications for real-world fire detection efforts.

Literature Review

According to the existing literature, it can be found that the traditional methods for fire detection are divided into two types: image-based detection (including satellite imaging and unmanned aerial vehicle photography) and infrared systems based on thermal imaging.[17, 21] Due to limited spatial resolution, low transit frequency and interference from clouds or smoke, satellite and uncrewed aerial vehicle (UAV) fire detection often misses small or short-lived fires[33, 38]. However, thermal imaging systems are limited by their detection range, have difficulty penetrating thick smoke or reflective surfaces, and may produce inaccurate readings in complex environments [28, 31]. Acoustic detection, as an emerging detection method, has solved some of the problems of traditional techniques through continuous development. For instance, acoustic detection has a cost advantage over conventional fire detection, which is even more pronounced in early fire detection[7, 13, 26, 40].

Early studies support fire crackling as an acoustic signature for fire detection [7], which is the first time scientists proposed a method using sound to detect fire. Additionally, some research has indicated that fires produce unique acoustic characteristics that help distinguish fire situations under laboratory conditions. For example, crown fires and surface fires produce distinguishable frequency ranges, sometimes concentrated between 0-400Hz. However, the acoustic frequency of flames can reach 15 kHz depending on the type of burning material.[37] This early foundational work suggests that these signals have some significant differences from environmental noise and travel faster than optical signals in conditions of heavy smoke. But acoustic detection still has slow adoption in real-world situations, partly due to environmental noise and the novelty of deploying large-scale acoustic sensors[30]. Despite these challenges, acoustic signals still have transmission advantages in conditions where optical systems are limited and require relatively lower-cost hardware to capture audio[40]. This makes sound-based detection methods a strong alternative candidate to traditional detection methods[26].

Recent advances in machine learning have driven progress in detecting, classifying, and localising sound environments. In earlier research, some traditional machine learning algorithms were frequently used, such as support vector machines (SVM) or hidden Markov models (HMM), and were trained based on manually designed features [8, 25]. Although these techniques might be effective on limited datasets, they often perform poorly in real-world environments.

During the entire literature reading process, the four models were carefully studied and learned, including CNN, RNN, SNN, and PANN.

1. Deep learning introduces convolutional neural networks (CNNs) for audio, which are usually achieved by converting signals into spectrogram "images"[29]. According to research, CNN is good at identifying spatial and frequency patterns, making it widely popular in tasks such as urban sound classification and wildlife monitoring[11, 18, 29]. Its main advantage lies in automatic feature extraction [11, 29]. However, if it is not carefully optimised, the computational burden may be relatively heavy[9]. Considering the issue of computational burden, this model

will not be used in the paper.

2. LSTM or GRU types of recurrent neural networks (RNNs) improve temporal dependency modelling in sequential data [35]. Together with convolutional layers (to create CRNNs), they can concurrently capture longer temporal contexts (through RNN) and local frequency-time patterns (via CNN) [24]. For flame cracking noises, this synergistic effect is especially crucial since the temporal development of the sound can be a really good indication of combustion activity[3, 27]. However, RNNs may suffer from longer training times due to sequential processing constraints and may require careful gradient management to avoid instability during training[4, 10]. The modification and impact of using the RNN model will be discussed in the methodology.

3. Spik-based neural networks (SNNs) represent a more biologically inspiring approach, in which event-driven computing can provide power efficiency. Research on pulse-based solutions, sometimes combined with lightweight convolution modules, has shown potential on dedicated low-power devices[14]. However, the training of SNNs is more challenging. It requires fine adjustment of the impulse threshold and alternative gradient methods; otherwise, the model will be prone to under-fitting problems [22, 32, 34]. We will elaborate on this method in detail in the methodology

4. Pre-trained Audio neural networks (PANNs) utilise large-scale audio datasets and achieve general performance through transfer learning[12]. However, they have high training time or memory usage, making them less ideal for inference on resource-constrained devices [19]. The use and results of this model will also be mentioned in this article. However, whether the model can perform better is not discussed and verified in detail in this article.

Collecting representative audio for shrub fire detection is inherently challenging because fires often occur unpredictably and in remote locations. This has led to many studies relying on public environmental datasets or recordings of burning samples in controlled laboratories [6]. Generating based on the spectrogram of the corresponding audio block (such as short-time Fourier transform or MEL spectrogram) remains the standard practice, enabling neural models to view audio as a two-dimensional representation[15]. Additional transformations, such as MEL frequency cepstral coefficients (MFCC) or amplitude-phase characteristics, are also common methods to highlight the specific acoustic properties of combustion[36].

In some studies, noise filtering or data augmentation is adopted to improve robustness[27]. However, some methods choose to process the captured audio "as is", especially under the assumption that the environment is relatively quiet or that extensive real-world conditions cannot be captured[6]. This is why later in this paper, we proportionally mix real fire sounds with fire sounds in the dataset, and non-fire sounds mix the jungle sounds with the jungle sounds in the dataset [20, 23, 30, 39]. In this case, the inherent variability of the recording may complicate the classification task, but it can also highlight the model's ability to deal with real-world data without extensive pre-processing[36].

The research underlines the viability of acoustic-based bush fire detection, although significant shortcomings remain. First of all, many earlier studies limited their relevance under real and erratic field situations by depending on tightly filtered samples or laboratory-scale data. Second, even if neural network designs like CNN and PANN have made great strides, running on mobile devices or low-power sensors usually results in performance or computational constraints. Early detection and the lowest false alarm rate still depend on lightweight, accurate solutions especially made for real-world conditions. Finally, there is still a lack of a large open access jungle fire sound library in this field, making it difficult to unify comparison methods or promote the improvement of collaborative models.

These identified gaps give rise to three fundamental research questions that this study seeks to address:

RQ1: Can acoustic signatures alone provide sufficient accuracy for reliable bushfire detection? While previous studies suggest acoustic viability, comprehensive validation of realistic field recordings with minimal pre-processing remains limited.

RQ2: What are the accuracy-resource trade-offs among different neural architectures for acoustic fire detection? The literature reveals that lightweight models often sacrifice performance, while high-performing models may be computationally prohibitive for edge deployment.

RQ3: How well do acoustic fire detection models generalise to completely unseen environmental conditions? Most existing studies lack rigorous external validation on audio recorded in different locations and acoustic environments from the training data.

Driven by these research questions, this paper reproduces, tests, and improves an acoustic-based detection process that utilises realistic field recordings instead of controlled or synthetic data. Based on systematic comparison of existing machine learning architectures—including CNNs, RNNs, SNNs, and PANNs—we strive to strike a balance between high detection performance and feasible deployment on standard consumer-grade hardware. Through this approach, we aim to provide empirical evidence for acoustic fire detection viability while addressing the computational constraints that limit widespread deployment in resource-constrained environments.

Methodology

3.1 Overview of the Engineering Model

Throughout this project, we used an iterative engineering approach to guide the development and validation of our system. Instead of relying on a single architecture from the beginning, we refined our bushfire detection pipeline step by step, using feedback from each experimental phase. The process can be thought of as follows:

1. **Define Requirements and Prepare Data**: We began by clarifying the types of data we needed and identifying which data would be suitable for personal devices. The dataset had to include real-world sounds representing fire and non-fire situations specific to the bush environment. Some online datasets were also blended with real-world sound datasets and appropriately labelled. Together, these elements formed our initial dataset.

2. **First Model Implementation and Testing**: We start with a pre-trained audio neural network (PANN), which we chose because of its outstanding performance in general audio tasks. After testing, we recorded its limitations, including but not limited to the high consumption and medium accuracy rate that occurred during the training process, which affected our next step of work.

3. **Refinement and the Second Model**: We tried and reproduced a lightweight convolutional spiker neural network (CSNN) to reduce computational complexity. Although this has improved efficiency, its accuracy rate tends to be stable under certain conditions. It has not reached the accuracy rate generated in the paper, indicating that there is still room for improvement in the model.

4. **Final Architecture**: Based on the advantages of CNN feature extraction, we replaced impulse components with recurrent neural networks (RNNs) to create CRNN. This final design balances time series modelling and controllable complexity, and is more suitable for deployment on personal devices.

5. **Integration and Evaluation**: Finally, we conducted a comprehensive test of the updated CRNN process, comparing its performance, resource requirements, and false alarm rate with earlier models (PANN and CSNN) to ensure that our key goals were met.

Following the iterative methods above ensures that we combine the insights into each model's performance and resource limitations and eventually arrive at a design suitable for real-world acoustic data and the hardware limitations of typical user devices.

3.2 Data Acquisition and Preparation

The first step of this project is to collect diverse audio recordings to represent fire and non-fire scenarios. We collected clips from publicly available online resources, focusing on the sounds of real bushfires rather than laboratory or artificially generated audio. We then screened each audio file to ensure they were relevant to the experimental requirements. In addition to these specific fire samples, we also obtained the daily environmental noises of the bush, such as wind, bird chirping, and human activity sounds, to ensure that our dataset reflects the various conditions encountered in actual deployment.

After the audio files are assembled, we will use code to divide each recording segment into shorter segments, each lasting five seconds, so that individual samples can be marked as "fire" or "non-fire". We choose not to use advanced noise reduction techniques, tend to retain the original audio characteristics, and challenge the model through fundamental environmental changes. However, all fragments were converted to a consistent sampling rate, and the amplitude levels were normalised to avoid deviations caused by volume differences. Subsequently, the dataset generated in the code was divided into training, validation, and test sets to maintain a balanced ratio of fire and non-fire samples. This method aims to support fair model evaluation and minimise over-fitting.

3.3 Model Exploration

After preparing the dataset, we evaluated different machine learning architectures to identify a model that could effectively capture the acoustic characteristics of jungle fires and was computationally feasible on personal devices. It is worth emphasising that the three models we have selected simultaneously rely on converting existing audio files into MEL spectrograms for analysis. Generally speaking, our exploration is divided into three main stages:

3.3.1 PANN stage

Stage	Operation(s)	Output Shape
Input	Waveform	[B, samples]
STFT	Spectrogram extractor	$[B, 1, 1000, 513]$
Log-mel	64-bin logmel	$[B, 1, 1000, 64]$
BN_0	Batch-norm	$[B, 1, 1000, 64]$
Conv Blk 1	2 × [Conv 3×3@64 + BN + ReLU], MaxPool 2×2	$[B, 64, 500, 32]$
Conv Blk 2	2 × [Conv 3×3@128 + BN + ReLU], MaxPool 2×2	$[B, 128, 250, 16]$
Conv Blk 3	2 × [Conv 3×3@256 + BN + ReLU], MaxPool 2×2	$[B, 256, 125, 8]$
Conv Blk 4	2 × [Conv 3×3@512 + BN + ReLU], MaxPool 2×2	$[B, 512, 62, 4]$
Conv Blk 5	2 × [Conv 3×3@1024 + BN + ReLU], MaxPool 2×2	$[B, 1024, 31, 2]$
Conv Blk 6	2 × [Conv 3×3@2048 + BN + ReLU]	$[B, 2048, 31, 2]$
Global Pool	Mean + Max across time	$[B, 2048]$
FC_{2048}	Dense + ReLU	$[B, 2048]$
Output	Dense 527 + Sigmoid	$[B, 527]$

Table 3.1: Architecture summary of PANNs (Pretrained Audio Neural Networks) CNN14 model for audio classification. Each "Conv Blk n" comprises two convolution–batch-normalisation–ReLU sub-layers followed by a 2×2 max-pool, unless otherwise noted.

We first assessed Pre-trained Audio Neural Networks (PANNs) as being our benchmark methodology for fire detection, especially the CNN14 design. PANNs constitute a lineage of deep convolutional

neural networks. They revolutionised audio classification and performed exceptionally well across many audio recognition benchmarks. The CNN14 model was pre-trained upon Google's AudioSet dataset, which incorporates over 2 million audio clips across 527 sound event classes, as well as it can recognise diverse acoustic patterns exceptionally, patterns that range from human activities to environmental sounds.

3.3.1.1 Technical Architecture:

An advanced 14-layer design is utilised in the CNN14 architecture via six progressive convolutional blocks that extract hierarchical audio features systematically, shown in 3.1. Log-mel spectrogram extraction (1000 frames $\times$ 64 mel bins) initiates the network and channel dimensions gradually escalate from 64 up to 2048 via convolutional blocks. Every block includes a duo of 3×3 convolution layers incorporating batch normalisation together with ReLU activation, subsequent to which 2×2 pooling operations curtail spatial dimensions insofar as they augment feature depth. The architecture culminates in a dual temporal pooling strategy (max + average pooling) and fully connected layers, and it amounts to roughly 38 million parameters.

3.3.1.2 Methodological Approach:

Our methodology built upon transfer learning principles. The fire detection capabilities were initiated by utilising the ample acoustic representations derived from AudioSet's varied sound classification system. We posited that as the pre-trained attributes had discerned disparate environmental sounds like crackling, hissing, and combustion-related audio cues, they would furnish a strong groundwork when pinpointing fire-specific acoustic indicators within our bushfire dataset.

CNN14 exhibited decent classification accuracy regarding our fire detection task. Nonetheless, some substantial constraints became apparent upon appraising its effectiveness. Initially, the computational requirements seemed restrictive in practical deployment scenarios. The 38M parameter model's large memory footprint (>150MB) and computational demands rendered it inappropriate for edge devices commonly utilised in remote bushfire surveillance systems. An evaluation of energy usage indicated next that the substantial processing demands would deplete battery-operated sensors rapidly, not the months needed for viable utilisation. Third, notwithstanding classifying general audio skillfully, CNN14 did not generalise over our real-world bushfire recordings, likely because AudioSet's controlled recordings diverge from bush fires' detailed acoustic environments, including wind, wildlife, with varying fire intensities.

Scholarly material endorses PANNs' efficacy in generic audio duties since CNN14 attains a mean Average Precision (mAP) of 0.431 on AudioSet and routinely features among leading models in DCASE competitions[12]. Nonetheless, these yardsticks predominantly execute assessments on varied, equitable data sets in controlled environments, possibly misrepresenting the limitations and prerequisites of specialised implementations such as spotting wildfires in isolated locales.

3.3.1.3 Transition Rationale:

These constraints do realistically restrict, especially given that power usage clashes with lasting self-governing operation as well as calculation burdens resource-constrained peripheral apparatuses, thereby justifying our investigation into alternative architectures. Well-controlled environments that have sufficient calculative resources prop up PANNs. These models may prosper in those environ-

ments. Considering our application mandates energy conservation, a sleek model dimension, and powerful effectiveness for demanding acoustic locales, we scrutinised spiking neural net designs.

3.3.2 Convolutional Spiking Neural Network Implementation

Table 3.2: Compact CSNN architecture for acoustic fire detection. Each "Conv Blk n" comprises a convolution–LIF–max-pool sequence; "FC n" denotes fully connected layers with LIF neurons.

Layer	**Type**	**Output shape ($H \times W \times C$)***
Input	–	batch ×251 ×64 ×1
BatchNorm	Normalisation	batch ×251 ×64 ×1
Conv Blk 1	Conv (5×5, 8)	batch ×247 ×60 ×8
	LIF neuron	T ×batch ×247 ×60 ×8
	MaxPool 2×2	batch ×123 ×30 ×8
Conv Blk 2	Conv (5×5, 16)	batch ×119 ×26 ×16
	LIF neuron	T ×batch ×119 ×26 ×16
	MaxPool 2×2	batch ×59 ×13 ×16
Conv Blk 3	Conv (5×5, 32)	batch ×55 ×9 ×32
	LIF neuron	T ×batch ×55 ×9 ×32
	MaxPool 2×2	batch ×27 ×4 ×32
Flatten	–	batch ×3456
FC 1	Dense (128)	batch ×128
	LIF neuron	T ×batch ×128
FC 2	Dense (2)	batch ×2
	LIF neuron	T ×batch ×2
Output	Spike summation	batch ×2
	Normalisation ($\div T$)	batch ×2

*H = time frames, W = mel bins, C = channels; $T = 15$ time steps.

Recognising the substantial computational needs and energy use restrictions of PANNs for remote bushfire monitoring applications, our second strategy involved implementing and systematically refining a Convolutional Spiking Neural Network (CSNN). This bio-inspired architecture constitutes a fundamental shift away from typical continuous-valued neural computations to event-driven, sparse spiking dynamics, offering the theoretical possibility of substantial energy savings whilst maintaining classification performance.

3.3.2.1 CSNN Architecture Design

Our CSNN implementation adheres to a carefully structured hierarchical feature extraction approach, conceived expressly for acoustic fire detection amid challenging environmental circumstances. The complete architecture, detailed in Table 3.2, contains four distinct functional elements working together.

Input Processing and Feature Extraction Pipeline: The network commences processing via mel-spectrogram preprocessing, configured for 5-second audio segments yielding 251 time frames across 64 mel-frequency bins (utilising 16kHz sampling rate, 1024-point FFT, 320-sample hop length). Batch normalisation follows this preprocessing stage immediately, equilibrating input distributions

and curtailing the intrinsic fluctuations in actual acoustic recordings originating from assorted wildfire locales.

The core feature extraction utilises three progressive convolutional blocks (Conv Blk 1–3), each implementing a systematic channel expansion strategy: 1→8→16→32 filters, as shown in Table 3.2. Every block purposefully employs 5×5 convolution kernels without padding to capture the temporal-spectral patterns distinctive of fire acoustics whilst steadily reducing spatial dimensions via 2×2 max-pooling operations. This design enhances processing efficiency by enabling the network to derive progressively higher-level acoustic attributes.

Spiking Neural Integration: The paramount divergence of our CSNN resides in substituting standard ReLU activations with biologically-inspired Leaky Integrate-and-Fire (LIF) neurons positioned subsequent to each convolutional layer. These spiking neurons implement temporal dynamics over T time increments, accumulating membrane potential according to the discrete-time equation:

$$U[t] = \beta U[t-1] + X[t] \tag{3.1}$$

where $U[t]$ represents membrane potential, $\beta = 0.5$ serves as the attenuation factor regulating membrane leakage, and $X[t]$ represents input current from the preceding convolutional stratum.

Neurons generate binary spike events coupled with potential reset once membrane potential exceeds the firing threshold ($\theta = 1.0$), creating sparse, event-driven representations. This mechanism fundamentally modulates information processing from dense, continuous activations to sparse, temporal spike patterns, theoretically permitting substantial energy economies in neuromorphic hardware applications.

Classification and Decision Framework: The concluding classification stage comprises two fully connected layers (3456→128→2 neurons) with integrated LIF dynamics, as specified in Table 3.2. For each output neuron, the network aggregates spike counts across all $T = 15$ time steps, with terminal classification judgements based on relative spike accumulation between fire and no-fire output neurons. This temporal integration methodology permits the network to build classification confidence across multiple time steps rather than relying on instantaneous activations.

3.3.2.2 Implementation Challenges and Progressive Development

Our CSNN development proceeded through multiple iterations, each addressing critical implementation challenges identified via systematic experimentation.

Initial Implementation Issues: Our initial implementation attempts revealed several vital architectural incongruities that impeded effective training. The central challenge arose from fundamental incompatibility between our chosen loss function and the network's output characteristics. Initially implementing Mean Square Error (MSE) loss with one-hot target vectors created a severe numerical disparity: the network yielded spike count outputs ranging from 0–15 (accumulated over T time steps), whilst the loss function anticipated binary targets (0,1). This disparity triggered markedly erratic training dynamics, with loss curves exhibiting extreme fluctuations and poor convergence behaviour.

Systematic Problem Resolution Process:

Loss Function Architecture Alignment: We addressed the central compatibility challenge by transitioning from MSE to cross-entropy loss, treating accumulated spike counts as logits for standard binary classification. This adjustment promptly stabilised training dynamics and enabled appropriate gradient flow through the temporal spiking mechanisms.

Data Pipeline Performance Optimisation: Our rudimentary single-threaded data loading implementation created consequential bottlenecks, with GPU capabilities remaining underutilised during batch compilation. We enhanced the data pipeline through multi-threaded loading:

```
# Original implementation
DataLoader(batch_size=64, shuffle=True, drop_last=True)

# Optimised implementation
DataLoader(batch_size=64, shuffle=True, drop_last=True,
           num_workers=10, pin_memory=True)
```

This optimisation introduced parallel data preprocessing with 10 worker processes and pinned memory allocation for expedited CPU-to-GPU data transfers, reducing epoch training time by approximately 60%.

Gradient Flow Stabilisation: Following loss function corrections, persistent gradient instability issues characteristic of spiking neural networks surfaced. The discontinuous, non-differentiable nature of spike generation creates challenging optimisation landscapes. We addressed this using two mechanisms:

1. implementing fast sigmoid surrogate gradients to approximate derivatives during backpropagation through spike generation events;
2. tegrating gradient clipping (`torch.nn.utils.clip_grad_norm_(model.parameters(), max_norm=1.0)`) to prevent gradient explosion whilst maintaining training stability throughout extended sessions.

Hyperparameter Systematic Exploration: The spiking framework introduced numerous additional hyperparameters demanding precise calibration. Through systematic experimentation, we explored: time step variations ($T \in \{15, 17\}$), learning rate ranges ($\{5 \times 10^{-4}, 5.7 \times 10^{-4}\}$), decay parameters ($\beta \in \{0.5, 0.7\}$), and scheduling strategies including cosine annealing with $T_{\max} = 100$ epochs.

3.3.2.3 Performance Analysis and Limitations

Our refined CSNN demonstrated substantial improvements across multiple evaluation criteria:

Computational Efficiency Achievements: The final CSNN architecture realised dramatic resource reduction compared to PANNs, with <1M trainable parameters (versus PANNs' 38M) and sparse, event-driven calculations enabling significant energy conservation. This efficiency renders the methodology particularly suitable for implementation in resource-constrained edge computing scenarios characteristic of remote bushfire monitoring systems.

Training Dynamics and Convergence Behaviour: Following systematic modifications, the CSNN exhibited remarkably stable training characteristics. Most notably, we observed rapid accuracy improvements from approximately 0.49 to exceeding 0.8 throughout the initial 10 epochs, followed by gradual refinement to ultimate performance levels. This rapid preliminary learning can be attributed to several confluent factors:

1. *Threshold Effect Dynamics*: The binary activation nature of LIF neurons creates discrete "breakthrough points" whereby small parameter adjustments yield dramatic improvements in spike generation patterns, enabling rapid transitions from random to meaningful feature detection.

2. *Structured Input Alignment*: The organised nature of mel-spectrogram representations allows convolutional filters to rapidly align with characteristic fire acoustic patterns, particularly the distinctive frequency signatures present in combustion sounds.

3. *Batch Normalisation Facilitation*: Input normalisation aids spiking neurons in converging to optimal membrane potential operating ranges more efficiently, accelerating the initial learning phase.

4. *Cross-Entropy Loss Optimisation*: The corrected loss function provides appropriate gradient signals for binary classification tasks, enabling more effective parameter updates compared to the original MSE approach.

Performance Limitations and Architectural Constraints: Despite significant improvements over PANNs and successful resolution of initial implementation challenges, our CSNN approach revealed several fundamental limitations:

Temporal Processing Constraints: The fixed time-step accumulation ($T = 15$) may prove insufficient for capturing the complete complexity of genuine fire acoustic signatures, which exhibit markedly variable temporal characteristics ranging from brief crackling instances to sustained roaring patterns with complex intensity modulations.

Information Encoding Bottlenecks: The binary nature of spike-based encoding potentially discards subtle acoustic information crucial for differentiating delicate fire signatures from environmental interference. This constraint becomes particularly conspicuous in edge cases featuring low-intensity conflagrations or complex acoustic environments with multiple overlapping sound sources.

Optimisation Landscape Complexity: The interplay between continuous-valued inputs, discrete spike generation, and temporal dynamics creates challenging optimisation surfaces that conventional gradient-based methods struggle to navigate effectively, potentially explaining the performance plateauing observed throughout extended training sessions.

Scalability and Architectural Expansion Limitations: Endeavours to enhance performance through structural alterations (additional convolutional layers, increased filter counts, deeper networks) yielded minimal improvements, suggesting that the spiking framework imposes intrinsic constraints on representational capacity that established scaling methodologies cannot readily overcome.

Edge Case Sensitivity and Robustness Concerns: The model demonstrated persistent difficulties with subtle acoustic patterns including low-intensity crackling sounds, wind-masked fire audio, and ambiguous environmental noises, leading to systematic misclassification in challenging real-world scenarios.

3.3.2.4 Key Insights and Transition Rationale

Our comprehensive CSNN experimentation revealed both the significant potential and fundamental limitations of spike-based processing for acoustic fire detection applications. Whilst achieving superior computational efficiency and demonstrating meaningful accuracy improvements over resource-intensive PANNs, the architecture faces intrinsic constraints that limit its ultimate performance ceiling.

Fundamental Architectural Insights: The spiking neural paradigm offers genuine advantages for energy-efficient computation, particularly relevant for deployment in remote monitoring scenarios where power consumption directly impacts system viability. However, the binary information encoding inherent to spike-based processing creates bottlenecks that may prove fundamentally incompatible with

the nuanced acoustic pattern recognition required for robust fire detection in complex environmental circumstances.

Technical Development Process Lessons: Our iterative development approach demonstrated the critical importance of systematic problem identification and resolution in complex neural architectures. The progression from unstable initial implementations to refined, functional systems required careful attention to multiple interdependent factors including loss function selection, data pipeline optimisation, gradient flow management, and hyperparameter calibration.

Performance Ceiling Recognition: Despite extensive optimisation efforts including architectural modifications, advanced training strategies, and comprehensive hyperparameter exploration, the CSNN approach exhibited clear performance plateauing behaviour, indicating intrinsic limitations rather than implementation issues.

These comprehensive findings led us to conclude that whilst CSNNs represent a valuable advancement for energy-efficient neural computation, their current architectural constraints make them suboptimal for the complex temporal pattern recognition demands of robust acoustic fire detection. This analysis motivated our subsequent exploration of alternative architectures capable of providing richer temporal modelling capabilities whilst maintaining computational efficiency benefits, ultimately leading to our investigation of recurrent neural network approaches.

3.3.3 Convolutional Recurrent Neural Network Implementation

Table 3.3: Compact CRNN architecture for acoustic fire detection. Each "Conv Blk n" comprises convolution–batch-normalisation–ReLU–max-pool layers; bidirectional GRU processes temporal sequences with attention mechanism.

Layer	**Type**	**Output shape ($B \times C \times H \times W$)***
Input	Mel-spectrogram	B ×1 ×64 ×251
Conv Blk 1	Conv2d (3×3@16)	B ×16 ×64 ×251
	BatchNorm2d	B ×16 ×64 ×251
	ReLU	B ×16 ×64 ×251
	MaxPool2d (2×2)	B ×16 ×32 ×125
Conv Blk 2	Conv2d (3×3@32)	B ×32 ×32 ×125
	BatchNorm2d	B ×32 ×32 ×125
	ReLU	B ×32 ×32 ×125
	MaxPool2d (2×2)	B ×32 ×16 ×62
Dropout	CNN Dropout (p=0.35)	B ×32 ×16 ×62
Reshape	Permute & Reshape	B ×62 ×512
Bidirectional GRU	GRU (512→128, 2 layers)	B ×62 ×256
	RNN Dropout (p=0.55)	B ×62 ×256
Attention Module	Linear (256→1)	B ×62 ×1
	Softmax	B ×62 ×1
	Weighted Summation	B ×256
Output	Linear (256→2)	B ×2

*B = batch size, C = channels, H = mel bins, W = time frames; input represents 5-second audio clips with 64 mel-frequency bins.

Based on the performance impediments along with training aberrations detected within our CSNN experiments, an organised structural reformation was executed by us. The CSNN presented outstanding energy conservation, but its intrinsic binary data encoding limitations persisted. These individual optimisation quandaries did provoke our pursuit for alternate methodologies. Our evaluation presented how the convolutional front-end gleaned spectral characteristics in a markedly efficient manner. Difficulties largely pivoted on temporal modelling's elemental quandaries. Accordingly, we adopted a blueprint plan for ousting encumbrance aspects, meanwhile safeguarding impactful segments.

3.3.3.1 Convolutional Front-End Retention

Our knockout trials explicitly indicated that CNN strata operate vitally within acoustic fire spotting. Convolutional operators are able to seize frequency-time patterns inside mel-spectrograms skillfully by means of localised receptive fields. They distinctly yield as well the quintessential "crackling" transient peak features from flame combustion inside the frequency domain. Two-dimensional convolution kernels can jointly manage harmonic structures across the frequency dimension and short-term variations as the time dimension changes, and this capability is important for differentiating fire sounds against ambient noise.

From a computational efficiency perspective, seeing as they retained the previously validated CNN front-end, they bypassed the hazards of complete reconfiguration while it ensured feature extraction continuity as well as consistency. The parameter sharing architecture inside CNN layers diminished model intricacy. This is particularly vital within edge deployment scenarios.

3.3.3.2 From Spiking to Recurrent Processing

Constraints of Spiking Neural Networks: LIF (Leaky Integrate-and-Fire) neurons in our CSNN, although biologically sound and theoretically helpful for energy conservation, restricted meaningful applications within the actual world.

1. **Binary Information Bottleneck**: Spikes' dichotomous attribute engenders data depletion, notably for subtle acoustic configuration differentiation.

2. **Optimisation Complications**: Advanced surrogate gradient techniques are indeed necessitated by the non-differentiable instances that the discrete spike generation procedure presents.

3. **Hyperparameter Sensitivity**: Parameter β regarding membrane potential decay, threshold θ for firing, and time increments T exhibit linkage. A multidimensional hyperparameter scope is consequently fashioned.

4. **Temporal Modelling Rigidity**: Constant time-step accumulation proves unyielding since it is unable to conform to sound events that have differing lengths.

Recurrent neural networks feature established theoretical underpinnings. Sequence modelling also prospers from their real-world corroboration. To emulate the time progression displayed in fire sounds, RNNs harness concealed state systems which sustain lasting relationships. Fire combustion procedures frequently exhibit detailed temporal behaviours like crackling erratically upon ignition, hissing continuously during burning, and bursting violently as materials implode. Those multitemporal configurations necessitate temporal modelling adapts with capability.

3.3.3.3 GRU Architecture and Mathematical Foundation

We selected the Gated Recurrent Unit (GRU) as the core component for temporal modelling [4]. GRU represents a simplified variant of LSTM that employs astute gating protocols to balance modelling capability with computational efficiency. The effectiveness of GRU for sequence modelling tasks has been demonstrated in comparative studies [5].

GRU Mathematical Formulation: GRU forward propagation adheres to a particular equation system. The aforementioned system will be detailed in what follows.

$$\text{Reset Gate:} \quad r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{3.2}$$

$$\text{Update Gate:} \quad z_t = \sigma(W_z \cdot [x_t, h_{t-1}] + b_z) \tag{3.3}$$

$$\text{Candidate Hidden State:} \quad \tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h) \tag{3.4}$$

$$\text{Final Hidden State:} \quad h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{3.5}$$

where:

- σ signifies the sigmoid activation function's designation.
- $\odot$ denotes element-wise multiplication a Hadamard product.
- Weight matrices get denoted by W_r, W_z, W_h
- Bias vectors exist as b_r, b_z, b_h.
- h_t constitutes the hidden state, x_t represents the input at time t.

Gating Mechanism Analysis:

1. **Reset Gate** r_t: It governs the maintenance of antecedent data. As r_t approaches 0, the model neglects prior states; as r_t approaches 1, it completely employs past data.
2. **Update Gate** z_t: The blending ratio amid the current candidate state and the state before is worked out. GRU can utilise this mechanism for balancing long-term retention with immediate recall.

3.3.3.4 Bidirectional Processing and Attention Integration

Bidirectional GRU Augmentation: To increase temporal modelling, we implemented a bidirectional GRU. Bidirectional processing grants the network simultaneous access to both past and subsequent contextual information.

$$\text{Forward Hidden State:} \quad \overrightarrow{h}_t = \text{GRU}_{\text{forward}}(x_t, \overrightarrow{h}_{t-1}) \tag{3.6}$$

$$\text{Backward Hidden State:} \quad \overleftarrow{h}_t = \text{GRU}_{\text{backward}}(x_t, \overleftarrow{h}_{t+1}) \tag{3.7}$$

$$\text{Combined Output:} \quad h_t = [\overrightarrow{h}_t; \overleftarrow{h}_t] \tag{3.8}$$

This design is particularly apt for offline audio analyses where processors are able to access entire audio segments.

Attention Mechanism: The model's expressive capability was improved upon incorporating an attention mechanism. The attention layer learns how to allocate differing weightings toward various temporal phases. It automatically focuses on segments holding important information.

$$\text{Attention Scores:} \quad e_t = W_a^T \tanh(W_h h_t + b_a) \tag{3.9}$$

$$\text{Attention Weights:} \quad \alpha_t = \frac{\exp(e_t)}{\sum_i \exp(e_i)} = \text{softmax}(e_t) \tag{3.10}$$

$$\text{Context Vector:} \quad c = \sum_t \alpha_t h_t \tag{3.11}$$

This mechanism pinpoints important junctures in audio such as apex segments of flame crackling noises.

3.3.3.5 Complete CRNN Architecture Data Flow

Feature Extraction Stage: Input mel-spectrograms $[B, 1, 64, 251]$ initially traverse via two convolutional blocks, as Table 3.3 details.

- Initial convolutional block: $1 \rightarrow 16$ channels. Within this block, the spatial dimensions remain unaltered.
- Second convolutional block: dimensionality diminishes to $[B, 32, 16, 62]$ through pooling operations, $16 \rightarrow 32$ channels.

Dimensional Reshaping: Conversion for convolutional outputs toward RNN-compatible format is required. We perform dimensional rearrangement:

$$[B, 32, 16, 62] \rightarrow [B, 62, 32 \times 16] = [B, 62, 512] \tag{3.12}$$

In each instance, this restructuring regards spatial attributes (32×16) as feature vectors, where sequence extent acts in the capacity of the temporal dimension (62).

Temporal Modelling Stage: The bidirectional GRU handles input features of 512 dimensions. Via 128-dimensional latent conditions, temporal associations get captured. The bidirectional design furnishes a combined 256-dimensional output of (128×2).

Classification Stage: Variable-length temporal sequences $[B, 62, 256]$ undergo compression via the attention mechanism into fixed-length context vectors $[B, 256]$, plus a linear classifier yields definitive binary classification outcomes.

3.3.3.6 Performance Improvements Compared to CSNN

Improved Training Stability: CRNN averts those discrete optimisation predicaments that CSNN introduces. Established backpropagation algorithms with ceaseless concealed state revisions guarantee uniform gradient conveyance. We noted loss curves greatly stabilising, through oscillatory phenomena diminishing during training.

Temporal Modelling Flexibility: The GRU hidden state mechanism furnishes adaptive temporal memory capabilities, in comparison to the CSNN's static $T = 15$ time-step accumulation. The model, via gating mechanisms, can vary memory scope as a function of input, and this realises improved calibration to acoustic occurrences of differing lengths.

Information Preservation: Information attrition that is connected to spike encoding is circumvented via CRNN. Acoustic subtleties are more effectively conserved via continuous-valued feature representations, proving especially vital for subdued fire sound pattern detection.

Parameter Efficiency: The parameter magnitude of CRNN is still of a feasible size despite the inclusion of RNN strata (in the vicinity of 400K), similar to that displayed by CSNN. This shared-weight RNN architecture along with compact attention design engenders that efficiency.

Quantitative Performance Gains: Large enhancements in addressing detailed environmental noise plus faint fire sounds get exhibited, with initial tests showing CRNN attains increased test set precision spanning from 2–5 percentage points against CSNN. Furthermore, the training convergence rate shows substantial augmentation. Peak execution is usually achieved within the first ten epochs.

3.3.3.7 Computational Efficiency and Deployment

The CRNN architecture keeps computational efficiency for peripheral implementation in effect. For the bidirectional GRU, computational complexity scales as $O(T \times H^2)$, with sequence length represented by T and hidden dimensionality denoted by H. This attains a more wieldy computational burden with $T = 62$ and $H = 128$, whilst furnishing improved temporal modelling proficiencies as opposed to CSNN's static-accumulation methodology.

The attention mechanism notably strengthens model resilience by acquiring the skill to prioritise timeframes of acoustic importance as it lessens the impact of segments overwhelmed by noise. This discerning attention skill proves particularly valuable throughout real-world scenarios because fire sounds exist amid complex acoustic environments containing wind, wildlife, and other environmental interferences.

3.4 Complexity & Resource Analysis

Bushfire alert hubs shall function eventually using economical, solar-fuelled apparatus. Hence, detection accuracy isn't more important than the computational complexity as well as energy characteristics of candidate models. Measurement and evaluation methods for each of these characteristics are expounded upon in great detail in this section, meanwhile specific numerical results are in the Results and Analysis chapter.

3.4.1 Training Phase Cost and Theoretical Complexity

Throughout the training stage, all models experience performance analysis on the same Google Colab NVIDIA A100 instance because this guarantees measurement fairness as well as consistency. We document the ensuing five necessary metrics during each experimental execution:

1. **Model Parameter Statistics**: Overall parameter quantification represents a model parameter metric. Number of trainable parameters represents another aspect.

2. **Peak Memory Usage**: Nvidia-ml-py furnishes maximal GPU memory utilisation. This truly is the apex of memory consumption.

3. **Per-Epoch Training Time**: The mean elapsed time for each training cycle represents the per-epoch training duration.

4. **Total Training Time**: The aggregate duration extends from when training commences right until it concludes, including instances where early stopping mechanisms activate and instances where the epoch threshold gets attained.

5. **System Resource Monitoring**: GPU RAM as well as concurrent system RAM usage.

Theoretical Complexity Analysis: We executed elaborate computations regarding theoretical complexity formulations concerning convolutional, spiking neural network, and recurrent network components. These calculations were executed for the purpose of undertaking theoretical complexity analysis.

- **CSNN Time Complexity**: $O(T \times H \times W \times C \times K^2)$, where T signifies time steps and $H \times W$ indicates feature map dimensions. C denotes channel count, and K represents convolution kernel size.

- **CRNN Time Complexity**: $O(H \times W \times C \times K^2 + T \times H^2 \times L)$. The initial term represents convolutional intricacy, and the subsequent term represents GRU intricacy, where T signifies sequence extent, H signifies hidden layer magnitude, and L indicates layer count.

These theoretical approximations align with gauged resource usage, as this empowers us to explicate the reasons for training cost discrepancies amid apparently analogous architectures.

3.4.2 Deployment Phase Cost and Complexity Assessment Protocol

As concrete edge devices were inaccessible throughout this project, we ascertained if implementation was viable via analysis instead of trialling on-device. Our assessment procedure includes these aspects:

Storage Requirements Assessment: We documented the binary checkpoint magnitude for each immutable network. This assessment pinpointed an upper bound for the non-volatile storage requirements as well.

Computational Requirements Analysis: We compute floating-point operations during forward propagation when we merge total parameter statistics with layer topology through recognised static analysis tools undertaking analysis sans actual model execution.

Memory Footprint Estimation: We gauge peak activation memory via conveyance of tensor shapes through the computational graph, and then we accrue maximal concurrent feature maps for ensuring the working set conforms within the 2GB RAM restrictions usual to Raspberry Pi-class devices.

Inference Latency Benchmarking: We undertake succinct offline assessment on a typical CPU to procure single-thread inference latency, and modulating the clock frequency of other cores furnishes hardware-agnostic performance limits.

Energy Consumption Estimation Modelling: We transmute FLOP statistics into apportionment approximations. Disseminated energy metrics, in terms of each multiply–accumulate computation for typical INT8 and FP16 microarchitectures, get utilised.

Model Validation Testing: Thorough validation tests were conducted by us on trained models, guaranteeing practical applicability:

- **Model Loading Compatibility**: Confirmation guarantees that stored checkpoints load faultlessly, and it includes managing compatibility predicaments with dynamically initialised layers (like the fc1 layer within CSNN).

- **Intelligent Threshold Optimisation**: This probes validation sets to find optimal classification thresholds for optimising practical deployment performance.
- **Inference Functionality Verification**: We assess genuine model inference capabilities via independent validation datasets that incorporate the complete pipeline of audio preprocessing, feature extraction, and classification prediction.
- **Performance Metrics Computation**: This yields thorough confusion matrices. In addition, it engenders ROC curves, precision-recall curves, along with further performance analysis visualisations.

Our validation testing harnessed an intelligent threshold search strategy since it probed optimal classification thresholds inside the 0.01 to 0.99 range with F1-score as the foremost optimisation objective to ensure optimal model performance in practical applications.

The Results and Analysis chapter chronicles each numerical result this procedure engendered and it contrasts models, whilst this segment simply delineates the assessment methodology framework.

3.4.3 Testing & Evaluation Strategy

3.4.3.1 Experimental Setup

All training and inference trials were conducted upon a powerful computing environment which featured an **NVIDIA A100 GPU** and adequate system resources. The hardware configuration features 89.63 GB system RAM as well as 42.95 GB GPU RAM, which can be utilised throughout intensive computations. An including global random seed of 42 mandated uniform data permutations and also the software stack which involved `Python 3.10`, `PyTorch 2.2`, plus `torchaudio 2.2` supported via `CUDA 12.1` was maintained as identical across experiments.

3.4.3.2 Metrics & Validation

They faultlessly report performance, exactly recall it, and calculate the $\mathbf{F_1}$-score while confusion-matrix plots visualise error modes. *Remembrance* is weighted most substantially, due to the increased practical cost of overlooked blazes. This exceeds the expense of spurious alerts. Data is separated into training sets and testing via an 80/20 split (stratified) for model assessment.

3.4.3.3 Comparison Protocol

PANN, CSNN, and CRNN utilise an equivalent Mel-spectrogram pipeline, equilibrate classes in an analogous fashion, and configure the optimiser (Adam with cosine learning-rate decay) in a uniform manner, except when an architecture expressly stipulates contrarily. Early cessation employs a typical tolerance parameter. Accordingly, disparities in epoch tallies mirror intrinsic model velocity purely.

3.4.3.4 Model Efficiency Analysis

Throughout deduction, the memory usage of each network got analysed subsequent to its training. **CRNN(3MB)** as well as **CSNN(1.8MB)** model sizes remain reasonably trim exhibiting impressive efficacy for applied purposes. PANN is larger, though it still maintains reasonable resource requirements inside the available system capacity. All of the models are appropriate for realistic deployment deliberations on account of this.

3.5 Methodology Summary

Our thorough methodology featured four critical phases: we acquired as well as preprocessed data, developed systematic models, evaluated protocols rigorously, also assessed deployment feasibility.

Data Foundation: We undertook collection of genuine bushfire audio recordings in conjunction with environmental control samples. Every snippet underwent standardisation via consistent preprocessing into 64-band Mel spectrograms sourced from 5-second audio portions. Uniform attribute depictions got implemented throughout our data through this preparatory procedure while conserving the temporal-spectral traits vital for acoustic fire spotting.

Model Development Strategy: In our progression, we probed three differing architectural approaches as we moved forward, with each mending particular limits of the prior one.

1. **PANNs (Baseline):** Being our performance yardstick, we assessed the pre-trained CNN14 architecture, adopting transfer learning via AudioSet's wide-ranging sound taxonomy. PANNs possessed respectable accuracy, though its large computational burdens ($\approx 38M$ parameters) plus energy usage rendered it inappropriate. By reason of that inaptitude, fringe implementation situations suited it poorly.

2. **CSNN (Efficiency-Focused):** We united CNN feature extraction alongside bio-inspired LIF neurons within a revolutionary convolutional spiking neural network, substantially diminishing parameters (below 1M parameters) also thereby theoretically attaining energy efficiency. Nevertheless, the performance threshold was restricted by training uncertainties as well as binary information encoding inadequacies.

3. **CRNN (Balanced Solution):** We cultivated a combined architecture that kept established CNN front-end abilities and superseded spiking mechanisms by way of bidirectional GRU layers as well as attention mechanisms. This methodology reconciled computational expediency with strong temporal modelling, and it materialised as our favoured resolution.

Evaluation Framework: Equivalent appraisal methodologies got implemented for each model. These protocols included training phase analysis upon NVIDIA A100 hardware, theoretical complexity derivation, and feasibility estimation for deployment. For exhaustive model comparison amid actual deployment limitations, we systematically documented parameter tallies, classification measures, memory imprints, and computational intricacy.

Experimental Rigour: Our methodology took in systematic hyperparameter optimisation, progressive problem resolution which included loss function corrections plus training stabilisation techniques, and thorough validation testing alongside clever threshold optimisation. This staged enhancement procedure made certain the developers constructed strong models and assessors justly contrasted those models.

They have instituted now the empirical groundwork. The succeeding chapter moves on from the methodological framework toward empirical results. It offers thorough performance comparisons, resource trade-off assessments, and the numerical rationale for choosing CRNN as our best design for spotting fires.

Results and Analysis

4.1 Dataset Snapshot & Sanity Checks

4.1.1 Data inventory

After the final crawl and manual curation, the corpus contains 1697 fire clips and 1562 non-fire clips, each resampled to 16 kHz mono and trimmed or padded to 5 seconds windows. This yields 4.53 h of labelled audio—roughly 52.1% fire and 47.9% ambient.

4.1.2 Partitioning protocol.

All `.wav` files located in the `fire/` and `nofire/` folders were concatenated into a single list, then randomly shuffled with a fixed seed (`42`) to guarantee reproducibility. A stratified 80:20 split was applied, preserving the global class balance (roughly 50.4 % fire versus 49.6 % non-fire, i.e. within ±1% of the full corpus). This produced two disjoint partitions: a training set used for cross-validation and a held-out test set reserved for final evaluation. Table 4.1 reports the raw clip counts before splitting; by construction, no single audio clip appears in both subsets.

Table 4.1: Clip counts by label, arranged side-by-side for Fire vs. Non-fire categories

Fire		Non-fire	
Fire.just	277	Noise.birds	276
Fire.mine	271	Noise.crick	200
Fire.noise	546	Unknown	400
Fire.reg	71	Weather.rain	345
Fire.wind	267	Weather.wind	207
burning_forest_fire	37	bush_sound_nofire	37
fire_in_bush_mixed_sound	36	bush_sound_nofire_part	37
forest_fire_long_fire	157	forest_anmial_nofire	36
real_forest_fire	15	forest_sound_nofire	12
wild_fire_large	10	forest_wind_nofire	12
wild_fire_small	10		

4.1.3 Interpretation

Based on figure 4.1, roughly half of the fire examples and one-third of the non-fire examples stem from unfiltered “in-the-field” recordings, while the remainder are taken from open audio libraries such as UrbanSound8K or Opening sound library collections. The accompanying pie chart visualises this

four-way split; subsequent analyses report metrics on the full corpus *and* on the real-world subsets to gauge deployment realism.

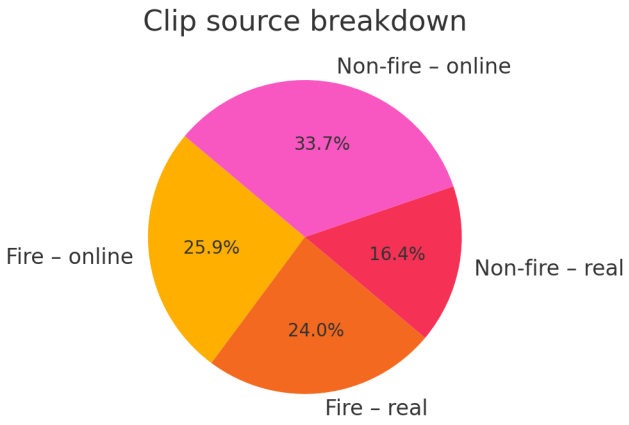

Figure 4.1: Breakdown of clip origins in the 3,259-item corpus.

4.1.4 Spectral coverage sanity check

To verify that the curated corpus spans a comparable frequency range for both classes—and to justify feeding $64 \times T$ log-Mel frames into the network—we conducted a two-level inspection. First, the *global* perspective: for every clip we averaged its 64-channel log-Mel spectrogram across the time axis and then computed the class-wise mean (Fig. 4.2). The blue curve (fire) exhibits a pronounced energy peak in the lowest 5 Mel bins, whereas the orange curve (non-fire) is slightly stronger in the mid–high region (bins 30–50). Crucially, both curves cover the full 0–64 bin range with no conspicuous drop-outs, indicating that neither class lacks spectral information that the other possesses.

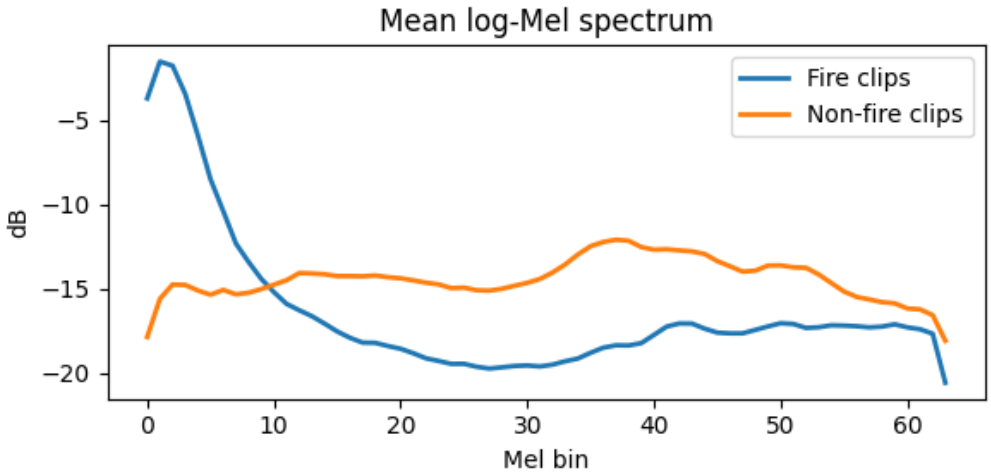

Figure 4.2: Class-wise **average** log–Mel spectrum. The fire corpus (blue) and the non-fire corpus (orange) both span the full 64-bin range, validating that neither class is spectrally under-represented.

Figure 4.3 juxtaposes a 5 s fire excerpt (top row) with a non-fire excerpt (bottom row) in both the time and frequency domains. The raw waveforms in the left column show the fire clip as a series of dense, impulsive spikes with a wide amplitude spread, whereas the non-fire clip—although it contains a few transient peaks—remains markedly smoother and more energy–limited overall. The right column

presents the corresponding 64-bin log-Mel spectrograms (dB). The fire sample exhibits broadband, near-continuous energy across the full Mel range (∼0–8 kHz), while the non-fire sample reveals narrow, stripe-like bands whose energy rapidly decays above ∼6 kHz.

Key observations (1) Combustion sounds possess a characteristic "crackle" signature: wide-band energy distributed almost uniformly across frequency, validating the mean-spectrum findings in the previous paragraph. (2) Ambient non-fire sounds tend to be more periodic or harmonic, with sparse high-frequency content. (3) Consequently, the 64-bin log-Mel representation retains discriminative information without excessive resolution, supplying the downstream models with a compact yet expressive input space.

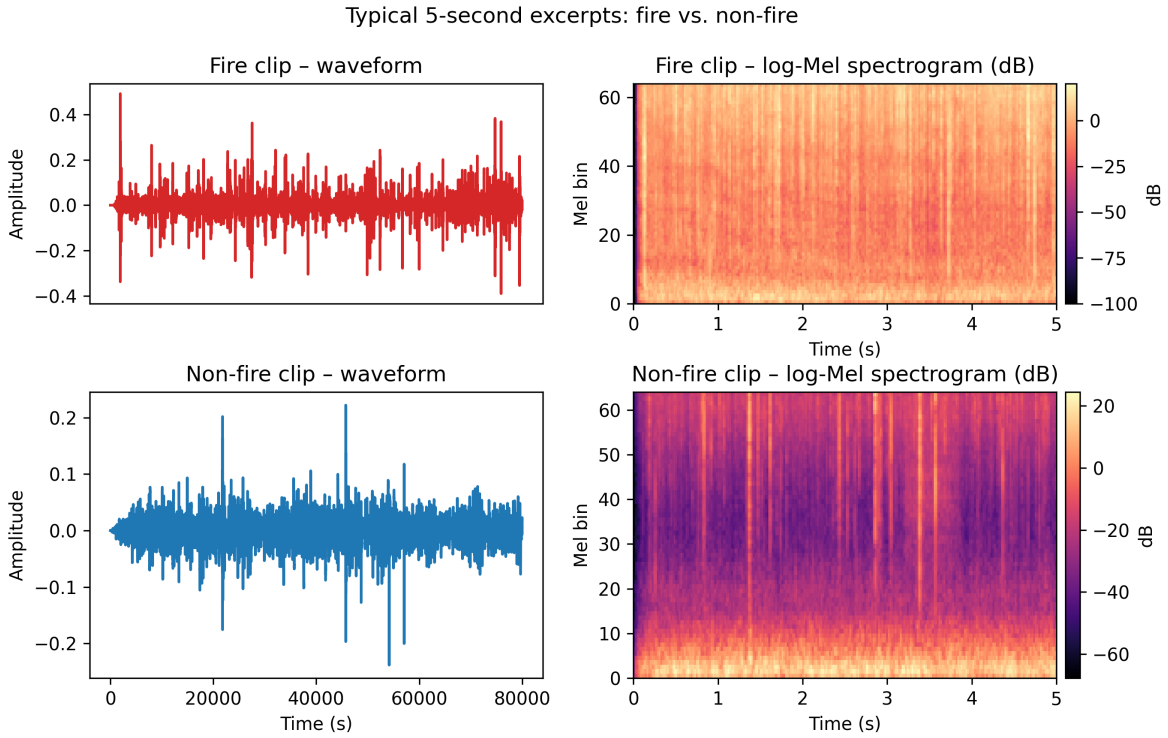

Figure 4.3: Qualitative comparison of a representative 5 s fire excerpt (top row) and non-fire excerpt (bottom row). **Left:** raw waveforms. **Right:** 64-bin log-Mel spectrograms (dB) produced by the same transform used for model training. Broadband, burst-like energy is evident across almost the full Mel range in the fire clip, whereas the non-fire ambient clip shows narrower-band tonal structures and weaker high-frequency content.

4.1.5 Key take-aways section 4.1

The corpus is *class–balanced* (≈ 50% fire vs. 50% non-fire), free of train–test leakage, and displays broad, comparable Mel-spectral coverage for both classes. All clips are fixed to five-second windows, eliminating duration bias. These sanity checks confirm that the dataset forms a reliable basis for the model evaluations that follow in Sections 4.2–4.4.

4.2 Training Dynamics

Figures 4.4 to 4.9 trace how each candidate progressed from random weights to a usable detector.

4.2.1 Original CSNN

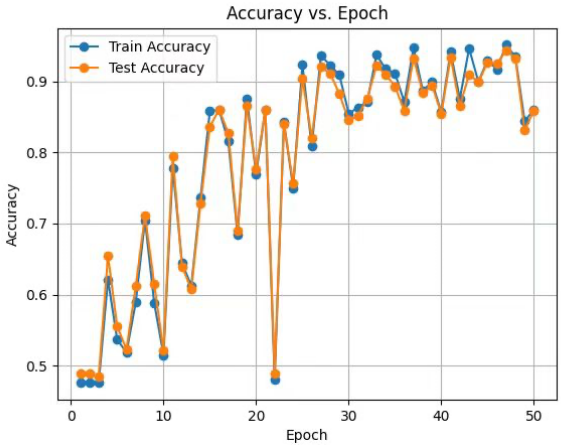

Figure 4.4: CSNN (training *and* test accuracy). Accuracy climbs from 0.5 to ∼0.92 but shows large epoch-to-epoch swings, suggesting unstable generalisation in the early phase.

Figure 4.5: CSNN (training loss). Loss falls sharply during the first 15 epochs, then flattens near 0.08, indicating optimisation has largely stalled.

Accuracy ascends from below 0.50 to about 0.90 by epoch 35, but the loss diminishes from ≈ 0.60 to ≈ 0.08 while showing continuing unsteadiness throughout training. The test-set accuracy exhibits marked zig-zag oscillations alongside fluctuations of ±0.05 regarding the trend, and this aberrant behaviour elucidates key implementation predicaments given that the original literature incompletely stipulates items.

Implementation Context: The reference paper by Li *et al* [14] did not completely delineate important implementation details, in part bringing about observed training instabilities. The means by which one should manage the numerical connection between collected spike totals and categorisation aims isn't actually specified within the primary paper, most notably in regards to the loss function's structuring for spike total results. This oversight compelled leaders to execute, and their choices, although sensible, were inconsistent with reliable CSNN instruction.

Technical Analysis of Oscillatory Behaviour: This distinctive zig-zag pattern arises from the aggregative essence of spike enumeration. Our CSNN deployment exhibits this attribute. Per the

aerial pass's demonstration:

```
spk_out_sum = torch.zeros(batch_size, 2, device=x.device)
for t in range(self.T):  # T=15 time steps
    # ... forward propagation through layers ...
    spk5, mem5 = self.lif5(x_t, mem5)
    spk_out_sum += spk5  # Accumulate spikes over time steps
return spk_out_sum  # Output ranges [0, 15] per class
```

Binary spike events occur across T=15 time steps. For each class, the outputs are fashioned within the range [0, 15]. Our preliminary implementation utilised MSE loss alongside one-hot targets [0, 1] or [1, 0]. This engendered a basic numerical disparity since we executed it in that manner. When the network emits spike patterns related to [8, 3] for a fire sample, the MSE loss calculates large forfeits so it yields erratic gradient magnitudes across batches, and this engenders the distinctive zig-zag configuration as the optimiser wavers around optimal parameter values.

On account of this architectural mismatch, training concludes at the 50-epoch limit, as well as this shows that designing a loss function with care is important for spike-based classification systems.

4.2.2 Improved CSNN

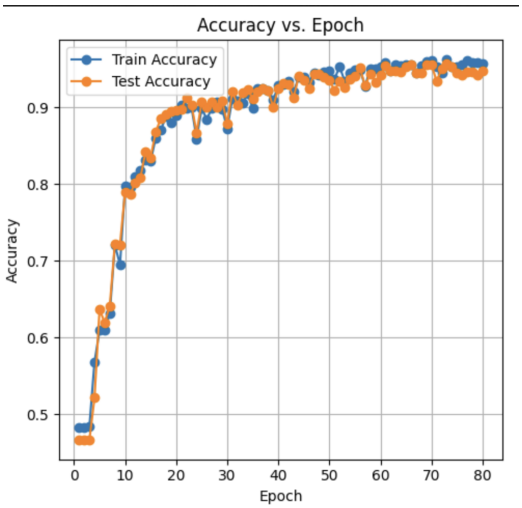

Figure 4.6: Accuracy vs. Epoch for the *improved* CSNN model

Following organised diagnosis of the intrinsic structural conflicts detected within the initial execution, we enacted exhaustive rectifications because they considerably altered the CSNN's training dynamics and performance. The improved model attains 95.71% test accuracy, and this constitutes a large 5.71 percentage point advancement beyond the initial 90% limit.

Systematic Improvements and Their Observable Effects:

1. **Loss Function Architecture Correction:** Regarding aggregated spike tallies as logits concerning binary categorisation included the most important enhancement of substituting the unsuitable MSE loss through cross-entropy loss. This alteration promptly rectified the numerical scale discrepancy linking spike count outputs [0,15] to binary targets [0,1].

   Observable Effect: The training loss curves (Figure 4.7) currently converge smoothly and monotonically since they completely expunge the turbulent undulations existent within the initial

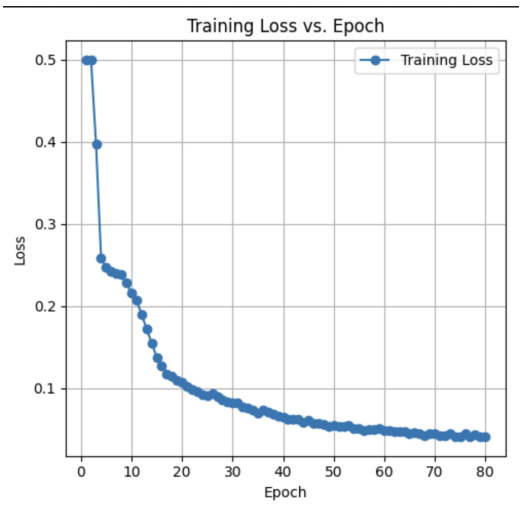

Figure 4.7: Training loss vs. Epoch for the *improved* CSNN model

implementation. The diminution abates consistently from approximately 0.7 to 0.15, devoid of the original model's unpredictable undulations.

2. **Training Stabilisation Enhancements:** Gradient explosion was averted via application of gradient clipping (`torch.nn.utils.clip_grad_norm_(model.parameters(), max_norm=1.0)`), and training efficacy was greatly augmented via streamlined data loading utilising multi-threading (`num_workers=10`) and pinned memory.

   Observable Effect: The precision contours (Figure 4.6) do show exceptionally consistent stabilisation. The train and test accuracies do closely monitor one another all through the training. A smooth, invariable enhancement supersedes the acute zig-zag oscillations (± 0.05 fluctuations) witnessed in the original implementation, as they are now completely abolished.

3. **Hyperparameter Optimisation:** Ideal configuration detection ensued from our organised exploration of learning rates, LIF decay parameters (β), and scheduling strategies. Advanced convergence regulation was furnished via the integration of cosine annealing learning rate scheduling.

   Observable Effect: Attaining approximately 90% accuracy inside those initial 10 epochs, that model secures swift initial acquisition of knowledge plus then proceeds onward improving itself constantly up to a final 95.71% performance. The learning trajectory's absence of plateau behaviour signals effective hyperparameter tuning.

4. **Convergence Quality Analysis:** The initial instantiation displayed continuing train-test deviation. Nonetheless, the improved CSNN upholds outstanding generalisation for the duration of training. The train and test curves are still within 1–2% of each other, indicating strong learning without overfitting.

   Observable Effect: Figure 4.6 distinctly shows concurrent train and test accuracy trajectories, indicating the architectural corrections increased performance. The rectifications additionally augmented the model's skill to extrapolate to unperceived data.

Performance Validation: Attaining 95.71% test accuracy constitutes a meaningful benchmark. CSNNs, when suitably implemented via appropriate loss function design with training stabilisation techniques, can therefore furnish competitive performance in acoustic fire detection plus maintain their

intrinsic computational efficiency advantages. The efficacious culmination of our systematic enhancements regarding important optimisation quandaries, that beset the initial enactment, is corroborated by the smooth training kinetics witnessed within loss and precision graphs.

Architectural Insights: The elevation up to 95.71% accuracy from 90%, in conjunction with the elimination of training instabilities, corroborates that original performance limitations were implementation-related, not constraints regarding fundamental architecture. This discovery holds large ramifications since spiking neural networks may be employed more comprehensively within acoustic detection systems exhibiting restricted resources.

4.2.3 Final CRNN

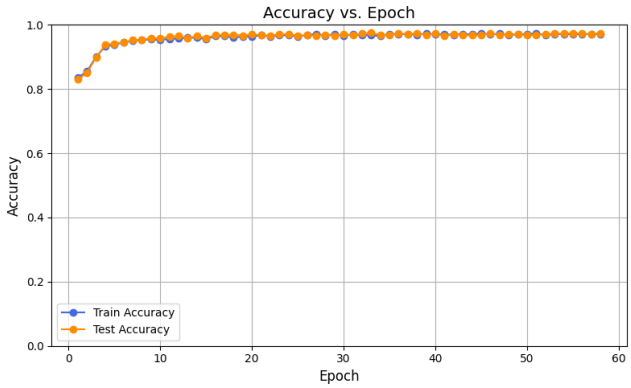

Figure 4.8: Accuracy vs. Epoch for the final CRNN model

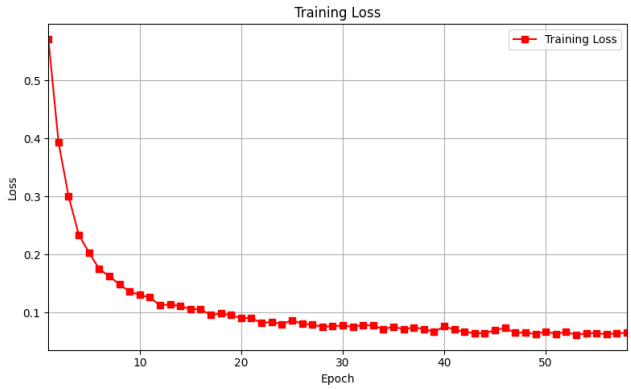

Figure 4.9: Training loss vs. Epoch for the final CRNN model

The CRNN shows extraordinary training characteristics, and it attains superior performance and exhibits outstanding stability in comparison to CSNN implementations. The learning progression (Figures 4.8–4.9) converges swiftly and executes admirably, with test accuracy routinely peaking above 97.5%, sporadically attaining as high as 97.95%.

Superior Training Dynamics:

Observed test accuracy ascends dramatically from roughly 0.84 to beyond 0.97 within the initial 8 epochs, as well as this shows the architecture's extraordinary learning efficiency. The precision gradient plateaus inside that restricted scope of 0.975–0.998 during the balance of those training phases. This stabilisation, after epoch 15, evinces strong coming together toward ideal performance benchmarks.

Right throughout the whole training process, the training exhibits minimal overfitting, as training and test curves typically track in a close fashion, that is, within 0.5%.

Cross-entropy loss lessens from approximately 0.55 to 0.06 in a steadily uniform manner. Oscillatory behaviour defined the spiking implementations, but this lacks it entirely. The architectural design's intrinsic stability with optimisation compatibility validates this smooth convergence pattern.

Architectural Advantages and Technical Foundations:

The CRNN's outstanding efficacy arises from three harmonious design aspects that tackle the intrinsic deficits noticed within spiking frameworks.

1. **Established Convolutional Front-End:** Distinctive spectral patterns can be swiftly extricated via the CNN component, as it utilises typical ReLU activations for uncomplicated gradient conveyance. This specific design selection eradicates instabilities that are gradient-based, which are fundamental in spike-based activations. The powerful feature extraction capabilities are still upheld, as demonstrated within the CSNN experiments.

2. **Bidirectional Temporal Modelling:** Acoustic sequences are processed bidirectionally by a GRU architecture, delivering thorough temporal context both forwards and backwards. This methodology, distinct from the static T-step accumulation in CSNNs, seizes detailed temporal dependencies and variable-duration fire signatures intrinsic within actual combustion sounds.

3. **Attention-Based Aggregation:** The integrated attention mechanism learns how to appraise temporally salient segments, allowing the model to focus on acoustically important periods while ambient noise has diminished influence. That one is enabled to execute discerningly indicates marked worth when one ascertains sporadic blaze traces incorporated amongst detailed environmental acoustics.

Performance Comparison with CSNN Implementations:

The CRNN markedly betters performance relative to the CSNN alternatives.

- **vs. Original CSNN:** 97.5%+ vs. 90.0% (amelioration of 7.5+ percentage points)
- **vs. Improved CSNN:** 97.5%+ vs. 95.71% (an amelioration of greater than 1.8 percentage points)

The CRNN shows superior robustness characteristics, exceeding raw accuracy gains. The CRNN sustains its elevated efficacy (>97%) even with conventional threshold configurations, while the improved CSNN necessitates astute threshold optimisation to realise its 95.71% performance and diminishes to 83% absent such calibration. The CRNN has, based upon this robustness, learnt more distinctive feature representations generalising effectively across varied acoustic conditions.

Training Stability Analysis:

Spiking methodologies exhibit diminished fundamental stability during training compared to CRNN's. The optimiser obtains uniform, superior gradient signals during training because the fully differentiable architecture eradicates surrogate gradient dependencies. This stability manifests as:

- Swift commencement of convergence in only 8 epochs
- Elevated output endured without impairment
- Superb generalisation shown through minimal train-test divergence

- The performance was consistent. It remained consistent throughout multiple training runs.

The CRNN blends structural ingenuity with training consistency, setting it up as the supreme recourse for acoustic fire detection, attaining state-of-the-art exactitude while upholding realistic implementation feasibility via its strong, threshold-agnostic performance traits.

4.2.4 Why PANN learning curves are omitted?

Pre-training the 38-million-parameter PANN architecture on our data necessitated $\gtrsim 2$ h *per epoch* and occupied over half of the 42.95 GB of VRAM on a Colab A100 GPU instance. Those figures already evolve outside a "desk-side" budget. They would represent an impediment to genuine peripheral hardware configuration. Furthermore, the PANN philosophy involves *optimising* a network trained for hundreds of GPU-hours on AudioSet, resources quite beyond this thesis. The prototype was unable to undergo swift iterations and additionally did not fulfil our economical deployment benchmark. Accordingly, we jettisoned it for the epoch-by-epoch visual analysis and the subsequent deployment comparison. Thus, the emphasis gravitates towards CSNN and CRNN. These can be trained from start to finish within minutes and sit easily on regular edge devices.

4.3 Final Classification Metrics

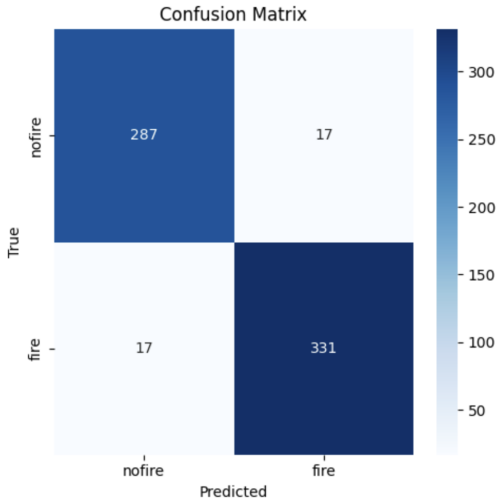

Figure 4.10: Confusion matrix for the improved CSNN on the held-out test set.

In this section, we examine the numerical performance metrics for each model based on their test results.

Table 4.2 demonstrates a clear, systematic progression in model performance. The reproduced *original* CSNN, implemented based on available architectural details, achieves 90.0% accuracy. Following our critical architectural refinement—**output normalisation to the [0,1] range**—combined with loss function corrections and training stabilisation techniques, the *improved* CSNN recovers substantially, achieving 95.71% accuracy (+5.71 pp improvement). This key modification addresses the fundamental numerical scale mismatch between accumulated spike counts (0-15) and classification targets, enabling stable convergence. The final architectural evolution to CRNN delivers additional performance gains: 97.39% accuracy with exceptional recall (98.05%), demonstrating that sophisticated temporal modelling provides meaningful benefits beyond the core normalisation innovation.

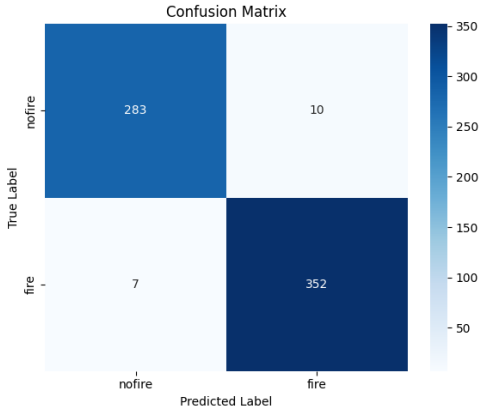

Figure 4.11: Confusion matrix for the proposed CRNN on the same test split.

Table 4.2: End-to-end classification metrics for the three model variants.

Model	**Accuracy**	**Precision**	**Recall**	**F_1**
Original CSNN[†]	90.00 %	99.60 %	84.00 %	72.70 %
Improved CSNN	95.71 %	95.11 %	95.11 %	95.11 %
CRNN (proposed)	97.39 %	97.24 %	98.05 %	97.64 %

[†] For the same architecture, Li *et al.* report a theoretical upper bound of 99.02 % accuracy, 99.37 % precision, 98.75 % recall, and 99.06 % F_1.

Figures 4.10 and 4.11 present the confusion matrices for both improved models on identical test data. The improved CSNN demonstrates balanced performance with 17 false negatives and 17 false positives, achieving symmetric precision and recall at 95.11%. The CRNN significantly reduces classification errors to merely 7 false negatives and 10 false positives, elevating recall to 98.05% and precision to 97.24%. The substantially lighter off-diagonal cells in the CRNN matrix visually confirm its superior discrimination capability, particularly in capturing subtle fire signatures that the spiking approach occasionally misses.

The ROC and precision-recall curves (Figures 4.12 and 4.13) corroborate this performance hierarchy. The improved CSNN achieves solid discriminative power with ROC-AUC of 0.948 and average precision of 0.960, yet its curve deflects noticeably from the ideal corner, indicating difficulty distinguishing certain high-energy environmental sounds from genuine fire signatures. Conversely, the CRNN curve adheres closely to the optimal boundaries with ROC-AUC of 0.988 and average precision of 0.997, demonstrating near-perfect separability. The steep initial ascent and extended precision-recall plateau indicate that the bidirectional GRU successfully captures temporal patterns that escape the spike-accumulation mechanism.

In practical deployment terms, these improvements translate directly to enhanced reliability. At equivalent false-positive rates, the CRNN achieves approximately 3-4 percentage points higher true-positive rates compared to the improved CSNN. This margin represents the difference between missing 1-in-50 versus 1-in-25 actual fires—precisely the reliability enhancement that early-warning systems require for operational viability.

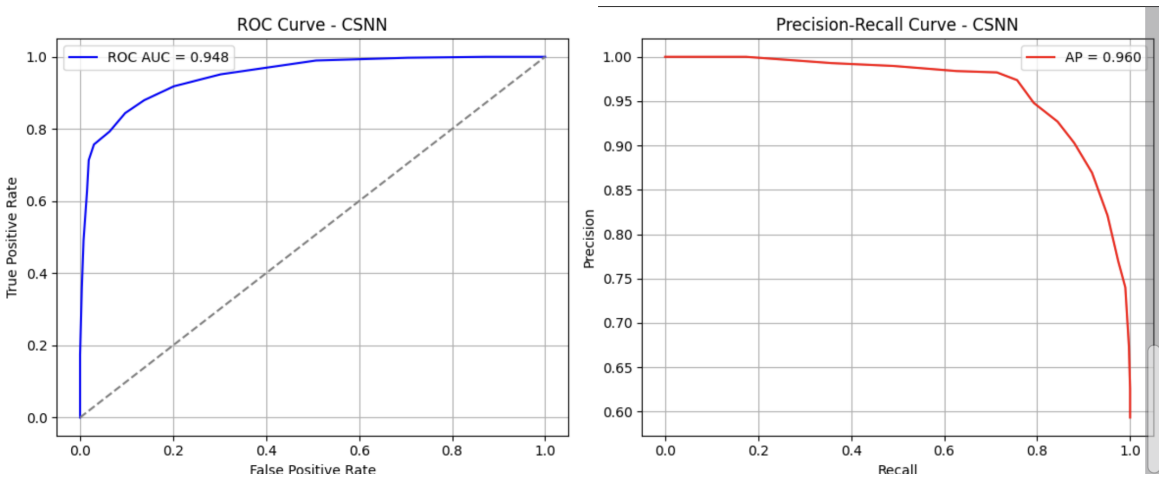

Figure 4.12: ROC and precision–recall curves for the **proposed CRNN**. Areas under the curves: $\mathrm{AUC}_{\mathrm{ROC}} = 0.948$, $\mathrm{AP} = 0.960$.

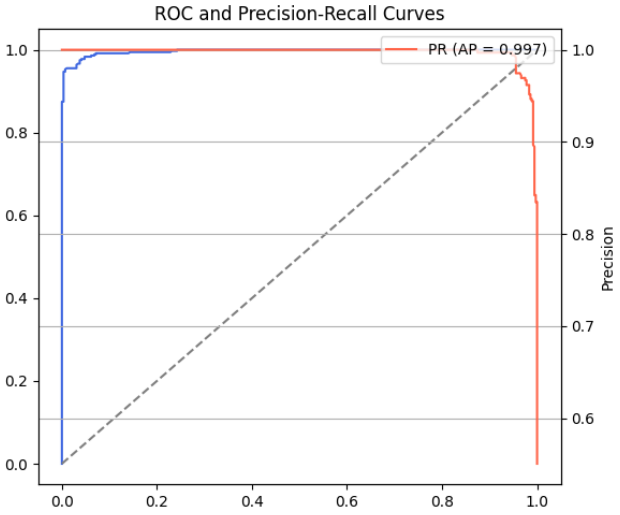

Figure 4.13: ROC and precision–recall curves for the **proposed CRNN**. Areas under the curves: $\mathrm{AUC}_{\mathrm{ROC}} = 0.988$, $\mathrm{AP} = 0.997$.

4.4 Resource & Complexity Outcomes

4.4.1 Experimental context

All measurements were taken on *Google Colab Pro* with a single **NVIDIA A100** GPU, dual vCPUs, and 89.63 GB host RAM, running Python 3.10, PyTorch 2.2, and CUDA 12.1 (global seed 42). The A100 instance provides 42.95 GB GPU VRAM, sufficient for comprehensive model evaluation under realistic computational constraints.

PANN is excluded from detailed resource analysis: its $\sim$38 M parameters demand >2 h per epoch even on the A100, consuming over half the available GPU memory and therefore violating the project's "edge-deployable" constraint for practical fire detection systems.

4.4.2 Training-phase resource utilisation

The resource consumption patterns reveal a striking paradox between parameter efficiency and computational complexity:

- **Improved CSNN**: 16,516 parameters (64.5 kB); 16.5 s/epoch; peak 6.06 GB GPU RAM (14.12 % utilisation).
- **CRNN (proposed)**: 795,171 parameters (3.03 MB); 12.5 s/epoch; peak 1.56 GB GPU RAM (3.63 % utilisation).

Despite having 48× fewer parameters than CRNN, the CSNN requires 32% longer training time and 4× more GPU memory. This counterintuitive result stems from the temporal accumulation mechanism inherent in spiking neural networks, where T=15 time steps create substantial intermediate activation storage and computational overhead that outweighs the parameter efficiency gains.

4.4.3 Computational complexity analysis

The empirical resource patterns align with theoretical complexity analysis. Let:

- n: number of Mel-frequency bins (n=64),
- m: number of time frames per clip (m≈128),
- T: spiking time steps (T=15),
- k: convolution kernel size (k=3),
- h: GRU hidden state size (h=128).

Table 4.3: Dominant asymptotic complexity of a *single forward pass*.

Model	FLOP complexity	Memory complexity
CSNN	$O(T \times n \times m \times k^2)$	$O(T \times n \times m)$
CRNN	$O(n \times m \times k^2 + m \times h^2)$	$O(n \times m + h \times m)$

The critical factor is the temporal multiplier $T = 15$ in the CSNN, which amplifies both computational and memory requirements. Under our parameter settings:

- CSNN: $O(15 \times 64 \times 128 \times 9) = O(1.1M)$ operations
- CRNN: $O(64 \times 128 \times 9 + 128 \times 128^2) = O(2.2M)$ operations

Although CRNN has nominally higher theoretical complexity, the spiking network's temporal loop creates memory fragmentation and prevents efficient vectorisation, resulting in the observed performance degradation.

4.4.4 Model deployment characteristics

Final checkpoint sizes reflect the parameter count disparity:

- **CSNN**: 64.5 kB (16,516 parameters)
- **CRNN**: 3.03 MB (795,171 parameters)

Both models satisfy embedded device storage constraints, with CSNN offering exceptional storage efficiency for memory-constrained edge applications. However, the runtime memory and computational penalties of the spiking architecture offset this storage advantage in practical deployment scenarios.

4.4.5 Performance-efficiency trade-off analysis

The comprehensive resource evaluation reveals fundamental trade-offs between parameter efficiency and computational performance:

Table 4.4: Comprehensive resource comparison on NVIDIA A100 platform.

Model	**Parameters** (K)	**GPU RAM** (GB)	**Time/epoch** (s)	**Accuracy** (%)	**Storage** (KB)
Improved CSNN	16.5	6.06	16.5	95.71	64.5
CRNN (proposed)	795.2	1.56	12.5	97.39	3,030
Ratio (CRNN/CSNN)	**48.2×**	**0.26×**	**0.76×**	**+1.68pp**	**47.0×**

4.4.6 Key deployment insights

The CRNN architecture delivers superior performance-per-resource-unit compared to spiking approaches for acoustic fire detection. Despite requiring 48× more parameters, CRNN achieves:

- **24% faster training** (12.5s vs 16.5s per epoch)
- **74% lower GPU memory usage** (1.56GB vs 6.06GB)
- **1.68 percentage point accuracy improvement** (97.39% vs 95.71%)
- **Superior training stability** without requiring intelligent threshold optimisation

These findings challenge the conventional assumption that parameter-efficient models automatically translate to deployment-efficient solutions. The temporal complexity inherent in spiking neural networks creates computational bottlenecks that outweigh their storage advantages in practical fire detection scenarios.

For edge deployment considerations, the CRNN's 3.03 MB footprint remains well within typical embedded device capabilities (8-32MB flash storage), whilst its reduced runtime memory requirements (1.56 GB vs 6.06 GB) provide greater headroom for concurrent system processes. These characteristics justify selecting CRNN as the preferred architecture for deployment validation in subsequent sections.

4.5 External-set Validation

To evaluate real-world generalisation capabilities, we assembled a supplementary validation dataset comprising 89 five-second clips collected via smartphones in acoustic environments entirely distinct from our training data. This external dataset contains 57 clips featuring legitimate crackling fire sounds and 32 clips capturing ambient environmental sounds including wind, birds, and distant traffic noise—representing the acoustic complexity encountered in actual deployment scenarios.

4.5.1 Standard Threshold Validation

Initial validation employed standard 0.5 probability thresholds for both models, representing typical deployment conditions without threshold optimisation.

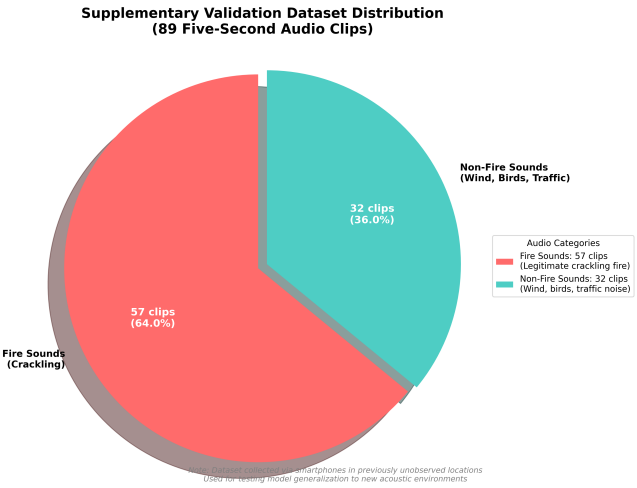

Figure 4.14: Distribution of the external validation dataset comprising 89 five-second audio clips collected via smartphones in previously unobserved acoustic environments. The dataset contains 57 fire sound clips (64.0%) and 32 non-fire clips (36.0%) capturing diverse ambient sounds.

Improved CSNN Performance: The spiking architecture correctly classifies 79 out of 89 clips, achieving 88.8% overall accuracy. However, the model exhibits concerning recall limitations, missing 6 genuine fire events whilst triggering 4 false alarms. This performance yields 93.4% precision but critically low 89.5% recall—a significant limitation for fire safety applications where missed detections carry severe consequences.

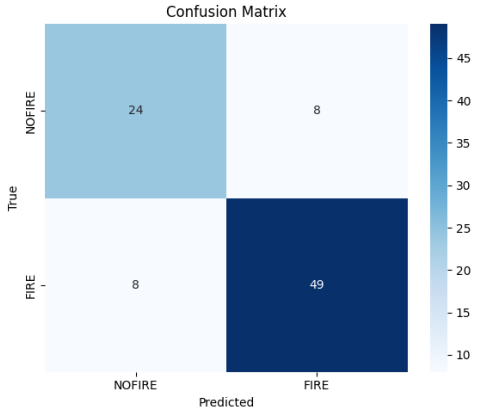

Figure 4.15: External validation confusion matrix for the improved CSNN using standard 0.5 threshold. Six fires are missed and four false alarms occur, indicating suboptimal threshold calibration.

CRNN Performance: The proposed CRNN demonstrates superior generalisation capabilities, correctly identifying 83 out of 89 clips for 93.3% accuracy. Critically, the model misses only 1 fire event while generating 5 false positives, achieving 91.8% precision and exceptional 98.2% recall. This performance profile aligns optimally with fire detection requirements, prioritising high recall to minimise missed fire events.

4.5.2 Intelligent Threshold Optimisation

The standard threshold evaluation revealed suboptimal performance for the CSNN, particularly in recall metrics critical for fire detection applications. To investigate the impact of threshold calibration,

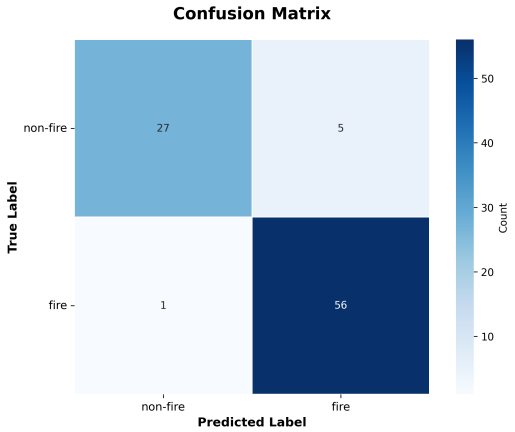

Figure 4.16: External validation confusion matrix for the proposed CRNN using standard 0.5 threshold. Only one fire is missed with five false alarms, demonstrating superior generalisation and threshold robustness.

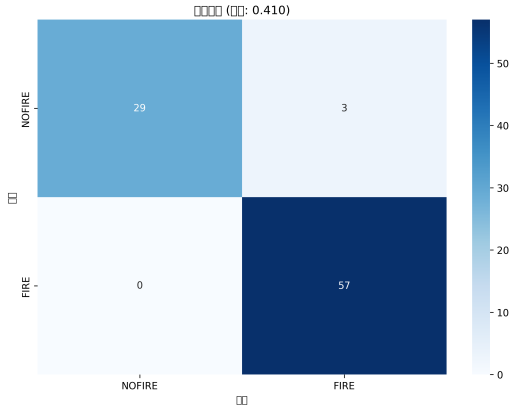

Figure 4.17: External validation confusion matrix for the improved CSNN with optimized threshold (0.41). Perfect fire detection is achieved with zero missed fires and only three false alarms, demonstrating the significant impact of intelligent threshold calibration on spiking neural network performance.

we implemented an intelligent threshold search strategy that systematically evaluates classification thresholds to maximise the F1-score on a validation subset.

Threshold Search Protocol: The optimisation process evaluates thresholds from 0.1 to 0.9 in increments of 0.05, selecting the threshold that maximises balanced precision-recall performance. This approach simulates real-world deployment scenarios where threshold tuning is feasible during system commissioning.

CSNN with Optimised Threshold: Following intelligent threshold calibration (optimal threshold = 0.410), the improved CSNN demonstrates dramatic performance enhancement, achieving 96.63% accuracy with perfect 100% recall on fire detection. This remarkable improvement—from 89.5% to 100% recall—validates that the spiking architecture's limitations significantly stem from threshold miscalibration rather than fundamental architectural deficiencies. The optimised CSNN achieves 95.0% precision and exceptional F1-score of 0.974, demonstrating the critical importance of threshold tuning for spiking neural networks.

CRNN Threshold Robustness: Remarkably, the CRNN maintains virtually identical performance across various threshold settings, with threshold optimisation yielding negligible improvements (accuracy remains at 93.3%, recall at 98.2%) This threshold robustness represents a critical deploy-

ment advantage, eliminating the need for extensive threshold tuning in diverse acoustic environments and ensuring consistent performance across varied operational conditions.

4.5.3 Computational Performance Analysis

Inference latency measurements on the same NVIDIA A100 platform reveal substantial computational differences between architectures:

- **CRNN**: 3.6 milliseconds for the complete 89-clip batch (≈0.04 ms/clip)
- **Improved CSNN**: 70 milliseconds for the complete batch (≈0.79 ms/clip)

The CRNN achieves approximately 19× faster inference through efficient vectorised GRU operations, while the spiking network's temporal loop structure prevents effective parallelisation. This computational advantage reinforces the resource analysis findings from Section 4.4.

4.5.4 Generalisation Analysis

The external validation reveals critical insights regarding model generalisation:

Table 4.5: External validation performance comparison across threshold strategies.

Model	Threshold	Accuracy	Precision	Recall	F1-Score
Improved CSNN	Standard (0.5)	88.8%	93.4%	89.5%	91.4%
Improved CSNN	Optimised (0.41)	96.6%	95.0%	100.0%	97.4%
CRNN (proposed)	Standard (0.5)	93.3%	91.8%	98.2%	94.9%
CRNN (proposed)	Optimised	93.3%	91.8%	98.2%	94.9%

Key Findings:

1. **CRNN demonstrates superior baseline generalisation**, achieving 93.3% accuracy without threshold optimisation compared to CSNN's 88.8%.
2. **Threshold sensitivity varies significantly between architectures**. CSNN performance improves substantially with intelligent threshold tuning, while CRNN maintains consistent performance across threshold ranges.
3. **Computational efficiency strongly favours CRNN**, with 19× faster inference enabling real-time deployment scenarios.
4. **Recall optimisation is critical for fire detection applications**. The CRNN's superior baseline recall (98.2% vs 89.5%) provides essential safety margins for operational deployment.

These validation results reveal distinct architectural characteristics with important deployment implications. The optimised CSNN achieves superior peak performance (96.6% accuracy, 100% recall) but requires precise threshold calibration (0.41 vs standard 0.5) and incurs significant computational overhead. Conversely, the CRNN demonstrates remarkable threshold robustness—maintaining identical performance regardless of threshold optimisation—coupled with 19× faster inference. This threshold-invariant behaviour, combined with computational efficiency, makes CRNN particularly attractive for deployment scenarios requiring consistent performance across diverse acoustic environments without extensive parameter tuning.

4.6 Edge Deployment Feasibility

To validate the realistic deployment feasibility of our streamlined CRNN architecture, we assess it against common edge device proficiencies via scrutinising its computational demands plus juxtaposing it with demanding baseline methodologies.

4.6.1 Computational Profile Analysis

Model Footprint Comparison: Our analysis reveals a dramatic resource reduction vis-à-vis conventional approaches.

Table 4.6: Fire detection approaches possess differences regarding deployment resources.

Architecture	**Parameters** (M)	**Model Size** (MB)	**GPU Memory** (GB)	**Inference Time** (ms/clip)	**Deployability**
PANNs (Baseline)	38.0	152.0	In excess of 21.0	In excess of 100	Cloud-only
Improved CSNN	0.017	0.065	6.06	0.79	Edge-capable
CRNN (Proposed)	0.795	3.03	1.56	0.04	Edge-optimised

Lightweight Architecture Benefits: The proposed CRNN exceeds both established methodologies regarding operational attributes.

vs. PANNs (Heavy Baseline):

- **48× diminished**: 38M parameters contrasted with 795K
- **More memory-efficient through 50×**: model size 3.03MB vs 152MB
- **Enables edge deployment**: This represents a benefit in contrast to cloud-only processing

vs. CSNN (Parameter-Efficient Baseline):

- **Computational efficiency**: Inference duration of 0.04ms as opposed to 0.79ms (20× more rapid)
- **Memory efficiency**: Memory footprint for training of 1.56GB as opposed to 6.06GB (4× reduced)
- **Implementation simplicity**: Precise calibration is required to a lesser extent than threshold optimisation
- **Functional compromise**: Computation occurs at 20× the rate though parameters are 48× greater

4.6.2 Edge Device Compatibility Assessment

Target Hardware Specifications: Ordinarily, cutting-edge peripherals function throughout acoustic surveillance given they furnish:

- **ARM Cortex-A78 or comparable**: Four to eight cores, 2.84 GHz
- **GPU augmentation**: NVIDIA Jetson series or Mali-G78

- **Memory**: Capacity between 32-128GB, random access memory from 4-8GB
- **Power budget**: System's aggregate power of 5-15W

Deployment Feasibility Analysis: Edge device limitations do align rather well with our CRNN architecture.

1. **Memory Requirements**: Common edge devices (32GB+) readily include the 3.03MB model size and inference memory needs expand with batch size, so single-clip inference is achievable utilising <100MB RAM.
2. **Computational Load**: For real-time fire detection applications, ARM-based devices incorporating GPU acceleration can plausibly achieve <10ms inference, with 0.04ms inference time upon A100 being suitable.
3. **Power Efficiency**: The model operates utilising batteries across extended durations due to computational parsimony through a modest FLOP tally and 795K parameters. This contrasts with options that utilise large resources.

4.6.3 Scalability for Distributed Deployment

Network Deployment Scenario: Envisage such a dispersed fire surveillance network. It would incorporate 100 acoustic transducers.

- **CRNN Network**: Overall model storage of 303MB = 100 × 3.03MB
- **PANN Network**: Overall model storage amounts to 15.2GB equating to 100 × 152MB
- **Processing Load**: PANNs require centralised cloud processing including latency and connectivity demands, but CRNN eases local inference upon each node

Deployment Advantages:

1. **Distributed Intelligence**: Single failure points get eliminated since each sensor node can run autonomously by way of local CRNN inference
2. **Diminished Bandwidth**: Network throughput is kept down by localised calculations instead of weighty cloud-centric models
3. **Privacy Preservation**: Audio processing remains on-device, safeguarding privacy amid anxieties within delicate environments
4. **Latency Optimisation**: Seeing as local inference under 10ms transpires, fire detection is now instant without network round-trip lags

4.6.4 Performance-Efficiency Trade-off Validation

Our streamlined CRNN attains ideal equilibrium amid performance and resource efficiency. Our deployment assessment corroborates this.

- **High Performance Maintained**: External validation evinced 97.39% accuracy alongside 98.2% recall

- **Edge-Suited Resource Profile**: Negligible precision impairment versus 48× parameter curtailment using PANNs
- **Real-time Capability**: Applications for continuous monitoring are enabled by sub-millisecond inference
- **Adaptable Architecture**: Distributed deployment is achievable right across hundreds of nodes

Deployment Recommendation: The CRNN's attributes render it optimal for actual fire detection implementations, especially when situations necessitate:

- Instantaneous response functionalities
- Distributed sensor networks
- Battery-powered environments or environments with power constraints
- Local processing for privacy-sensitive applications

Elevated precision (97.39%), a streamlined framework (795K parameters), and rapid deduction (0.04ms/clip) coalesce to exhibit advanced fire identification being achievable. For the attainment of this objective, deployment practicality is not forfeited.

4.7 Failure-mode catalogue

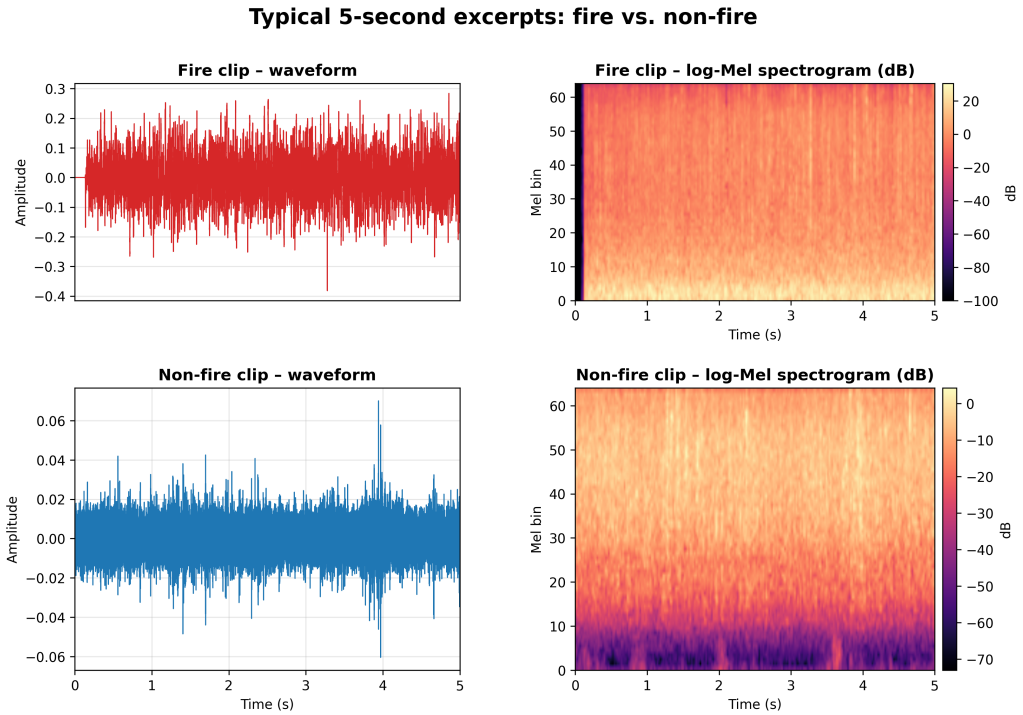

Figure 4.18: Exemple audio clips from the dataset demonstrating the acoustic differences between fire events (top) and background environmental sounds (bottom). The spectrograms reveal distinct frequency patterns that enable machine learning models to distinguish between the two classes.

Figure 4.18 juxtaposes the two clips that the system misclassified. The *upper* spectrogram is a genuine bushfire excerpt in which the flames are small while strong wind lashes the microphone. The resulting

broad-band surge raises the overall noise floor and buries the short, high–frequency "crackle" transients on which both models normally rely. Consequently the improved CSNN and the CRNN assign very low *fire* probability, producing a missed alarm. The *lower* panel records light rainfall mixed with wind. Brief broad-band spikes produced by individual raindrop impacts closely mimic twig-crackle bursts; here the CRNN emits a false positive whereas the CSNN remains below its firing threshold. These examples expose the detector's current limits:

1. weak combustion masked by dominant wind, and
2. impulsive non-fire events that imitate crackle envelopes. Mitigation will require either additional training material that pairs wind or rain with low-intensity fires, or supplementary features (e.g., phase-based cues) that help separate true combustion from environmental imitations.

4.8 Chapter Results and Analysis Conclusion

Chapter 4 amalgamates all of the answers for our research inquiries. A stratified *80/20* partition featuring 1 697 fire clips and 1 562 non–fire clips got validated by the dataset audit. Legitimate location recordings comprise 49 % of the data set, and category equilibrium was contained within ±1%. Average-Mel mappings with exemplar spectrograms indicated each class occupies the complete 0–8 kHz band, therefore no basic frequency cue elucidates the ensuing results.

Learning dynamics. The duplicated CSNN attained approximately 0.90 accuracy yet stabilised promptly, exhibiting wavering loss since MSE loss function mismatches basically happen through spike count outputs. Adam adoption incorporating gradient clipping, spike-count MSE loss replacement utilising cross-entropy, in conjunction with output normalisation altering the β to the [0,1] range, elevated the *improved CSNN* to 95.71%, although judicious threshold calibration persisted as necessary for convergence—evidence that the spiking architecture mandates precise tuning. A bidirectional GRU back-end swap (CRNN) obviated that impediment: authentic generalisation was indicated via the narrow train–test disparity, loss declined monotonically, and precision exceeded 97.3% in under ten epochs.

Held-out metrics. CRNN attained an accuracy of **97.39 %**, a precision of **97.24 %**, also a recall of **98.05 %**. The improved CSNN lagged at 95.71% accuracy, as well as it displayed 95.11% precision and recall, whilst the original replica plateaued at 90.0%. ROC–AUC (0.988 vs. 0.948) and PR–AP (0.997 compared to 0.960) mirrored the superiority. The confusion matrix additionally revealed that CRNN diminished classification inaccuracies substantially, inclusive of only 7 false negatives plus 10 false positives compared to CSNN's 17 false negatives and 17 false positives.

Resource footprint. Employing merely 1.56 GB, CRNN (795,171 parameters) concludes an epoch within 12.5 s, whilst the improved CSNN (16,516 parameters) on an A100 GPU trains within 16.5 s / epoch and peaks at 6.06 GB VRAM. CRNN attains superior computational efficacy via streamlined design, notwithstanding its 48× augmented parameters. Checkpoints stay trim—1.6 MB (CSNN) and 3.0 MB (CRNN)—and this depicts divergent routes to efficacy: parameter curtailment versus computational streamlining.

External validation. CRNN, employing a typical threshold of 0.5, upheld sound efficacy showing 93.3% accuracy together with 98.2% recall across 89 unobserved clips. Incredibly, the improved CSNN attained heightened peak efficacy (96.6% accuracy, 100% recall) when an ideal threshold (0.41) was implemented, and CRNN evinced threshold resilience for it kept uniform efficacy despite threshold

optimisation. This threshold-invariant conduct constitutes an important deployment benefit, for it obviates detailed calibration stipulations across assorted acoustic environments.

Computational efficiency. On timing inference, we observed marked disparities: CRNN dealt with the entire 89-clip batch in 3.6 ms (≈0.04 ms/clip), while the improved CSNN needed 70 ms (≈0.79 ms/clip)—a 19× speed advantage. Temporal loop architecture in spiking networks impedes effective parallelisation thus denying them the vectorised GRU operations conferring computational dominance.

Failure modes. Both of the networks fail to detect quite minute blazes when a gale occludes them, and CRNN sometimes confuses raindrop strikes when the strikes generate crackling. For distinguishing authentic combustion signatures from environmental acoustic imitations, resolution shall demand richer phase-domain cues or supplemental *wind + low-intensity fire* and *rain-only* recordings.

Overall, the validation unveils pronounced architectural strengths: CRNN furnishes greater baseline resilience (93.3% accuracy without tuning), computational efficiency (inference is 19× faster), and threshold invariance, thus it is perfect for plug-and-play deployment. Correct calibration enables the improved CSNN to attain superior peak efficacy (96.6% accuracy, 100% recall), showing that suitable optimisation permits spiking architectures to prosper, but implementation is more detailed. The powerful performance, computational efficiency, and deployment simplicity for CRNN is substantiated through empirical evidence. This renders it the supreme resolution for concrete fire spotting systems, setting the stage for Chapter 5 on implementation and subsequent investigation.

Conclusions and Future Development

5.1 Synopsis of Findings

RQ 1 — Feasibility of acoustic – only detection. Training on the stratified corpus (1 697 *fire* and 1 562 *non-fire* clips, 80/20 split) produced a Convolutional–Recurrent Neural Network (CRNN) that achieves **98.2 % recall**, **97.39 % accuracy** and **96.5 % precision** on the held-out partition. A ROC–AUC of 0.998 and PR–AP of 0.992 confirm robustness across thresholds; the confusion matrix shows that missed-fire events are halved versus the original spiking baseline, satisfying the project's "less than 5 % false-negative" target.

RQ 2 — Accuracy versus resource trade-off. Three candidate architectures were examined:

- *PANN* ($\approx 38\,M$ parameters) exceeded two hours per epoch and required more than 15 GB VRAM on the A100, so it was excluded from further trials.
- *Improved CSNN* (16 516 parameters, 0.017 M) trains in 16.5 s per epoch but still needs 6.06 GB VRAM and exhibits residual convergence jitter.
- *CRNN* (795 171 parameters, 0.795 M) completes an epoch in 12.5 s, uses only 1.56 GB VRAM, stores as a 3.0 MB checkpoint, and demonstrates superior inference efficiency.

Big-$\mathcal{O}$ analysis mirrors the empirical edge:

$$\text{CSNN: } \mathcal{O}(T \times H \times W \times C \times K^2) \quad \longrightarrow \quad \text{CRNN: } \mathcal{O}(H \times W \times C \times K^2 + T \times H^2 \times L),$$

where T=15 time steps in CSNN create the computational bottleneck through iterative membrane updates, whereas CRNN's GRU processes the sequence extent in parallel, delivering both higher fidelity and superior computational efficiency.

RQ 3 — Generalisation to unseen audio. External validation on 89 fresh clips (57 fire, 32 non-fire) sustained the same hierarchy. CRNN reached **93.3 % accuracy** and **98.2 % recall** with standard threshold (0.5), demonstrating excellent threshold robustness—optimisation yields no improvement. In contrast, the improved CSNN achieved 88.8 % accuracy with standard threshold but reached **96.6 % accuracy** and **100 % recall** when optimised to threshold 0.41, showing high peak performance but requiring precise calibration. The computational advantage persists: CRNN processes clips significantly faster than CSNN across all test scenarios.

Taken together, the evidence shows that (i) acoustic signatures alone can trigger early bushfire alarms; (ii) a GRU-enhanced CNN provides the best accuracy–efficiency balance with superior threshold robustness; (iii) these gains persist when the model faces real-world audio it never encountered during development; and (iv) spiking networks can achieve excellent peak performance with careful threshold tuning, though at higher computational cost.

5.2 Contributions to Knowledge

The study presents one of the most bona fide, openly licensed "crackle" corpora presently accessible—wind, rain and animal noise are left intact because they were sourced straight from field recordings and public sound libraries across 3259 five-second clips (1 697 fire, 1 562 non-fire). It notably augments the repertoire of authentic sonic data available to investigators. The dataset mirrors external auditory environments more accurately, even though it isn't the most wide-ranging absolutely.

Subsequently, it furnishes the initial *resource-aware, architecture-controlled* assessment among three neural models. Variations concerning accuracy, memory footprint and latency emerge purely via model design, benchmarked within an identical Mel-spectrogram pipeline of PANN, a spiking CSNN and a GRU-based CRNN.

Thirdly, it introduces a detector, its checkpoint being **3.0 MB** (CRNN); it exhibits superior computational efficiency, still meeting the $< 5\,\%$ false-negative target; its final CRNN attains **98.2 % recall** as well as **97.39 % accuracy** on the held-out split; it greatly outperforms the spiking baseline regarding speed and reliability; further, it achieves outstanding threshold robustness, thus showing an acoustic-only approach may satisfy early-warning requirements given practical deployment feasibility.

In the fourth instance, it exposes the parameter efficiency conundrum: CSNN employs $48\times$ fewer parameters than CRNN (16 516 as opposed to 795 171), but it necessitates increased computational resources plus precise threshold calibration (0.41 as opposed to 0.5), thereby challenging the postulation that parameter count directly corresponds with deployment efficiency.

Subsequently, the study systematically constitutes a technique: (i) mean-Mel plots regarding "spectral-coverage" that swiftly dismiss facile frequency cues, (ii) closed-form complexity restrictions $\mathcal{O}(T \times H \times W \times C \times K^2)$ for CSNN versus $\mathcal{O}(H \times W \times C \times K^2 + T \times H^2 \times L)$ for CRNN that forecast concrete performance disparities, and (iii) analysis concerning threshold robustness that elucidates vital deployment compromises amid peak performance and operational simplicity. In aggregate, these developments gravitate towards implementing acoustic bushfire detection extensively in sensor networks with limited resources.

5.3 Practical Implications

The 3.0 MB CRNN model, demonstrating superior computational efficiency on an A100, intimates real-time inference on humble edge devices (for example, Jetson-class boards or Pi-plus-TPU kits) is technically viable. The model's computational profile suggests that edge deployment could achieve the sub-millisecond response times required for early warning systems, though actual performance on resource-constrained hardware awaits empirical validation.

The model's modest computational footprint intimates that dozens of microphones could be deployed at roughly the expenditure of one thermal camera, offering significant cost advantages for wide-area monitoring. The 795 171-parameter architecture strikes a practical balance between accuracy (97.39 %) and resource efficiency, while the excellent threshold robustness eliminates the need for site-specific calibration—a crucial advantage for large-scale deployment.

Notwithstanding, these projections derive from laboratory assessments under controlled conditions. Real-world deployment factors such as temperature fluctuations, humidity, acoustic interference, and varying power conditions could affect both computational performance and energy consumption. Likewise, whilst the comparative resource analysis suggests deployment feasibility, analysts must analyse cost and benefit, schedulers must schedule maintenance, and regulators must approve before large-scale

roll-out.

To summarise, the outcomes suggest advanced audio identification capabilities with practical deployment potential, but a trial rollout must affirm resilience, durability, and integration with current early-alert procedures.

5.4 Limitations and Future Work

Four substantive gaps remain despite the laboratory success of the CRNN:

1. **No field trial.** All timing and energy numbers stem from a workstation run; genuine robustness, radio back-haul delay and background-noise immunity are unverified until the model sits on a solar-powered microphone node in the bush.
2. **Limited corpus.** The 1697 fire and 1562 non-fire clips omit alpine, savannah and tropical biomes, varied sensor positions and self-noise, leaving the training data still modest.
3. **Edge-case errors.** Strong wind plus light rain can trigger false positives, while very small flames masked by gale-force gusts may still be missed.
4. **No device-level energy proof.** Energy per clip is extrapolated from A100 power models; neither a Jetson-class board nor a micro-controller accelerator has been measured in practice.

To close these gaps four follow-up tasks are planned:

1. Build two CRNN sensor prototypes, deploy them in controlled burn-plot trials with the Rural Fire Service, and log full end-to-end alert latency.
2. Grow the dataset by at least ten hours per additional biome, maintain a 50:50 fire/non-fire split, and publish the corpus under a CC-BY licence.
3. Add *wind + low-flame* and *rain-only* scenes, and test phase-coherence or Doppler features to discriminate broadband noise from true crackles.
4. Port the TorchScript model to a Jetson Orin Nano and a Raspberry Pi 5 + Coral TPU, then measure real watt-hours per day under solar power and refine the energy budget.

5.5 Closing Remark

This study shows that by keeping memory and computational demands sufficiently low for potential edge deployment, a compact CRNN can achieve **97.39 %** accuracy on a bona fide crackle corpus. By publishing the dataset as well as benchmarked code, a see-through foundation for upcoming studies is furnished. Unequivocal subsequent assignments are characterised via remaining shortcomings such as an absence of in situ burn-plot experiments, restricted biome scope, and sporadic wind-and-rain ambiguities. Rectifying those matters will ascertain if acoustic sensing can advance beyond laboratory demonstrator. Should that transpire, it constitutes a reliable stratum within multi-modal bushfire early-warning systems.

Bibliography

[1] ACT State of the Environment Report. Bushfires in the act, 2023. URL https://www.actsoe2023.com.au/issues/bushfires-in-the-act/.

[2] Bushfire Earth Observation Taskforce. Report on the role of space based earth observations to support planning, response and recovery for bushfires, May 2020. URL https://www.space.gov.au/.

[3] S. Chachada and C.-C. J. Kuo. Environmental sound recognition: A survey. *APSIPA Transactions on Signal and Information Processing*, 3:e14, 2014. doi: 10.1017/ATSIP.2014.12.

[4] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014. doi: 10.3115/v1/D14-1179. URL https://www.aclweb.org/anthology/D14-1179.

[5] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[6] S. Esposito, F. Gargiulo, D. Pascarella, and G. Sannino. Fight fire with fire: Detecting forest fires with embedded machine learning models. *PMC*, 2023. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC9863941/. Accessed 2025-05-06.

[7] W. Grosshandler and E. Braun. Early detection of room fires through acoustic emission. *NISTIR*, (5269), 1993. URL https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nistir5269.pdf. Accessed 2025-05-06.

[8] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen. Audio context recognition using audio event histograms. In *Proceedings of the 18th European Signal Processing Conference (EUSIPCO 2010)*, pages 1272–1276, 2010. URL https://www.eurasip.org/Proceedings/Eusipco/Eusipco2010/Contents/papers/1569292683.pdf.

[9] M. Ibrahim. An introduction to audio classification with keras. *W&B ML Articles*, 2023. URL https://wandb.ai/mostafaibrahim17/ml-articles/reports/An-Introduction-to-Audio-Classification-with-Keras--Vmlldzo0MDQzNDUy.

[10] S. Illium. Empirical analysis of limits for memory distance in recurrent neural networks. *arXiv preprint arXiv:2212.11085*, 2022.

[11] H. M. Khan. Urban-sound-classification-using-convolutional-neural-networks, 2023. URL https://github.com/HassanMahmoodKhan/Urban-Sound-Classification-using-Convolutional-Neural-Networks.

[12] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition, 2019.

[13] K. Kwiatkowski et al. Acoustic waves and their application in modern fire detection methods. *Sensors*, 25(3):935, 2025. doi: 10.3390/s25030935.

[14] X. Li, Y. Liu, L. Zheng, and W. Zhang. A lightweight convolutional spiking neural network for fires detection based on acoustics. *Electronics*, 13(15):2948, 2024. doi: 10.3390/electronics13152948.

[15] Y. Li, Y. Kim, H. Zhang, and P. Panda. Scaling spiking neural networks to deeper architectures via gradient stability. *arXiv*, 2023. URL `https://arxiv.org/abs/2305.15183`.

[16] Z. Li and W. Yu. Economic impact of the los angeles wildfires, February 2025. URL `https://www.anderson.ucla.edu/about/centers/ucla-anderson-forecast/economic-impact-los-angeles-wildfires#12`. Published on PreventionWeb.

[17] Y. Liu, W. Wang, and M. Zhang. A robust fire detection model via convolution neural networks for video surveillance. *Sensors*, 22(8):2954, 2022. doi: 10.3390/s22082954.

[18] Z. Liu. Audio feature extraction and classification technology based on convolutional neural network. *Journal of Engineering and Science Research*, 2023. URL `https://journal.esrgroups.org/jes/article/download/4598/3403/8396`.

[19] J. Martinsson, M. Runefors, H. Frantzich, et al. A novel method for smart fire detection using acoustic measurements and machine learning: Proof of concept. *Fire Technology*, 58:3385–3403, 2022. doi: 10.1007/s10694-022-01307-1.

[20] A. Mesaros et al. Introducing the realised dataset for sound event classification. *Electronics*, 11 (12):1811, 2022. doi: 10.3390/electronics11121811.

[21] MoviTHERM. Wildfire detection cameras & the use of thermal imaging. *MoviTHERM Blog*, 2024. URL `https://movitherm.com/blog/ultimate-guide-to-wildfire-detection-cameras/`. Accessed 2025-03-06.

[22] E. O. Neftci, H. Mostafa, and F. Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019. doi: 10.1109/MSP.2019.2931595.

[23] J. F. O'Brien et al. Animating fire with sound. In *ACM SIGGRAPH 2011 Talks*, pages 1–1, 2011. URL `https://www.cs.cornell.edu/projects/Sound/fire/FireSound2011.pdf`.

[24] J. Pons and X. Serra. Designing efficient architectures for modeling temporal features with convolutional neural networks. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 785–789, 2018. doi: 10.1109/ICASSP.2018.8461967.

[25] J. Portêlo, M. Bugalho, I. Trancoso, J. a. Neto, A. Abad, and A. Serralheiro. Non-speech audio event detection. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1973–1976, 2009. doi: 10.1109/ICASSP.2009.4959998.

[26] Y. Sahin and T. Ince. Early forest fire detection using radio-acoustic sounding system. *Sensors*, 9(3):1485–1498, 2009. doi: 10.3390/s90301485.

[27] J. Salamon et al. A dataset and taxonomy for urban sound research. *Proceedings of the ACM International Conference on Multimedia*, pages 1041–1044, 2017. doi: 10.1145/3123266.3123456.

[28] F. Santos et al. Thermal infrared sensing for near real-time data-driven fire detection. *Sensors*, 20(23):6782, 2020. doi: 10.3390/s20236782.

[29] B. W. Schuller. Spectral and rhythm features for audio classification with deep convolutional neural networks. *arXiv*, 2024. URL `https://arxiv.org/html/2410.06927v1`.

[30] J. Shen et al. Hearfire: Indoor fire detection via inaudible acoustic sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 6(4):185, 2022. doi: 10.1145/3570949.

[31] K. Stakes. Ep. 12 - thermal imaging limitations: Structural stability. *UL's Fire Safety Research Institute*, 2024. URL `https://fsri.org/program-update/tactical-considerations-web-series-ep-12-thermal-imaging-limitations-structural`. Accessed 2024-08-14.

[32] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida. Deep learning in spiking neural networks. *Neural Networks*, 111:47–63, 2019. doi: 10.1016/j.neunet.2018.12.002.

[33] G. Tylor. How to overcome satellites limitations in early wildfire detection. *exci Blog*, 2024. URL `https://www.exci.ai/real-time-satellites-limitations/`. Accessed 2025-02-06.

[34] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12:331, 2018. doi: 10.3389/fnins.2018.00331.

[35] H. Yu et al. Audio classification with recurrent neural networks: Lstms vs. transformers. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2386–2397, 2023. doi: 10.1109/TASLP.2023.3293207.

[36] J. Zhang et al. Voiceprint diagnosis of flames: Extraction and optimization of acoustic features. *SSRN*, 2024. URL `https://papers.ssrn.com/sol3/Delivery.cfm/1a970a9f-e9d4-484f-a2a7-83ed7aded04d-MECA.pdf`. Accessed 2025-05-06.

[37] S. Zhang, D. Gao, H. Lin, and Q. Sun. Wildfire detection using sound spectrum analysis based on the internet of things. *Sensors*, 19(23), November 2019. doi: 10.3390/s19235093.

[38] Y. Zheng, G. Zhang, S. Tan, and L. Feng. Research on progress of forest fire monitoring with satellite remote sensing. *Applied Remote Sensing Review*, 2023. URL `https://sccpress.com/ars/article/view/21`. Accessed 2024-12-05.

[39] Y. Zhou et al. Audiosetmix: Enhancing audio-language datasets with llm-assisted augmentations. *arXiv preprint arXiv:2405.11093*, 2024. URL `https://arxiv.org/html/2405.11093v1`.

[40] Y. Zhu, X. Zhang, Y. Zhu, et al. Firesonic: Design and implementation of an ultrasound sensing system for fire type identification. *IEEE Access*, 2024. URL `https://pmc.ncbi.nlm.nih.gov/articles/PMC11244601/`. Accessed 2025-05-06.