Independent University **Bangladesh (IUB)**

IUB Academic Repository

Computer Science and Engineering | Undergraduate Thesis

2026-04

Object Detection and Localization Using Lightweight Convolutional Neural Networks for Low-Resolution Thermal Sensor Data

Barua, Protik

IUB

https://ar.iub.edu.bd/handle/11348/1167

Downloaded from IUB Academic Repository



Object Detection and Localization Using Lightweight Convolutional Neural Networks for Low-Resolution Thermal Sensor Data

Prepared By

Protik Barua

ID: 2110082

Spring 2026

Department Of Computer Science and Engineering

Independent University Bangladesh

Supervisor:

Dr. Raqibul Hasan

Assistant Professor,

Department of Computer Science and Engineering

School of Engineering Technology and Science

Independent University Bangladesh

Attestation

We understand what plagiarism is and how strict the university's anti-plagiarism policy is. This is our original work, we attest to that. Additionally, any third-party software or text included in this project is properly cited and compliant with well recognized academic standards.

Protik Barua

_____	_____
Name	Signature with Date

Acknowledgement

We commence by offering our heartfelt gratitude to the Almighty God for endowing us with the fortitude, perseverance and capability to diligently embark on and successfully complete our senior project. Our profound gratitude extends to Dr. Raqibul Hassan, our esteemed Supervisor, Assistant Professor of Independent University Bangladesh. His invaluable guidance, boundless patience and devoted investment of time have been instrumental in steering us through the intricacies of our project journey and shaping the formulation of this comprehensive report. Under their expert tutelage, our project progressed seamlessly and their insightful counsel proved indispensable at every juncture.

Moreover, We express our sincere gratitude to the committee members for their valuable contribution during the defense session, which rendered the experience not only fruitful but also enriching. Their discerning feedback and invaluable recommendations served to augment the depth and quality of our project significantly.

In addition to our academic mentors, we are deeply indebted to our family and friends for their support and encouragement throughout the journey in the BSc program, particularly during the more challenging phases of our senior project.

We extend our deepest gratitude to Independent University Bangladesh for their exceptional senior project program. It has been instrumental in building our skills and equipping us for the demands of the research industry. Their support has developed our academic and professional journeys, and we are truly grateful for that.

Letter Of Transmittal

Dr. Raqibul Hasan
Senior Project Supervisor
Assistant Professor
Department of Computer Science and Engineering
Independent University Bangladesh

Subject: Senior Project Report on "Object Detection and Localization Using Lightweight Convolutional Neural Networks For Low Resolution Thermal Sensors Data"

Dear Sir,

We are truly honored to share our senior project report on "Object Detection and Localization Using Lightweight Convolutional Neural Networks For Low Resolution Thermal Sensors Data" with you. Working on this project under your supervision has been an incredible experience, teaching us so much about Robotics while bringing our vision of a smarter Agriculture System to Life. Our project with its design and complete development reflects the skills we have learned and the late night efforts we poured into making it work. Your constant support, insightful feedback and encouragement kept us going and we are thankful for that. We hope this report captures the heart of our project and meets your expectations. It's been a pleasure to learn from you throughout the semesters.

Yours sincerely,

Protik Barua (2110082)

Department of Computer Science and Engineering
Independent University Bangladesh

Evaluation Committee

Supervisor

Name: _____ Signature: _____

Internal Examiner 1

Name: _____ Signature: _____

Internal Examiner 2

Name: _____ Signature: _____

External Examiner

Name: _____ Signature: _____

Head of Department

Name: _____ Signature:_____

List of Abbreviations

Abbreviation	Full Form
CNN	Convolutional Neural Network
J-Filter	Custom preprocessing pipeline (background subtraction)
AMG8833	Panasonic Grid-EYE 8×8 Thermal Sensor
IR	Infrared
RGB	Red, Green, Blue (color image format)
MTL	Multi-Task Learning
CE	Cross-Entropy (Loss Function)
MSE	Mean Squared Error
MAE	Mean Absolute Error
RMSE	Root Mean Squared Error
IoT	Internet of Things
I2C	Inter-Integrated Circuit (communication protocol)
FC	Fully Connected (Layer)
ReLU	Rectified Linear Unit
GAP	Global Average Pooling
LR	Learning Rate
AdamW	Adaptive Moment Estimation with Weight Decay
CV	Cross-Validation
SOTA	State-of-the-Art
ESP32	Microcontroller (used for edge deployment)
ML	Machine Learning
DL	Deep Learning

Abstract

Embedded monitoring systems that rely on RGB cameras raise well documented privacy concerns in sensitive spaces. Low-resolution thermal sensors offer an alternative: the Panasonic AMG8833 Grid-EYE outputs only an 8×8 grid of 64 temperature values too coarse to identify individuals, yet structured enough to distinguish an empty room, a person, and a fire event.

Prior work on AMG8833-class sensors has relied on handcrafted features (thresholding, interpolation, blob detection) rather than deep learning. Such pipelines are task-specific and do not generalize to joint multi-class recognition and localization at native 8×8 resolution.

This thesis proposes JFilterLocalizationCNN, a lightweight dual-head convolutional neural network for simultaneous classification and heat-source localization on native 8×8 thermal data. Three classes are used - No Object, Object, and Object with Fire - with normalized (x, y) coordinates for the dominant heat source. A J-filter removes ambient offset via per-frame background subtraction; normalization maps readings to $[0, 1]$ for stable training without distorting spatial signatures needed for localization.

The architecture uses a VGG-style shared backbone: three convolutional blocks reduce spatial size from 8×8 to 1×1 (effective path $8 \to 4 \to 2 \to 1$), then global average pooling feeds two heads (classification and sigmoid-bounded localization). About 115,365 trainable parameters fit the memory budget of microcontroller-class devices (e.g., Seeed XIAO ESP32).

Training uses AdamW with combined cross-entropy and MSE losses on 3,087 labeled frames. On a stratified test set of 618 samples, the model reaches 99.68% classification accuracy (F1 above 0.99 per class) and 0.37-pixel MAE localization (RMSE 0.67 pixels). The two tasks share one forward pass and one set of convolutional weights, which keeps the design small enough to reason about on paper and, in principle, to port to an embedded runtime later.

The experiments here were conducted in a single indoor setting with a controlled heat source for the fire-related class. That scope limits how far one should generalize the numbers, yet they still suggest that joint learning on native 8×8 frames is workable when preprocessing preserves relative heat structure. On-device latency, power use, and post-training quantization are left for future work.

Chapters that follow spell out the dataset, preprocessing, layer stack, training schedule, and metrics in full. The aim is a thesis someone can audit line by line rather than a high-level claim alone.

Contents

1 Introduction **11**

1.1 Background of the Project 11
1.2 Problem Statement 11
1.3 Research Objectives 12
1.4 Contributions 12
1.5 Thesis Organization 12

2 Literature Review **14**

2.1 Low-Resolution Thermal Sensing for Indoor Monitoring 14
2.2 Lightweight CNN Architectures 15
2.3 Object Detection and Localization 16
2.4 Multi-Task Learning 16
2.5 TinyML and Embedded Inference 17
2.6 Research Gaps and Motivation 17

3 Dataset and Preprocessing **19**

3.1 Hardware Setup 19
3.2 Data Collection 19
3.3 Dataset Overview 20
3.4 Preprocessing Pipeline 21
3.4.1 J-Filter (Background Subtraction) 21
3.4.2 Per-Frame Normalization 22
3.4.3 Coordinate Normalization 22
3.4.4 Reshape 22
3.5 Data Split Strategy 22

4 Proposed Methodology **24**

4.1 System Overview 24
4.2 JFilterLocalizationCNN Architecture 25
4.2.1 Shared Convolutional Backbone 26
4.2.2 Classification Head 27
4.2.3 Localization Head 27

4.2.4 Layer-by-Layer Summary . . . 28
4.3 Loss Functions . . . 28
4.3.1 Classification Loss . . . 28
4.3.2 Localization Loss . . . 29
4.3.3 Total Loss . . . 29
4.4 Architecture Design Justification . . . 29

5 Experimental Design **31**
5.1 Implementation Details . . . 31
5.2 Training Configuration . . . 31
5.3 Evaluation Metrics . . . 32
5.3.1 Classification Metrics . . . 32
5.3.2 Localization Metrics . . . 32
5.4 Training Rationale and Design Decisions . . . 33

6 Results and Analysis **34**
6.1 Training Convergence . . . 34
6.2 Classification Performance . . . 36
6.2.1 Confusion Matrix . . . 36
6.2.2 Per-Class Metrics . . . 37
6.3 Localization Performance . . . 38
6.3.1 Predicted vs. True Coordinates . . . 38
6.3.2 Per-Class Localization Error . . . 39
6.4 Combined Results Summary . . . 39
6.5 Discussion . . . 40
6.6 Cross-Dataset Generalization . . . 41
6.6.1 External Datasets . . . 41
6.6.2 Preprocessing and Distribution Analysis . . . 41
6.6.3 Zero-Shot Evaluation . . . 43
6.6.4 Fine-Tuning Transfer . . . 44
6.6.5 5-Fold Cross-Validation . . . 46
6.6.6 Cross-Dataset Summary . . . 47
6.6.7 Analysis of Findings . . . 48
6.7 Ablation Study . . . 48
6.7.1 Experimental Configurations . . . 49
6.7.2 Ablation Results . . . 49
6.7.3 Analysis of Ablation Findings . . . 52
6.7.4 Design Validation Summary . . . 53

7 Conclusion and Future Works **55**

7.1 Conclusion 55
7.2 Limitations 55
7.3 Ethical Considerations 56
7.4 Future Work 56

Bibliography **56**

List of Figures

1 Sample raw and J-filtered thermal heatmaps for each class. Top row: raw sensor output showing absolute temperatures. Bottom row: after J-filter background subtraction, highlighting relative thermal differences. Columns represent No Object, Object, and Object with Fire classes respectively. 20

2 J-Filter preprocessing pipeline: raw thermal frame → background subtraction → per-frame normalization → reshape to CNN input tensor. 21

3 System methodology: end-to-end pipeline from AMG8833 thermal sensor through J-filter preprocessing to the dual-head JFilterLocalizationCNN producing classification and localization outputs. 24

4 JFilterLocalizationCNN architecture: shared convolutional backbone (3 blocks) with global average pooling, feeding into separate classification and localization heads. 26

5 Training and validation loss curves over 50 epochs, showing total loss, classification cross-entropy loss, and localization MSE loss for both training and validation sets. 34

6 Classification accuracy over training epochs for both training and validation sets. Training accuracy reaches 100% by epoch 15; validation accuracy stabilises at 99.68%. 35

7 Localization MSE over training epochs. Training MSE steadily decreases from 0.0387 to 0.0046 across 50 epochs, while validation MSE stabilizes around 0.0104. 35

8 Loss decomposition: stacked area plot showing the contribution of cross-entropy (CE) and MSE losses to the total training loss across epochs. After approximately epoch 15, the CE loss is negligible and localization MSE dominates. . . 36

9 Confusion matrix on the test set (618 samples). The model achieves 99.68% overall accuracy with minimal misclassifications across all class pairs. 37

10 Classification report heatmap showing per-class precision, recall, and F1-score on the test set. All classes exceed 0.99 F1-score. 37

11 Scatter plot of predicted vs. true localization coordinates on the test set. Each point represents a test sample, color-coded by class. Short error lines indicate predictions that closely match ground truth. 38

12 Box plot of per-class Euclidean localization error (in pixels). Median and mean values are annotated for each class. 39

13 Pixel value distributions after J-filter and normalization. The AMG8833 training data concentrates near zero; LINAIGE and MLX90640 are shifted rightward, explaining the domain shift that degrades zero-shot performance. 42

14 Sample 8×8 thermal frames from each dataset (J-filtered, normalized). Each row corresponds to one dataset; columns alternate between No Object and Object classes. The visual differences between rows illustrate the domain shift that the model must overcome. 43

15 Zero-shot confusion matrices. Left: LINAIGE (5,000 samples). Right: MLX90640 (384 samples). The model biases heavily toward predicting Object on both external distributions. 44

16 Fine-tuning loss and accuracy curves for LINAIGE (left) and MLX90640 (right). The dashed line marks the transition from frozen backbone (Stage 1) to full model (Stage 2). On LINAIGE, the frozen backbone already achieves 91.8% test accuracy. 45

17 Fine-tuned confusion matrices. Left: LINAIGE. Right: MLX90640. The systematic Object bias from zero-shot evaluation has been largely corrected. . . . 45

18 Transfer learning improvement over zero-shot baseline. Purple arrows indicate the accuracy gain from fine-tuning. The dashed line marks the original AMG8833 in-distribution accuracy (99.68%). 46

19 5-fold cross-validation box plots for LINAIGE and MLX90640 across accuracy, precision, recall, and F1. Red diamonds mark fold means; the tight LINAIGE boxes contrast with the wider MLX90640 spread. 47

20 Cross-dataset performance comparison across all evaluation methods. The AMG8833 baseline (dark) is the original 3-class result; external datasets show binary (No Object / Object) results under three evaluation protocols. 47

21 Classification accuracy and macro F1 score across all ablation variants. Most configurations retain >99% accuracy; the *Localisation Only* variant confirms that the untrained classification head produces near-random predictions. 50

22 Localisation MSE (left) and MAE in pixels (right) for each ablation variant. Removing the localisation loss (*Classification Only*) degrades coordinate prediction to near random, while weighting the MSE term higher improves precision. 51

23 Validation loss over 50 epochs grouped by ablation category. Preprocessing and training variants converge similarly to the baseline; architectural variants show minor divergence; multi-task variants exhibit qualitatively different loss trajectories due to differing loss compositions. 51

24 Radar plots showing normalised performance across four metrics, grouped by ablation category. The baseline polygon (black) serves as the reference; deviations indicate degradation. 52

List of Tables

1 Comparison of low-resolution thermal sensing studies 15

2 Hardware specifications 19
3 Class distribution in the dataset 20
4 Data split configuration 23

5 Layer-by-layer summary of JFilterLocalizationCNN 28
6 Design choices and their justifications 29

7 Training hyperparameters 31

8 Training convergence at selected epochs 36
9 Localization performance on the test set 38
10 Combined test set performance summary 39
11 Comparison of dataset properties 41
12 Zero-shot binary classification on external datasets 43
13 Fine-tuned binary classification on external datasets 44
14 Accuracy improvement from zero-shot to fine-tuned 44
15 5-fold stratified cross-validation results (mean $\pm$ std) 46
16 Complete cross-dataset evaluation summary 47
17 Ablation study results. ΔAcc denotes the change from the baseline. 49
18 Variants that outperform the baseline on at least one metric. Bold values mark improvements; the baseline row is shaded for reference. 54

1 Introduction

1.1 Background of the Project

The demand for indoor monitoring has grown steadily over the past decade. Applications range from occupancy-aware building management and elderly fall detection to early fire warning systems in residential and industrial settings. Most commercial solutions rely on RGB cameras, which deliver rich visual information but simultaneously capture identifiable imagery of occupants. In privacy sensitive environments hospitals, care homes, private residences this trade-off is difficult to justify. Regulations such as the GDPR have made the tension between surveillance capability and personal privacy even more acute.

Low-resolution infrared thermal sensors offer a different path. The Panasonic AMG8833 Grid-EYE, a thermopile array that outputs an 8×8 grid of 64 temperature readings over I^2C, is physically incapable of producing identifiable images. A person registers as a warm blob spanning a few pixels; a fire source appears as a localized hot spot. The spatial resolution is, by any conventional standard, extremely low. Yet that very limitation is also its strength: privacy is preserved by hardware, not by software anonymization that could be reversed.

The question, then, is whether such coarse thermal data carries enough spatial structure to support meaningful detection and localization tasks. Prior work on AMG8833-class sensors has largely relied on handcrafted features thresholding, interpolation, statistical distribution fitting - to extract information from these tiny grids. Deep learning approaches, which have transformed high-resolution computer vision, remain largely unexplored at this resolution. A gap exists between the demonstrated potential of lightweight CNNs and their application to ultra-low-resolution thermal sensing.

1.2 Problem Statement

Existing approaches to indoor fire and person detection fall into two broad categories, each with significant drawbacks. Camera based systems achieve high accuracy but compromise occupant privacy. High-resolution thermal cameras (e.g., 160×120 or 320×240 pixels) avoid the privacy issue but remain expensive, power-hungry, and too large for many embedded scenarios.

Low-resolution thermal arrays like the AMG8833 are cheap, compact, and inherently private. However, the 8×8 pixel resolution presents a challenge that most existing work addresses through handcrafted image processing pipelines blob detection, adaptive thresholding, or statistical feature extraction. These methods tend to be task specific and brittle. No prior study, to the best of our knowledge, applies a learned CNN directly to native 8×8 thermal data for joint classification and localization. Nor does any existing lightweight architecture target this particular input geometry, where spatial dimensions are exhausted after just three pooling operations.

This thesis addresses that gap by designing a compact dual-head CNN capable of operating on

the raw 8×8 sensor grid, jointly classifying the thermal scene and predicting the coordinates of the dominant heat source, all within a parameter budget small enough for potential microcontroller deployment.

1.3 Research Objectives

The primary objectives of this research are:

1. To design a lightweight CNN that operates directly on native 8×8 thermal sensor data without upscaling or interpolation.
2. To develop a multi-task framework that classifies thermal events (No Object, Object, Object with Fire) and localizes the dominant heat source in the grid.
3. To evaluate deployment feasibility on resource constrained platforms such as the Seeed XIAO ESP32 microcontroller.

1.4 Contributions

The specific contributions of this work are as follows:

- A custom J-filter preprocessing pipeline that removes ambient temperature offset through per-frame background subtraction, making the model invariant to room temperature variations across recording sessions.
- A lightweight dual-head CNN architecture (JFilterLocalizationCNN, ∼115K parameters) that jointly performs 3-class classification and heat-source coordinate regression on native 8×8 thermal frames.
- A curated dataset of 3,087 labeled thermal frames across three classes (No Object, Object, Object with Fire) collected using the AMG8833 Grid-EYE sensor, with ground-truth localization coordinates encoded in the filename metadata.
- An empirical evaluation demonstrating 99.68% classification accuracy and 0.37-pixel mean absolute localization error on the 8×8 grid, establishing baseline performance for this task.

1.5 Thesis Organization

The remainder of this thesis is organized as follows:

- **Chapter 2** reviews related work on low-resolution thermal sensing, lightweight CNN architectures, object detection and localization, multi-task learning, and TinyML.
- **Chapter 3** describes the hardware setup, dataset collection, class definitions, and the J-filter preprocessing pipeline.
- **Chapter 4** presents the proposed JFilterLocalizationCNN architecture, including the shared convolutional backbone, dual-head design, and loss formulation.
- **Chapter 5** details the experimental design, including training configuration, hyperparameters, and evaluation metrics.
- **Chapter 6** presents the experimental results and analysis, covering training convergence, classification performance, and localization accuracy.
- **Chapter 7** concludes the thesis with a summary, known limitations, and directions for future research.

2 Literature Review

This chapter examines five areas of research relevant to the proposed system: low-resolution thermal sensing for indoor monitoring, lightweight CNN architecture design, object detection and localization, multi-task learning, and embedded machine learning inference. Each section identifies the state of current knowledge and the specific gaps that motivate this work.

2.1 Low-Resolution Thermal Sensing for Indoor Monitoring

A number of studies have explored low-resolution infrared arrays for privacy-preserving indoor applications. Newaz and Hanada [13] used thermal data from an infrared array sensor for human activity recognition, relying on interpolation and mathematical feature extraction rather than any form of deep learning. Their approach detected activities from thermal silhouettes with lower privacy risk than RGB cameras, but the handcrafted features limited the system's ability to generalize beyond the specific activities it was designed to recognize.

Lin and Zhao [7] worked directly with AMG8833 data for human occupancy monitoring and positioning. They employed interpolation, adaptive thresholding, and blob detection combined with a speed responsive sliding window. Their system demonstrated that 8×8 thermal arrays can support indoor positioning tasks and proved more stable under movement than static thresholding methods. However, the entire pipeline was built on traditional image processing no learned representations were involved, and the system handled only occupancy and positioning, not event classification or fire detection.

Márquez et al. [11] used a slightly higher resolution sensor (32×24 pixels) for classifying activities and detecting falls among older adults. Their skeletonization approach required more spatial detail than an 8×8 array can provide, and no lightweight CNN was considered for the task. Newaz and Hanada [14] later proposed a fall detection method based on statistical distributions of thermal signatures obtained from a stand-alone IR array device. Earth Mover's Distance served as the primary discriminator between fall and non-fall thermal patterns. While effective for binary fall detection, this statistical approach does not generalize to multi-class scenarios or simultaneous localization.

Sousa et al. [19] reviewed thermal infrared sensing for fire detection and monitoring, but their focus was on full thermal cameras used in outdoor and UAV contexts - a different hardware class entirely from ultra-low-resolution arrays. Naser et al. [12] applied a deep encoder-decoder network to thermal array data for human segmentation and occupancy estimation, achieving high accuracy with adaptive sensor placement. Their work is perhaps the closest to ours in spirit, yet it addressed segmentation and occupancy only, did not consider fire recognition, and used a more complex architecture than what an 8×8 input warrants.

Across this body of work, a consistent pattern emerges: studies either use handcrafted features on low-resolution arrays or apply deep learning to higher-resolution thermal cameras. No prior study combines a learned CNN with native 8×8 thermal input for joint classification and local-

ization. Table 1 summarizes the key differences between existing thermal sensing approaches and the system proposed in this thesis.

Table 1: Comparison of low-resolution thermal sensing studies

Study	Resol.	Method	Class.	Loc.
Newaz & Hanada (2024)	8×8	Statistical	Activity	No
Lin & Zhao (2025)	8×8	Blob det.	Occupancy	Pos.
Márquez et al. (2022)	32×24	Skeleton.	Falls	No
Newaz & Hanada (2025)	IR array	EMD stats.	Falls	No
Sousa et al. (2020)	High-res	Thermal img.	Fire	No
Naser et al. (2021)	IR array	Enc.-dec.	Segm.	No
Ours	**8×8**	**CNN**	**3-class**	**Coord.**

The table makes the gap visible at a glance: existing 8×8 approaches use handcrafted pipelines, and none jointly addresses both classification and coordinate-level localization through a learned model. The only study that applies deep learning to thermal array data (Naser et al.) operates at a different task granularity (segmentation) and does not consider fire detection.

2.2 Lightweight CNN Architectures

The design of compact convolutional networks has received considerable attention, driven largely by mobile and edge deployment requirements. Simonyan and Zisserman [18] established the principle that stacking small 3×3 convolution kernels can achieve the receptive field of larger kernels while introducing more non-linearity - a design philosophy that directly informs our architecture. He et al. [4] later demonstrated that residual connections enable training of very deep networks (up to 152 layers), but such depth is unnecessary and counterproductive for 8×8 inputs where spatial dimensions are exhausted within three pooling operations.

Several architectures have specifically targeted mobile deployment. MobileNetV2 [17] introduced inverted residual blocks with linear bottlenecks, achieving a favorable accuracy-to-computation trade-off on standard benchmarks. MobileNetV3 [5] refined this further using neural architecture search. ShuffleNet [24] and its successor ShuffleNet V2 [10] employed group convolution and channel shuffle operations; notably, Ma et al. argued that FLOPs alone are a poor predictor of actual inference speed, and that hardware-aware design matters more. GhostNet [3] generated feature maps through cheap linear transformations, while EfficientNet [20] proposed compound scaling to balance network width, depth, and resolution. Pelee [21] demonstrated competitive real-time detection on mobile devices using a conventional convolution backbone instead of depthwise separable operations.

A common thread across these architectures is the reliance on depthwise separable convolutions, channel manipulation, or neural architecture search to reduce computational cost while maintaining accuracy on standard benchmarks. The underlying assumption in every case is that the input is a standard-resolution RGB image, typically 224×224 pixels or larger, with three color channels. None of them address the unique constraints of an 8×8 single-channel thermal input, where the entire spatial extent of the image is consumed in just three downsampling steps.

The mismatch runs deeper than resolution alone. Techniques such as depthwise separable convolutions achieve savings by factoring spatial and channel-wise processing, but on an 8×8 grid the spatial dimensions are so small that the factorization yields negligible benefit while adding architectural complexity. Similarly, compound scaling strategies assume a continuous range of model sizes where width, depth, and resolution can be traded off against each other; at 8×8 input, the resolution dimension is already at its minimum, and depth is bounded by three pooling stages. Adapting any of these architectures to ultra-low-resolution thermal input would require fundamental redesign rather than simple scaling, motivating our custom architecture.

2.3 Object Detection and Localization

Modern object detection has been shaped by two paradigms. Two-stage detectors like Faster R-CNN [16] use region proposal networks to identify candidate regions, then classify and refine bounding boxes. Single-stage detectors such as YOLO [15] and SSD [9] predict class labels and bounding box coordinates directly from feature maps in a single forward pass. Feature Pyramid Networks [6] introduced top down pathways with lateral connections for multi-scale detection, which has become standard in modern detectors.

Wu et al. [22] investigated the tension between classification and localization in object detection, finding that the two tasks benefit from different head architectures fully connected layers favor classification while convolutional layers favor localization. This observation provides theoretical support for our dual-head design with separate specialized branches.

However, these detection frameworks are designed for natural images with multiple objects, bounding box annotations, and rich spatial hierarchies. Our task differs in three fundamental respects. First, the input is an 8×8 single channel thermal grid, not a high-resolution RGB image. Second, we localize a single dominant heat source using $(x,\, y)$ coordinate regression rather than predicting bounding boxes the concept of a bounding box is poorly defined when the object of interest spans only two or three pixels in a 64-pixel image. Third, the spatial hierarchy available in three convolutional blocks is far shallower than what standard detectors expect; multi-scale feature pyramids cannot be constructed from three levels of resolution that span just $8 \to 4 \to 2$.

Despite these fundamental differences, the theoretical insights from this literature remain valuable. Wu et al.'s finding that classification and localization benefit from different head architectures directly motivates our dual-head design. YOLO's demonstration that classification and coordinate regression can be solved jointly in a single forward pass validates the multi-task approach. The key contribution of our work lies in translating these established principles to an extreme operating regime - ultra-low resolution, single-channel, and microcontroller-scale parameters - where no prior detection architecture has been applied.

2.4 Multi-Task Learning

Multi-task learning (MTL) leverages shared representations across related tasks to improve generalization and efficiency. Zhang and Yang [25] surveyed the field extensively, documenting how shared feature learning reduces overfitting when tasks are related and training data is limited - both conditions that apply to our problem. Liu et al. [8] proposed task-specific soft attention modules for end-to-end MTL, demonstrating that combining shared features with task-specialized branches outperforms hard parameter sharing alone. Dai et al. [2] showed that cascaded multi-task architectures can effectively handle related vision tasks (proposal generation,

mask prediction, categorization) through a shared CNN backbone.

These studies validate the dual-head design principle: a shared backbone extracts common features, while separate heads specialize for each task. The theoretical justification is straightforward - when tasks share underlying structure, joint training acts as an inductive bias that improves generalization, particularly when per-task data is limited.

In our setting, classification and localization both depend on spatial heat patterns within the 8×8 grid, making them naturally related tasks that should benefit from shared representations. Determining whether a frame contains a person requires detecting a warm region in the thermal grid; localizing that person requires identifying where the warm region is centered. Both tasks rely on the same spatial feature maps produced by the convolutional backbone. Our architecture follows this principle with a simple shared backbone and two lightweight heads, avoiding the attention mechanisms or cascade structures that would be disproportionately complex for the small input size.

The loss formulation is correspondingly simple: an unweighted sum of cross-entropy and mean squared error. More sophisticated approaches such as the uncertainty based weighting of Kendall et al. or the gradient normalization of Chen et al. have been proposed for balancing multiple task losses, but our empirical results suggest that the two loss terms in this problem happen to operate on comparable scales, making the simple sum a reasonable first choice.

2.5 TinyML and Embedded Inference

The TinyML movement aims to bring machine learning inference to microcontrollers with severely limited memory and compute resources. Banbury et al. [1] presented TensorFlow Lite Micro as a portable runtime for deploying trained models on devices with as little as a few kilobytes of RAM. The framework handles model loading, memory allocation, and inference execution, but it requires the model itself to be small and efficient enough to fit within the target device's constraints.

This places the burden squarely on architecture design. A model targeting an ESP32-class microcontroller cannot simply be a scaled-down version of a desktop model; it must be designed from the ground up with a constrained parameter budget. The XIAO ESP32, for instance, has only 520 KB of SRAM and limited clock speed compared to GPU-equipped workstations. A model that requires even a few megabytes of weights during inference would exceed the device's memory capacity, regardless of how efficient the runtime framework might be.

The constraint shapes the entire design philosophy. Standard practices like using large batch normalization statistics, deep residual blocks with many channels, or attention layers become impractical when every additional parameter consumes precious on-chip memory. Our JFilterLocalizationCNN, with $\sim$115K parameters (approximately 450 KB in 32-bit floating point, or $\sim$115 KB if quantized to INT8), falls well within the capabilities of TinyML deployment frameworks. The three-block convolutional architecture was designed specifically to maximize feature extraction within this budget while matching the sensor's native 8×8 resolution. Actual on-device inference latency and memory usage remain to be validated in future work, but the model's compact size makes such deployment a realistic near-term goal.

2.6 Research Gaps and Motivation

The literature reviewed above reveals four gaps that this thesis addresses:

1. **No deep learning on native 8×8 thermal data.** All prior work on AMG8833-class sensors uses handcrafted features (thresholding, interpolation, statistical fitting). No study applies a CNN directly to the raw 8×8 grid.

2. **No joint classification and localization.** Existing thermal array studies address either detection/classification or positioning, but not both simultaneously within a single model.

3. **No lightweight dual-head architecture for this input geometry.** Standard lightweight CNNs (MobileNet, ShuffleNet, EfficientNet) assume inputs of 224×224 or larger. The 8×8 input requires a custom architecture where spatial dimensions are exhausted in exactly three pooling steps.

4. **No embedded-targeted thermal detection model.** While TinyML frameworks exist, no task-specific architecture has been designed to jointly detect and localize thermal events within the parameter budget of a microcontroller.

Taken individually, each of these gaps has been partially addressed in adjacent domains deep learning is well established for high-resolution images, multi-task learning has been validated extensively in computer vision, and TinyML frameworks exist for embedded deployment. The contribution of this thesis lies in bringing these elements together for the first time in the specific context of ultra-low-resolution thermal sensing, where the input geometry, the task requirements, and the deployment constraints are qualitatively different from those assumed by prior work.

This thesis addresses all four gaps by proposing JFilterLocalizationCNN, a purpose-built dual-head CNN with ∼115K parameters that operates on J-filtered 8×8 thermal frames for simultaneous 3-class classification and heat-source localization. The following chapters describe the dataset and preprocessing pipeline (Chapter 3), the model architecture and loss formulation (Chapter 4), the experimental setup (Chapter 5), and the results obtained under controlled conditions (Chapter 6).

3 Dataset and Preprocessing

3.1 Hardware Setup

The sensing hardware centers on the Panasonic AMG8833 Grid-EYE, a compact thermopile array sensor that measures temperature across an 8×8 grid of 64 pixels. Each pixel reports an absolute temperature in the range of 0–80°C for objects, transmitted over a standard I^2C interface at frame rates up to 10 Hz. The sensor's field of view and small physical footprint make it suitable for ceiling or wall-mounted deployment in indoor environments.

The target deployment platform is the Seeed Studio XIAO ESP32, a microcontroller with limited RAM and processing power that represents the class of edge devices where privacy-preserving thermal monitoring could be most valuable. For the purposes of this thesis, data collection and model training were performed on a desktop workstation; embedded deployment remains a goal for future work. The hardware specifications are summarized in Table 2.

Table 2: Hardware specifications

Component	**Specification**
Sensor	Panasonic AMG8833 Grid-EYE
Resolution	8×8 pixels (64 temperature values)
Temperature Range	0–80°C (object)
Frame Rate	Up to 10 Hz
Interface	I^2C
Target MCU	Seeed Studio XIAO ESP32

3.2 Data Collection

Thermal frames were recorded in a controlled indoor environment under three distinct scenarios. In the first scenario, the room was empty with no person or significant heat source present (No Object class). In the second, a single person occupied the sensor's field of view at varying positions (Object class). In the third, a controlled heat source simulating fire was placed near a person (Object with Fire class). Across all three scenarios, the raw sensor output 64 floating-point temperature values per frame was captured and stored as Python dictionaries in `.joblib` files.

Ground-truth localization coordinates were derived from the corresponding PNG image filenames, which encode the row and column indices of the dominant heat source in the format `frame_XXXX_(row_col).png`. For the No Object class, these coordinates correspond to the pixel with the highest background temperature. Row and column indices (integers in $[0, 7]$) are normalized to $[0, 1]$ by dividing by 7, aligning with the sigmoid output range of the localization

3.3 Dataset Overview

The complete dataset comprises 3,087 thermal frames distributed across three classes, as summarized in Table 3. The class distribution is reasonably balanced, with no single class dominating the dataset by a large margin. The No Object class has the most samples (37.4%), followed by Object with Fire (34.8%) and Object (27.8%).

Table 3: Class distribution in the dataset

Class ID	Class Name	Samples	Proportion
0	No Object	1,154	37.4%
1	Object	858	27.8%
2	Object with Fire	1,075	34.8%
	Total	**3,087**	100%

Figure 1 shows representative raw and J-filtered thermal frames for each class. The raw frames (top row) display absolute temperatures dominated by ambient room heat, making class distinctions difficult to see by eye. After J-filter processing (bottom row), the relative thermal patterns become far more apparent: the No Object frames show nearly uniform low values, the Object frames reveal a warm region corresponding to a person, and the Object with Fire frames show a pronounced hot spot alongside the person's thermal signature.

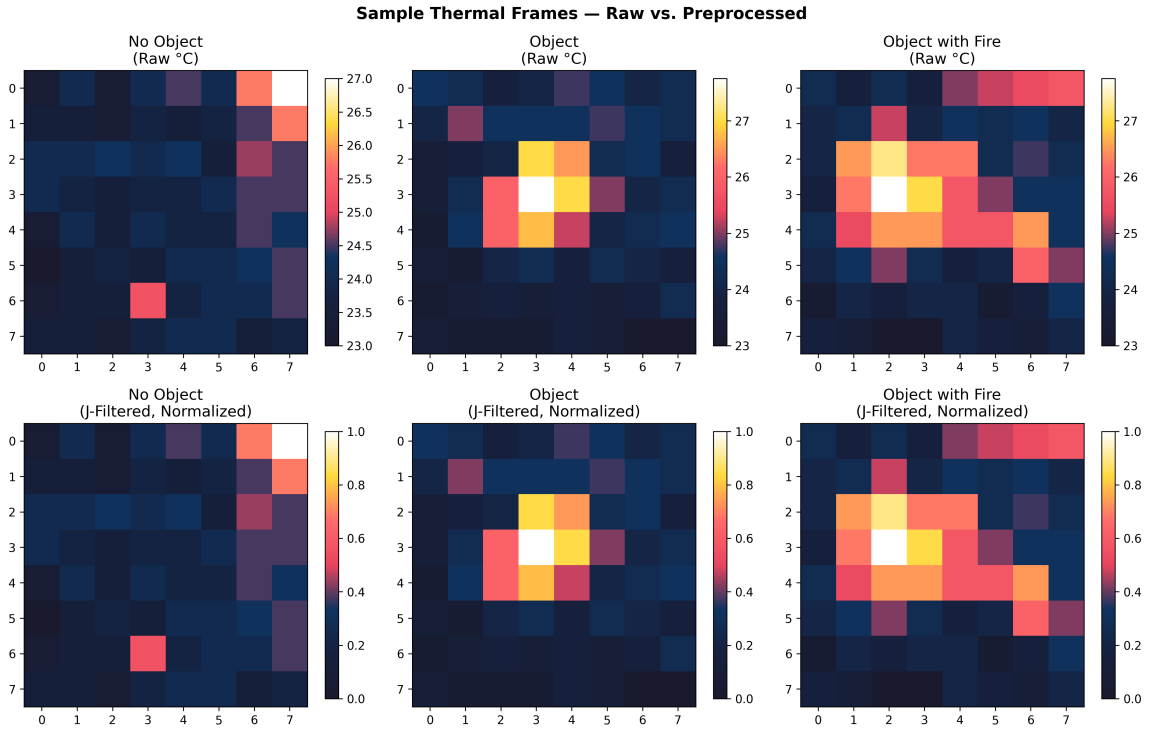


Figure 1: Sample raw and J-filtered thermal heatmaps for each class. Top row: raw sensor output showing absolute temperatures. Bottom row: after J-filter background subtraction, highlighting relative thermal differences. Columns represent No Object, Object, and Object with Fire classes respectively.

3.4 Preprocessing Pipeline

Raw thermal frames from the AMG8833 require preprocessing before they can serve as CNN input. The pipeline consists of three stages: J-filter background subtraction, per-frame normalization, and tensor reshaping. The complete preprocessing flow is illustrated in Figure 2. Taken together, the J-filter and normalization stages securely map the AMG8833 sensor data to a $[0, 1]$ tensor optimized for CNN weight convergence, without distorting the spatial thermal signatures required for accurate localization.

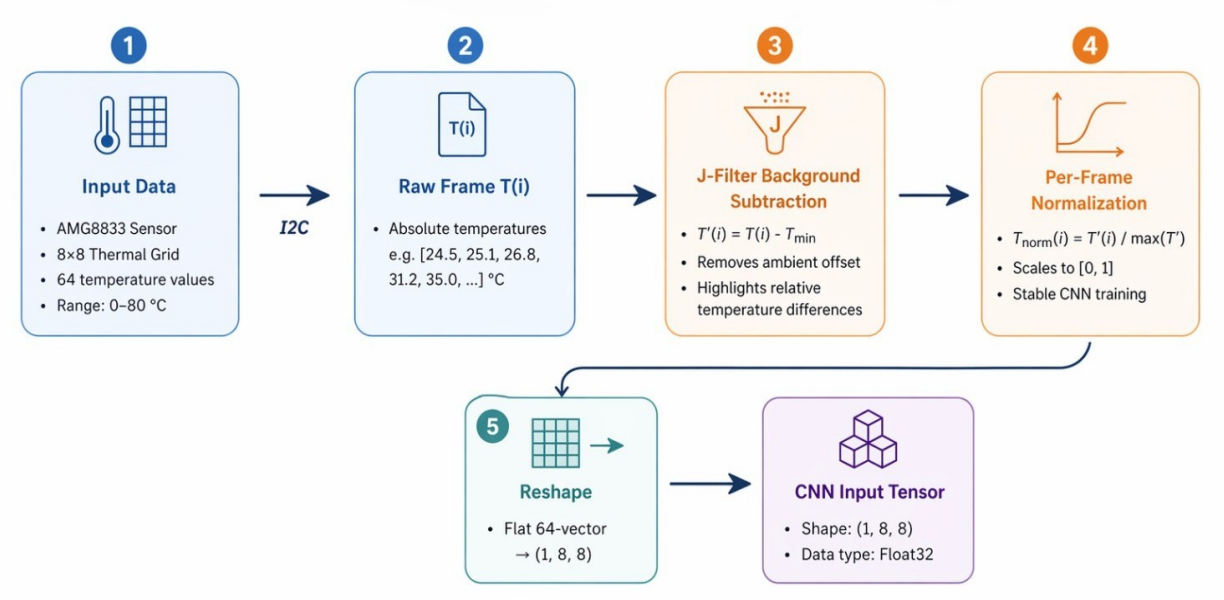


Figure 2: J-Filter preprocessing pipeline: raw thermal frame → background subtraction → per-frame normalization → reshape to CNN input tensor.

3.4.1 J-Filter (Background Subtraction)

The AMG8833 reports absolute temperatures, which means that the same scene recorded at different ambient room temperatures produces different raw values even though the relative heat distribution remains unchanged. A model trained on absolute values would likely fail when room temperature drifts between recording sessions. The J-filter addresses this by subtracting the minimum pixel value from every pixel in each frame:

$$T'(i) = T(i) - T_{\text{min}} \tag{3.1}$$

where $T(i)$ is the raw temperature at pixel i, and $T_{\text{min}} = \min\{T(1), T(2), \ldots, T(64)\}$ is the minimum temperature across all 64 pixels in the frame. After this operation, the coldest pixel reads zero, and all other pixels reflect their temperature elevation above that baseline. The result is a representation that captures relative thermal structure - the spatial pattern of heat sources - independent of ambient conditions.

This approach is deliberately simple. More complex background models (e.g., temporal averaging or Butterworth filtering across frame sequences) were considered but not adopted, since per-frame spatial preprocessing proved sufficient for the classification and localization tasks in this study.

3.4.2 Per-Frame Normalization

After background subtraction, the J-filtered values range from zero to $T_{\max} - T_{\min}$, which varies across frames depending on the intensity of heat sources present. To bring all frames to a common scale, each one is divided by its own maximum value:

$$T_{\text{norm}}(i) = \frac{T'(i)}{\max(T')} \tag{3.2}$$

where $\max(T')$ is the maximum value in the J-filtered frame. If $\max(T') = 0$ (a perfectly uniform frame), the result remains all zeros. This per-frame normalization scales every input to the $[0, 1]$ range, which stabilizes gradient magnitudes during training and promotes consistent weight convergence across the CNN. Because the operation is a simple linear scaling that preserves the relative ordering and spacing of pixel values, it does not distort the spatial thermal signatures that the localization head depends on.

3.4.3 Coordinate Normalization

The ground-truth localization coordinates, originally encoded as integer row and column indices in the range $[0, 7]$, are normalized to $[0, 1]$:

$$x = \frac{\text{row}}{7}, \qquad y = \frac{\text{col}}{7} \tag{3.3}$$

This normalization matches the sigmoid activation output range of the localization head, ensuring that the MSE loss operates on targets and predictions that share the same bounded $[0, 1]$ domain.

3.4.4 Reshape

The preprocessed 64-element flat vector is reshaped into a single-channel 2D image in PyTorch's (C, H, W) format:

$$\mathbf{x} \in \mathbb{R}^{64} \rightarrow \mathbf{X} \in \mathbb{R}^{1\times 8\times 8} \tag{3.4}$$

It is worth emphasizing that no Butterworth filter, temporal smoothing, or spatial interpolation is applied in this pipeline. The CNN operates on the native 8×8 resolution, receiving each frame independently.

3.5 Data Split Strategy

The dataset is partitioned into training, validation, and test sets using stratified sampling to preserve class proportions across all three splits. The allocation follows a 70/10/20 ratio, implemented through two sequential splits with a fixed random seed for reproducibility. Table 4 summarizes the configuration.

Table 4: Data split configuration

Split	Samples	Proportion	Method
Train	∼2,160	70%	Stratified by class
Validation	∼309	10%	Stratified by class
Test	∼618	20%	Stratified by class

The split is performed in two stages. The first stage holds out 20% for testing using `test_size=0.2` and `random_state=42`. The second stage splits the remaining 80% into training and validation using `test_size=0.125` (12.5% of that subset, i.e., 10% of the full dataset). Stratification ensures each split reflects the overall class distribution, which matters given moderate imbalance between the Object class (858 samples) and the other two classes.

The test set is held out entirely during training and used only for final evaluation. The validation set serves two purposes: monitoring generalization during training (via the validation loss, which determines when to save the best model checkpoint) and controlling the learning rate schedule (ReduceLROnPlateau triggers based on validation loss trends). By keeping these roles separate from the test set, the reported final performance reflects a genuinely unseen evaluation.

No data augmentation was applied in this study. Standard image augmentation techniques such as rotation, flipping, or random cropping are designed for natural images where spatial transformations preserve semantic meaning. On an 8×8 thermal grid, however, the spatial structure is tightly coupled to the physical arrangement of the sensor pixels, and the localization ground truth encodes specific row and column indices. Rotating or flipping the grid would require corresponding coordinate transformations that risk introducing inconsistencies, particularly for boundary pixels. The dataset size of 3,087 frames proved sufficient for the model to converge without augmentation, as evidenced by the absence of significant overfitting in the training curves (Chapter 6).

It is also worth noting that each thermal frame represents a single snapshot of the room at a specific moment, recorded independently. No temporal ordering is preserved or exploited in the dataset; each sample is treated as an i.i.d. draw from its class distribution. This simplification enables standard supervised learning but discards potentially useful temporal patterns, which future work could address through sequence-aware models.

4 Proposed Methodology

4.1 System Overview

The proposed system follows an end-to-end pipeline from raw sensor acquisition to dual-task prediction. The AMG8833 sensor transmits a 64-element temperature vector over I^2C at each time step. This vector passes through the J-filter preprocessing pipeline described in Chapter 3 - background subtraction, normalization, and reshape - to produce a single-channel $1 \times 8 \times 8$ tensor. The tensor is then fed into JFilterLocalizationCNN, which simultaneously outputs a 3-class classification prediction and a pair of normalized (x, y) coordinates indicating the location of the dominant heat source.

The model is trained using a combined loss function that sums cross-entropy (for classification) and mean squared error (for localization), with both tasks sharing a convolutional backbone. This design keeps the total parameter count low, since the feature extraction layers are shared rather than duplicated, and allows the two tasks to benefit from common learned representations of spatial heat patterns.

Figure 3 presents the complete system pipeline.

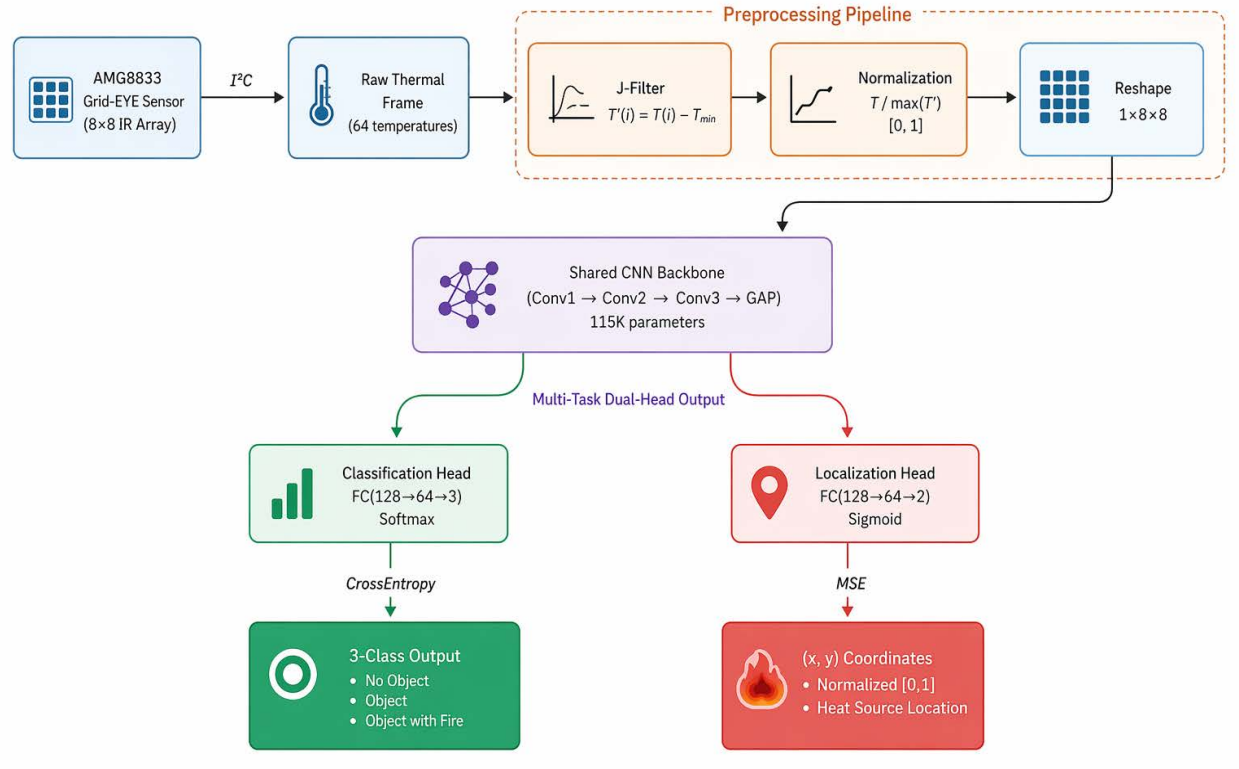


Figure 3: System methodology: end-to-end pipeline from AMG8833 thermal sensor through J-filter preprocessing to the dual-head JFilterLocalizationCNN producing classification and localization outputs.

4.2 JFilterLocalizationCNN Architecture

JFilterLocalizationCNN is a custom VGG-style dual-head CNN designed specifically for the 8×8 input geometry of the AMG8833 sensor. It is not derived from any standard architecture such as ResNet, MobileNet, or EfficientNet. The model is trained from scratch without transfer learning or pre-trained weights, and it operates directly on the native sensor resolution without any spatial upscaling or interpolation.

The architecture consists of a shared convolutional backbone followed by two task-specific heads: one for classification and one for localization. The backbone progressively extracts spatial features through three convolutional blocks, each reducing the spatial dimensions by half through max pooling. After the third block, a global average pooling layer collapses the spatial dimensions entirely, producing a compact 128-dimensional feature vector that serves as input to both heads. The total parameter count is approximately 115,365 - small enough to fit within the memory constraints of microcontroller-class devices.

The choice to build a custom architecture rather than adapting an existing lightweight CNN (such as MobileNet or ShuffleNet) was driven by the extreme input geometry. Standard lightweight architectures assume inputs of at least 224×224 pixels and rely on techniques like depthwise separable convolutions that yield diminishing returns when the spatial dimensions are already at single-digit scales. At 8×8 resolution, the spatial information budget is so constrained that every convolutional layer must count: there is no room for redundant blocks, skip connections to distant layers, or channel expansion strategies designed for much larger feature maps. The VGG-style stacking of small 3×3 filters offers the most non-linearity per parameter at this scale, while the channel progression ($1\rightarrow32\rightarrow64\rightarrow128$) ensures that feature richness increases as spatial resolution decreases.

Inference is a single feed-forward pass: the same 128-dimensional embedding feeds both heads, so classification and localization stay tied to one another at runtime rather than requiring two separate models or two passes over the tensor. The heads are intentionally parallel in depth (two linear stages each) so that neither task dominates the parameter budget, even though their outputs differ in type (discrete logits versus continuous coordinates). This symmetry is a practical choice; it could be revisited if one task were markedly harder than the other in a future dataset.

Inputs are single-channel thermal maps; no RGB fusion and no temporal stack are used in this version. That matches the recorded data and keeps the compute path short, at the cost of ignoring motion cues that a sequence model might exploit later.

Figure 4 illustrates the complete architecture, and Table 5 provides a layer-by-layer breakdown.

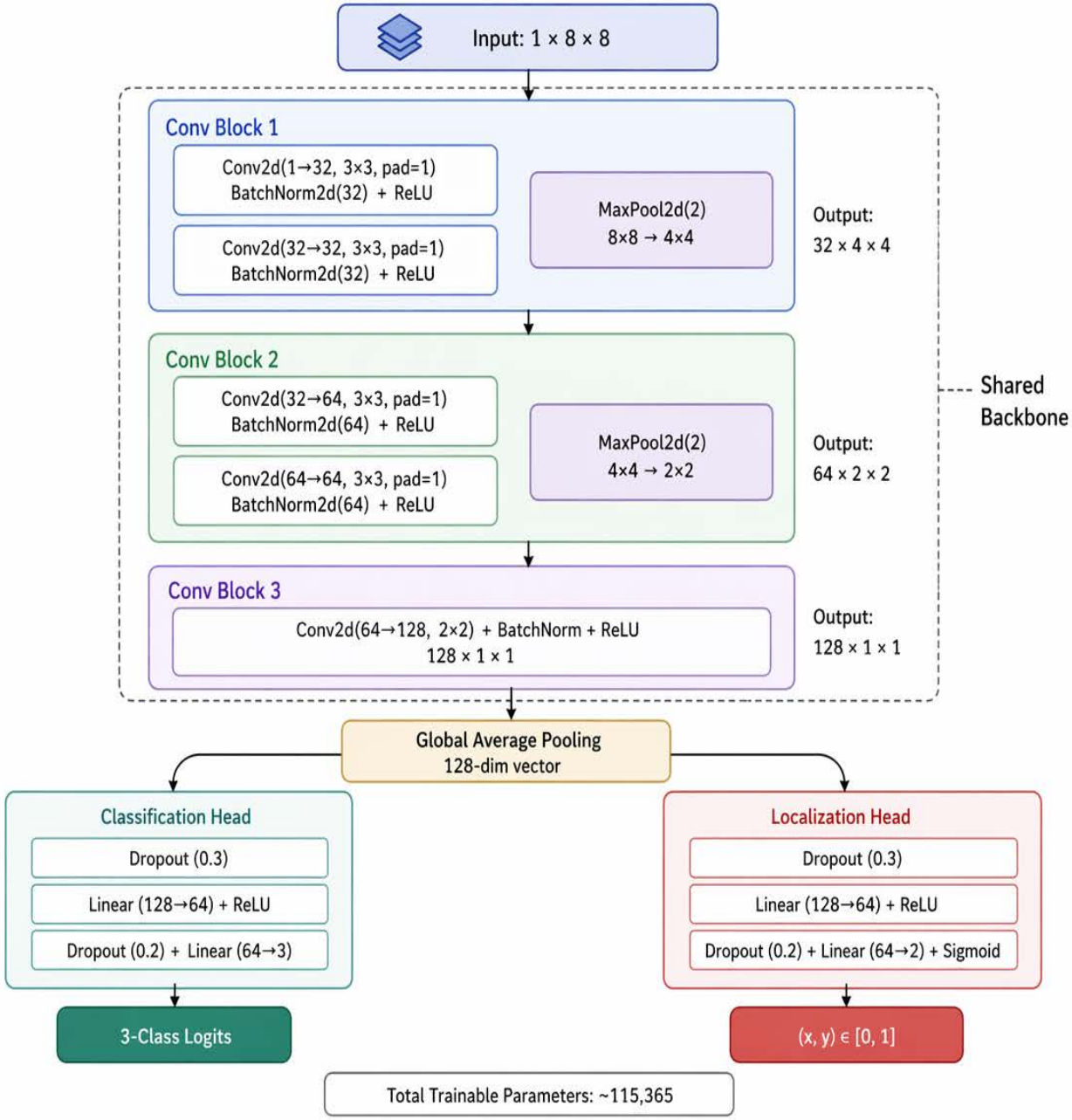


Figure 4: JFilterLocalizationCNN architecture: shared convolutional backbone (3 blocks) with global average pooling, feeding into separate classification and localization heads.

4.2.1 Shared Convolutional Backbone

The backbone follows a VGG-inspired design principle: stacking pairs of small 3×3 convolutions with batch normalization and ReLU activation. Two stacked 3×3 convolutions achieve an effective receptive field of 5×5 with twice the non-linearity and fewer parameters than a single 5×5 kernel. This is particularly valuable at 8×8 resolution, where every additional non-linear transformation helps the network distinguish subtle thermal patterns.

The three blocks are designed to exactly exhaust the spatial dimensions of the input. Starting from 8×8, the first max pool reduces to 4×4, the second to 2×2, and the final convolutional block (using a 2×2 kernel without padding) produces a 1×1 spatial output. Channel depth increases at each stage ($1\rightarrow32\rightarrow64\rightarrow128$) to compensate for the loss of spatial information

Convolutional Block 1

The first block processes the single-channel input:

$$\begin{aligned} X_1 &= \sigma\left(\mathrm{BN}\left(\mathrm{Conv}_{1\to32}^{3\times3}(X_0)\right)\right) \\ X_2 &= \sigma\left(\mathrm{BN}\left(\mathrm{Conv}_{32\to32}^{3\times3}(X_1)\right)\right) \\ X_3 &= \mathrm{MaxPool}_{2\times2}(X_2) \end{aligned} \tag{4.1}$$

Output: $32 \times 4 \times 4$

Convolutional Block 2

The second block doubles the channel depth:

$$\begin{aligned} X_4 &= \sigma\left(\mathrm{BN}\left(\mathrm{Conv}_{32\to64}^{3\times3}(X_3)\right)\right) \\ X_5 &= \sigma\left(\mathrm{BN}\left(\mathrm{Conv}_{64\to64}^{3\times3}(X_4)\right)\right) \\ X_6 &= \mathrm{MaxPool}_{2\times2}(X_5) \end{aligned} \tag{4.2}$$

Output: $64 \times 2 \times 2$.

Convolutional Block 3

The final block uses a 2×2 kernel (matching the remaining spatial dimensions) to collapse the feature map:

$$X_7 = \sigma\left(\mathrm{BN}\left(\mathrm{Conv}_{64\to128}^{2\times2}(X_6)\right)\right) \tag{4.3}$$

Output: $128 \times 1 \times 1$.

A global average pooling layer (AdaptiveAvgPool2d(1)) then produces a 128-dimensional feature vector. At 1×1 spatial resolution, the pooling effectively acts as a squeeze operation. It is retained for architectural consistency and to ensure the model remains well-defined if applied to slightly larger inputs in the future.

$$z = \mathrm{GAP}(X_7),\ z \in \mathbb{R}^{128}$$

4.2.2 Classification Head

The classification branch takes the 128-dimensional feature vector from the shared backbone and maps it to 3-class logits through two fully connected layers with dropout regularization. The first dropout layer (rate 0.3) is applied before the first linear transformation, which projects from 128 to 64 dimensions with ReLU activation. A second dropout (rate 0.2) precedes the final linear layer that maps from 64 to 3 output logits. During inference, a softmax function converts these logits to class probabilities.

$$\hat{y}_{\text{cls}} = W_2 \cdot \mathrm{ReLU}(W_1 \cdot \mathrm{Dropout}_{0.3}(\mathbf{z})) \in \mathbb{R}^3 \tag{4.4}$$

where $W_1 \in \mathbb{R}^{64\times128}$ and $W_2 \in \mathbb{R}^{3\times64}$, with an additional Dropout(0.2) before W_2.

4.2.3 Localization Head

The localization branch shares the same 128-dimensional input and mirrors the classification head's structure: Dropout(0.3), Linear(128→64), ReLU, Dropout(0.2), Linear(64→2). The key difference is the final activation: a sigmoid function constrains the output to $[0, 1]^2$, matching the normalized coordinate range of the ground-truth labels.

$$\hat{y}_{\text{loc}} = \sigma(W_4 \cdot \text{ReLU}(W_3 \cdot \text{Dropout}_{0.3}(\mathbf{z}))) \in [0,1]^2 \tag{4.5}$$

where $W_3 \in \mathbb{R}^{64\times128}$, $W_4 \in \mathbb{R}^{2\times64}$, and σ denotes the element-wise sigmoid. This design means the network never needs to learn coordinate values outside the valid range, which simplifies the optimization landscape for the localization task.

4.2.4 Layer-by-Layer Summary

Table 5: Layer-by-layer summary of JFilterLocalizationCNN

Layer	**Output Shape**	**Parameters**
Input	1 x 8 x 8	–
Conv Block 1		
Conv2d(1 to 32, 3x3, pad=1)	32 x 8 x 8	320
BatchNorm2d(32)	32 x 8 x 8	64
Conv2d(32 to 32, 3x3, pad=1)	32 x 8 x 8	9,248
BatchNorm2d(32)	32 x 8 x 8	64
MaxPool2d(2)	32 x 4 x 4	–
Conv Block 2		
Conv2d(32 to 64, 3x3, pad=1)	64 x 4 x 4	18,496
BatchNorm2d(64)	64 x 4 x 4	128
Conv2d(64 to 64, 3x3, pad=1)	64 x 4 x 4	36,928
BatchNorm2d(64)	64 x 4 x 4	128
MaxPool2d(2)	64 x 2 x 2	–
Conv Block 3		
Conv2d(64 to 128, 2x2)	128 x 1 x 1	32,896
BatchNorm2d(128)	128 x 1 x 1	256
AdaptiveAvgPool2d(1)	128	–
Classification Head		
Dropout(0.3) + Linear(128 to 64)	64	8,256
Dropout(0.2) + Linear(64 to 3)	3	195
Localization Head		
Dropout(0.3) + Linear(128 to 64)	64	8,256
Dropout(0.2) + Linear(64 to 2) + Sigmoid	2	130
Total Trainable Parameters		**~115,365**

4.3 Loss Functions

Training a dual-head model requires a composite loss function that accounts for both tasks. Each head has its own loss term, and the two are summed to form the total training objective.

4.3.1 Classification Loss

The classification head is trained using cross-entropy loss, which penalizes incorrect class predictions proportionally to the confidence of the incorrect prediction:

$$\mathcal{L}_{\text{CE}} = -\sum_{c=1}^{3} y_c \log(\hat{p}_c) \tag{4.6}$$

where y_c is the one-hot encoded ground truth and $\hat{p}_c$ is the predicted probability for class c.

4.3.2 Localization Loss

The localization head is trained using mean squared error, which penalizes deviations between predicted and ground-truth coordinates:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{2} \sum_{d \in \{x,y\}} (\hat{y}_d - y_d)^2 \tag{4.7}$$

where $\hat{y}_d$ and y_d are the predicted and ground-truth normalized coordinates.

4.3.3 Total Loss

The total training loss is the unweighted sum of both task-specific losses:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{MSE}} \tag{4.8}$$

Equal weighting is used because the two loss terms happen to operate on comparable numerical scales in practice: both are small positive values that decrease during training. More sophisticated approaches - such as uncertainty-based task weighting or learned loss balancing - could potentially improve performance, but the simple sum proved effective and avoids introducing additional hyperparameters.

4.4 Architecture Design Justification

Each design decision in JFilterLocalizationCNN was driven by the constraints of the 8×8 input and the target deployment platform. Table 6 summarizes the rationale behind the key architectural choices.

Table 6: Design choices and their justifications

Design Choice	**Justification**
3x3 kernels	Maximizes feature extraction depth on tiny 8x8 input; two stacked 3x3 gives an effective 5x5 receptive field with more non-linearity
3 conv blocks only	Exactly matches the 8x8 resolution: spatial dimensions are fully exhausted at 8, 4, 2, 1
Channel expansion	Channels increase 32, 64, 128 to compensate for spatial information loss with richer feature representations
Global Average Pooling	Reduces parameters compared to flattening; acts as spatial regularizer
BatchNorm	Stabilizes training on small-scale inputs; enables higher learning rates
Dropout (0.3 + 0.2)	Prevents overfitting on approximately 3,087 samples
Dual-head design	More parameter-efficient than two separate models; shared features benefit both tasks
Sigmoid on localization	Constrains coordinate output to [0, 1], matching normalized ground truth
Approx. 115K parameters	Small enough for potential embedded deployment on XIAO ESP32

Several alternative design choices were considered during development. Residual connections were evaluated but not adopted; with only three convolutional blocks and a total depth of six convolutional layers, the gradient path is short enough that the vanishing gradient problem does not arise. Adding skip connections would introduce additional parameters and memory overhead without a clear training benefit at this depth. Depthwise separable convolutions were also considered, following the MobileNet paradigm, but the spatial dimensions are too small for the channel-wise and spatial-wise factorization to provide meaningful computational savings.

The activation function throughout the backbone is ReLU, chosen for its computational simplicity and well-understood gradient behavior. Alternatives such as Leaky ReLU, GELU, or Swish were not explored, since ReLU has proven sufficient for networks of this scale and avoids introducing additional hyperparameters. The sigmoid activation on the localization head is a deliberate design choice that constrains the output to the valid coordinate range $[0, 1]$, eliminating the need for output clipping during inference and providing a smooth gradient signal throughout the target range.

The justification table should be read as a set of coupled decisions rather than independent toggles. For example, batch normalization and dropout address different failure modes - unstable activations on tiny maps versus overfitting on a few thousand frames - but both assume a training regime with fixed random seeds and stratified splits; change the data regime and the balance between them might need to shift. Global average pooling is paired with the narrow channel growth precisely because the spatial map disappears after the third block: once the tensor is 1×1, pooling is almost a formality, yet keeping it preserves a clear separation between "convolutional feature extraction" and "fully connected reasoning" in the implementation.

From a deployment perspective, the 115K-parameter figure is an order-of-magnitude statement, not a guarantee on any specific board. SRAM limits, tensor arena sizing in TinyML runtimes, and whether weights sit in flash or RAM all affect what "fits" in practice. The architecture is small enough that quantization (e.g., INT8) appears plausible, but that step would need its own validation because small networks can sometimes lose more accuracy per bit than large ones when ranges are tight.

Finally, the design deliberately avoids exotic modules - no attention, no deformable convolution, no custom CUDA kernels - because the bottleneck here is input resolution, not raw FLOPs on a server GPU. If the sensor geometry ever changes (e.g., a denser array), the block count and kernel schedule would need to be re-derived rather than scaled by a fixed rule of thumb.

5 Experimental Design

5.1 Implementation Details

The model was implemented in PyTorch and trained on a desktop workstation. All random seeds (NumPy and PyTorch) were fixed to 42 to ensure reproducibility across runs. The data loading pipeline uses PyTorch's `DataLoader` with the `ThermalLocalizationDataset` custom class, which returns each sample as a tuple of $(X, y_{\text{class}}, y_{\text{coords}})$ - the preprocessed thermal tensor, the integer class label, and the normalized (x, y) coordinate pair.

During training, the model with the lowest total validation loss (not highest accuracy) is saved to disk. This criterion was chosen because training classification accuracy reaches 100% by around epoch 15, after which the validation loss is driven primarily by localization error. Saving based on loss rather than accuracy ensures that the best checkpoint reflects continued improvement in coordinate prediction.

5.2 Training Configuration

The training setup uses the AdamW optimizer, which decouples weight decay from the gradient update step. This distinction matters for regularization: standard Adam applies weight decay as part of the gradient, which can interact poorly with adaptive learning rates. AdamW applies it separately, leading to more consistent regularization behavior across parameters with different gradient magnitudes.

The learning rate scheduler (ReduceLROnPlateau) monitors the total validation loss and halves the learning rate if no improvement is observed for 5 consecutive epochs. This adaptive strategy allows the model to make large steps early in training and progressively finer adjustments as it approaches convergence. Table 7 summarizes the complete configuration.

Table 7: Training hyperparameters

Parameter	**Value**
Framework	PyTorch
Optimizer	AdamW
Learning Rate	0.001
Weight Decay	0.01
Scheduler	ReduceLROnPlateau (patience=5, factor=0.5)
Batch Size	32
Epochs	50
Random Seed	42
Model Saving Criterion	Lowest total validation loss

A batch size of 32 was selected as a balance between gradient noise and memory efficiency. Smaller batches introduce more stochastic variation per update, which can act as implicit regularization on a small dataset like ours. Larger batches would smooth the gradient estimates but risk reducing this beneficial noise on only ~2,160 training samples.

5.3 Evaluation Metrics

Model performance is assessed using separate metrics for the classification and localization tasks. Classification metrics evaluate discrete label prediction quality, while localization metrics quantify the spatial accuracy of coordinate regression.

5.3.1 Classification Metrics

Accuracy

Overall classification accuracy measures the fraction of correctly classified test samples:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total predictions}} \times 100\% \tag{5.1}$$

Precision

Per-class precision quantifies the proportion of true positives among all samples predicted as that class:

$$P_c = \frac{TP_c}{TP_c + FP_c} \tag{5.2}$$

Recall

Per-class recall measures the proportion of actual positives correctly identified by the model:

$$R_c = \frac{TP_c}{TP_c + FN_c} \tag{5.3}$$

F1-Score

The harmonic mean of precision and recall, balancing the two metrics into a single number:

$$F1_c = 2 \cdot \frac{P_c \cdot R_c}{P_c + R_c} \tag{5.4}$$

Where:

TP_c = True Positives for class c

FP_c = False Positives for class c

FN_c = False Negatives for class c

5.3.2 Localization Metrics

Mean Squared Error (MSE)

Measures the average squared difference between predicted and true normalized coordinates:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} \left[(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2 \right] \tag{5.5}$$

Mean Absolute Error (MAE)

Measures the average Euclidean distance in pixel coordinates, providing a more interpretable measure of spatial accuracy:

$$\text{MAE}_{\text{px}} = \frac{1}{N} \sum_{i=1}^{N} \sqrt{(7\hat{x}_i - 7x_i)^2 + (7\hat{y}_i - 7y_i)^2} \tag{5.6}$$

where the factor of 7 converts normalized coordinates back to the $[0, 7]$ pixel range of the 8×8 grid.

Root Mean Squared Error (RMSE)

$$\text{RMSE}_{\text{px}} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} [(7\hat{x}_i - 7x_i)^2 + (7\hat{y}_i - 7y_i)^2]} \tag{5.7}$$

RMSE penalizes large errors more heavily than MAE, making it sensitive to outlier predictions. The difference between MAE and RMSE provides an indication of how uniformly distributed the localization errors are across test samples.

5.4 Training Rationale and Design Decisions

Several aspects of the training configuration warrant further explanation, as they reflect deliberate choices rather than arbitrary defaults.

The AdamW optimizer was preferred over standard SGD with momentum because the adaptive learning rate per parameter is well suited to a multi-task loss landscape where different parameter groups (backbone, classification head, localization head) may have different gradient magnitudes. Decoupled weight decay provides consistent L2 regularization regardless of these differences, which is important when the classification loss drops to near zero within the first 20 epochs while the localization loss continues to provide gradients.

The ReduceLROnPlateau scheduler was chosen over fixed-step or cosine annealing schedules because the training dynamics of this model are not easily predicted in advance. Classification converges quickly, while localization improves gradually; a fixed schedule might reduce the learning rate too early or too late relative to the localization task's needs. Monitoring the total validation loss and reducing the learning rate only when progress stalls allows the optimizer to adapt to the actual convergence behavior of this particular model-data combination.

The choice of 50 training epochs was determined empirically. Preliminary experiments showed that classification accuracy stabilized by epoch 15 and localization MSE showed diminishing returns after epoch 40. Running for 50 epochs ensures that the localization head has sufficient time to converge without excessively prolonging training. On the desktop workstation used for this study, 50 epochs complete in under two minutes, making computational cost a non-issue.

Weight initialization follows PyTorch's default Kaiming initialization for convolutional layers, which accounts for the ReLU activation function by scaling initial weights to preserve gradient variance across layers. No special initialization strategy was needed; the relatively shallow network depth and batch normalization layers ensure stable gradients from the beginning of training.

6 Results and Analysis

6.1 Training Convergence

The model was trained for 50 epochs, with the best checkpoint saved at the epoch that achieved the lowest total validation loss. Training convergence was fast: training classification accuracy reached 100% by epoch 15 and remained there for subsequent epochs, while validation accuracy stabilised at 99.68%, peaking at 100% in several epochs. The classification cross-entropy loss dropped below 0.001 within approximately 20 epochs, effectively becoming negligible.

Localization, by contrast, improved more gradually. The training MSE decreased from 0.0387 at epoch 1 to 0.0046 at epoch 50 - roughly an eightfold reduction. Validation MSE followed a similar trajectory, dropping from 0.0187 to 0.0104, though it showed the expected slight divergence from training MSE in later epochs. This divergence remained small, suggesting that the model did not overfit significantly despite the relatively small dataset size.

The gap between training and validation loss in the final epochs is driven almost entirely by the localization component; the classification loss is near zero on both sets. This pattern makes intuitive sense: classifying a 3-class problem with distinct thermal signatures is an easier task than precisely regressing continuous coordinates, and the localization head continues to refine its predictions long after classification has been solved.

Figure 5 shows the training and validation loss curves across all 50 epochs.

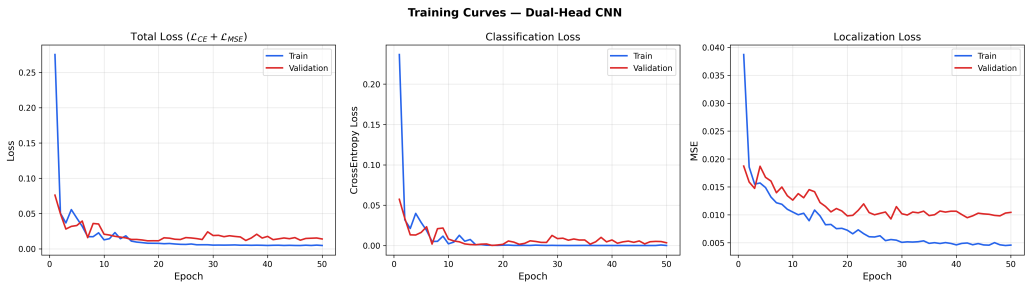


Figure 5: Training and validation loss curves over 50 epochs, showing total loss, classification cross-entropy loss, and localization MSE loss for both training and validation sets.

Figure 6 presents the classification accuracy convergence.

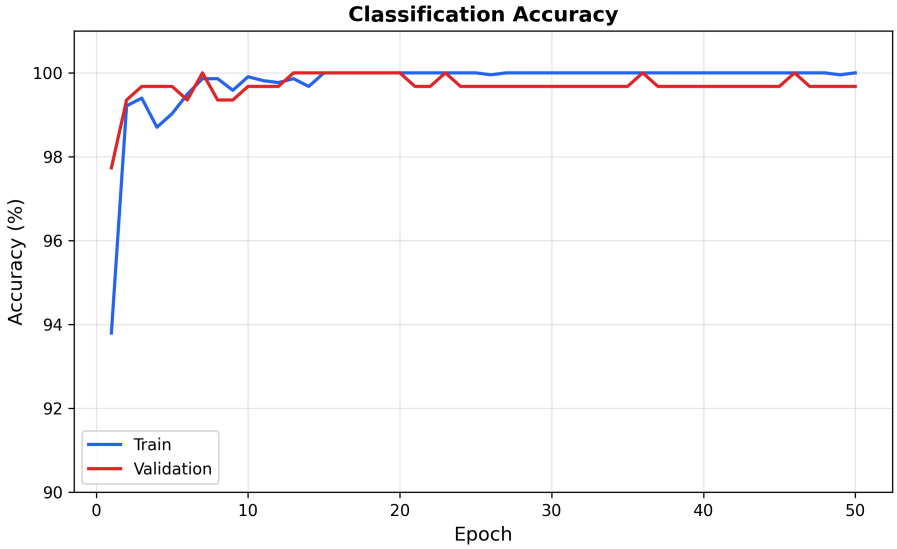


Figure 6: Classification accuracy over training epochs for both training and validation sets. Training accuracy reaches 100% by epoch 15; validation accuracy stabilises at 99.68%.

Figure 7 tracks the localization MSE throughout training, showing the steady improvement of coordinate prediction accuracy.

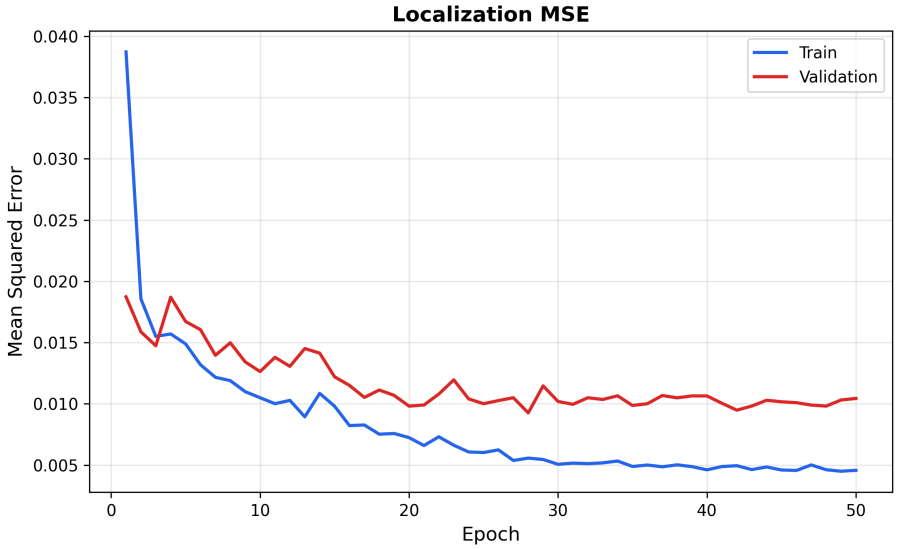


Figure 7: Localization MSE over training epochs. Training MSE steadily decreases from 0.0387 to 0.0046 across 50 epochs, while validation MSE stabilizes around 0.0104.

Figure 8 illustrates how the relative contribution of the two loss components shifts during training. Classification loss dominates early on, but once the classification task is effectively solved (around epoch 15), localization loss becomes the sole driver of further optimization.

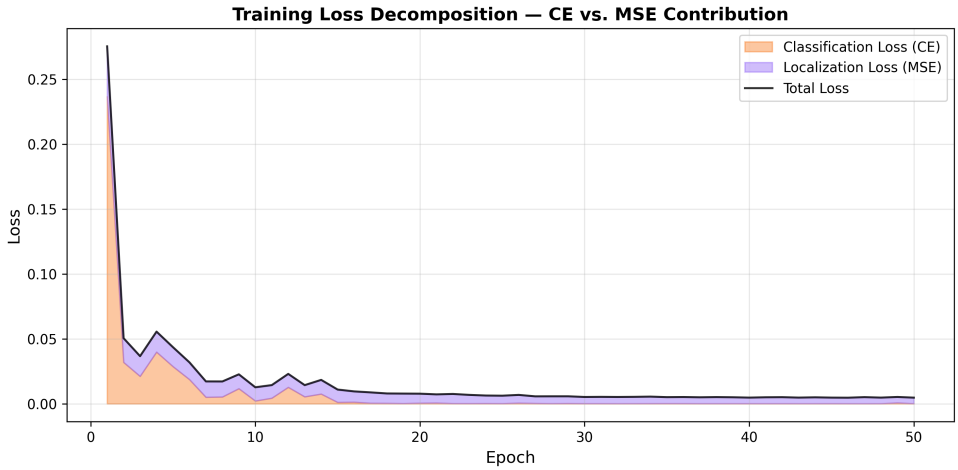


Figure 8: Loss decomposition: stacked area plot showing the contribution of cross-entropy (CE) and MSE losses to the total training loss across epochs. After approximately epoch 15, the CE loss is negligible and localization MSE dominates.

Table 8 summarizes key training metrics at selected epochs.

Table 8: Training convergence at selected epochs

Epoch	Train Loss	Val Loss	Train Acc	Val Acc	Train MSE	Val MSE
1	0.2753	0.0761	93.80%	97.73%	0.0387	0.0187
5	0.0436	0.0330	99.03%	99.68%	0.0149	0.0167
10	0.0127	0.0207	99.91%	99.68%	0.0105	0.0126
20	0.0078	0.0114	100.00%	100.00%	0.0072	0.0098
30	0.0052	0.0188	100.00%	99.68%	0.0051	0.0102
40	0.0047	0.0176	100.00%	99.68%	0.0046	0.0106
50	0.0047	0.0139	100.00%	99.68%	0.0046	0.0104

6.2 Classification Performance

On the held-out test set of 618 samples, the model achieves an overall classification accuracy of 99.68%. This corresponds to approximately 2 misclassified samples out of 618. The near-perfect result indicates that the three thermal event classes - No Object, Object, and Object with Fire - produce sufficiently distinct spatial heat patterns on the 8×8 grid for the CNN to discriminate with high confidence.

6.2.1 Confusion Matrix

Figure 9 shows the test set confusion matrix. The diagonal entries dominate overwhelmingly, with only a handful of off-diagonal errors. The misclassifications do not cluster in any particular pair of classes, suggesting that no two classes are systematically confused.

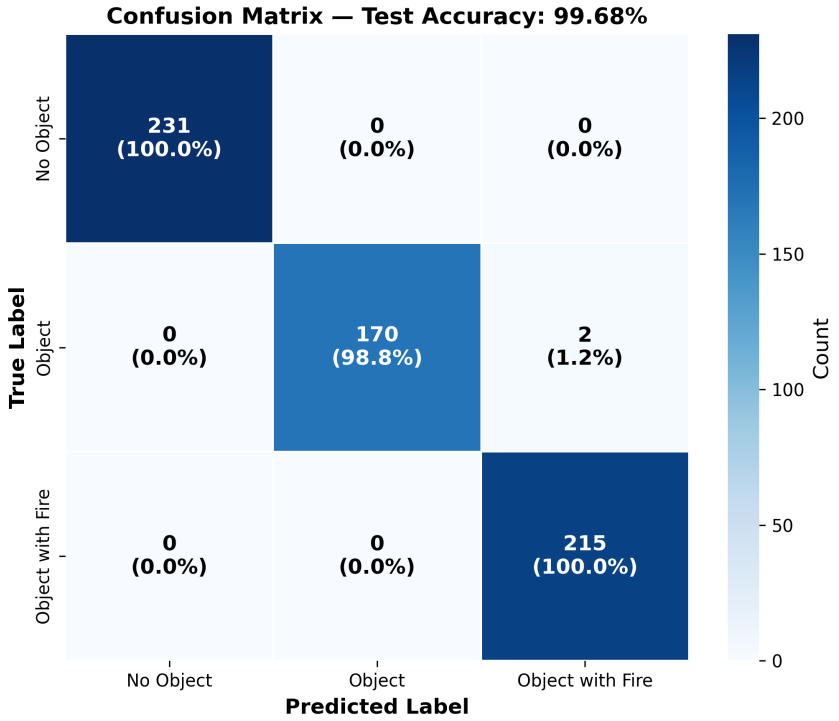


Figure 9: Confusion matrix on the test set (618 samples). The model achieves 99.68% overall accuracy with minimal misclassifications across all class pairs.

6.2.2 Per-Class Metrics

Figure 10 presents a heatmap visualization of precision, recall, and F1-score for each class. All three classes achieve F1-scores above 0.99, confirming that the high overall accuracy is not an artifact of class imbalance - each class is individually well-recognized.

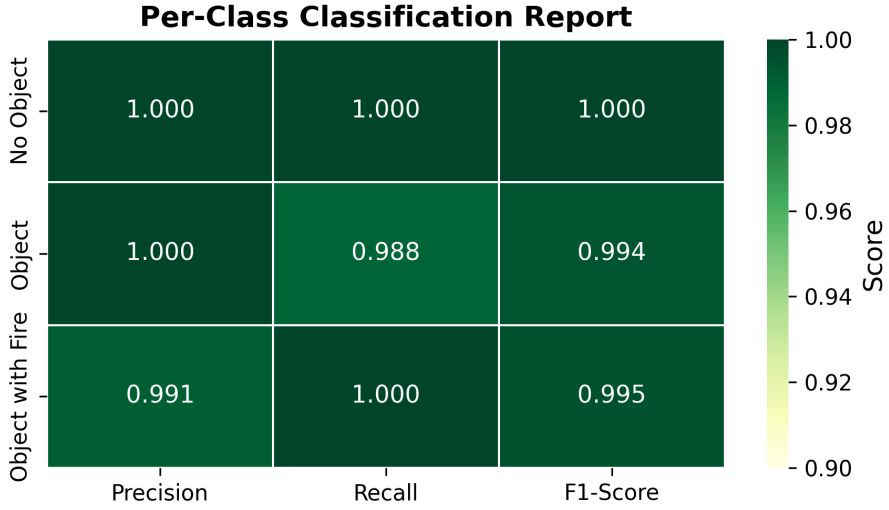


Figure 10: Classification report heatmap showing per-class precision, recall, and F1-score on the test set. All classes exceed 0.99 F1-score.

The consistently high per-class metrics deserve some qualification. The dataset was collected in a single controlled indoor environment, and the three classes represent fairly distinct thermal scenarios. A person produces a warm region that is absent in empty-room frames, and a fire source

creates a localized hot spot that is markedly different from body heat alone. In a more diverse or adversarial setting - multiple rooms, varying sensor distances, or subtle heat sources - these metrics could change. The results presented here establish what the model can achieve under controlled conditions; generalization to unconstrained environments remains an open question.

6.3 Localization Performance

The localization head achieves a normalized MSE of 0.0091 on the test set, which translates to a mean absolute error (MAE) of 0.37 pixels on the 8×8 grid. The RMSE is 0.67 pixels. Sub-pixel accuracy at this resolution is a strong result - the model's predictions land, on average, well within half a pixel of the true heat-source location.

Table 9 summarizes the localization metrics.

Table 9: Localization performance on the test set

Metric	Value
MSE (normalized)	0.0091
MAE (normalized)	0.0522
MAE (pixels)	0.37
RMSE (pixels)	0.67

The relatively small gap between MAE (0.37 px) and RMSE (0.67 px) indicates that the error distribution is fairly uniform, without many large outliers pulling the RMSE up. If the model were occasionally making large errors on specific samples while performing well on others, the RMSE would be substantially higher than the MAE.

6.3.1 Predicted vs. True Coordinates

Figure 11 visualizes the predicted versus true localization coordinates, with error lines connecting each prediction to its ground truth.

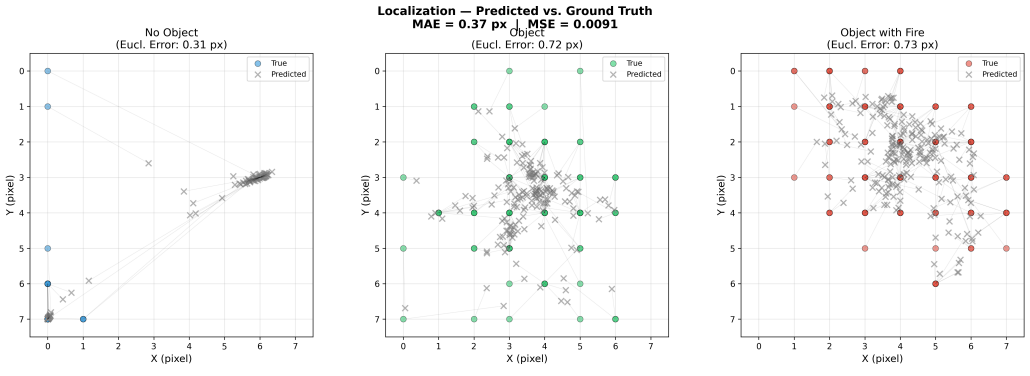


Figure 11: Scatter plot of predicted vs. true localization coordinates on the test set. Each point represents a test sample, color-coded by class. Short error lines indicate predictions that closely match ground truth.

The scatter plot reveals that predictions cluster tightly around the ground-truth positions for all three classes. The error lines are uniformly short, consistent with the sub-pixel MAE reported above.

6.3.2 Per-Class Localization Error

Figure 12 shows the distribution of Euclidean localization errors for each class.



Figure 12: Box plot of per-class Euclidean localization error (in pixels). Median and mean values are annotated for each class.

The No Object class produces the smallest mean localization error (0.31 px, median 0.02 px), because in most empty-room frames the hottest background pixel occupies a consistent location that the model learns reliably. However, this class also shows the largest variance: in frames where several background pixels share near-equal temperatures, the "hottest" pixel becomes ambiguous, producing occasional outliers above 5 px. The Object and Object with Fire classes show similar mean errors (0.72 and 0.73 px, respectively) with tighter distributions, reflecting the inherent difficulty of pinpointing the centre of a diffuse warm region that spans multiple pixels. Even so, the median error for all three classes remains below one pixel.

6.4 Combined Results Summary

Table 10 presents the complete test set performance summary.

Table 10: Combined test set performance summary

Metric	Test Set
Classification Accuracy	99.68%
Localization MSE (normalized)	0.0091
Localization MAE (pixels)	0.37
Localization RMSE (pixels)	0.67
Total Trainable Parameters	∼115,365

These results demonstrate that a compact CNN can extract meaningful spatial features from 8×8 thermal data. The classification task, in particular, appears to be well within the model's capacity - the three classes are distinguishable with near-perfect accuracy under controlled conditions.

Localization performance is also strong, with sub-pixel accuracy suggesting that the spatial heat patterns carry enough structure for precise coordinate regression.

No directly comparable baseline exists for this specific task, since prior studies on AMG8833-class sensors have not applied learned models for joint classification and localization. The results presented here therefore serve as a first benchmark. Future work comparing against alternative architectures (e.g., adapted MobileNet, simple fully connected networks, or SVM-based approaches) on the same dataset would provide a more complete picture of where this architecture sits in the performance landscape.

6.5 Discussion

Several aspects of the results merit additional discussion regarding their significance, limitations, and implications for practical deployment.

The near-perfect classification accuracy (99.68%) is notable but must be interpreted in context. The three classes produce qualitatively different thermal patterns on the 8×8 grid: an empty room shows a relatively uniform temperature field, a person introduces a broad warm region spanning several pixels, and a fire source creates a sharp localized hot spot. These differences are large relative to sensor noise and ambient variation within the controlled recording environment. The high accuracy reflects the separability of these particular scenarios more than any general claim about CNN capability on thermal data. In settings with more subtle distinctions - for example, distinguishing between a person and a large pet, or between a fire and a heated appliance - the task would be considerably harder.

The sub-pixel localization accuracy (0.37 px MAE) deserves careful interpretation as well. On an 8×8 grid, each pixel covers a substantial physical area determined by the sensor's field of view and mounting distance. A localization error of 0.37 pixels is impressive from a signal processing perspective - the network recovers coordinate information that is finer than the native sensor resolution - but the physical meaning of sub-pixel precision depends on the deployment geometry. At typical indoor mounting distances, one pixel corresponds to several centimeters, so sub-pixel accuracy on the grid translates to meaningful spatial precision in the physical world.

The model's parameter count ($\sim$115K) places it in a different category from the lightweight architectures reviewed in Chapter 2. MobileNetV2 has approximately 3.4 million parameters, ShuffleNet V2 approximately 2.3 million, and even the smallest EfficientNet variant exceeds 5 million. Our model is roughly 20 to 40 times smaller than these architectures, a reduction made possible by the extremely small input size and the correspondingly shallow network depth. This compact size is not merely an academic curiosity; it directly determines whether the model can fit in the SRAM of a microcontroller alongside the operating system, sensor drivers, and communication stack.

The equal weighting of classification and localization losses ($\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{MSE}}$) worked well in practice, but the training dynamics reveal an implicit scheduling effect. Early in training, the classification loss is large and dominates the gradient, steering the shared backbone toward features that distinguish the three classes. After classification converges (around epoch 15), the classification loss becomes negligible, and the backbone is fine-tuned exclusively by localization gradients. This natural curriculum - first learn what, then learn where - may partially explain the strong performance on both tasks, even without explicit loss balancing.

6.6 Cross-Dataset Generalization

A strong in-distribution result on one dataset does not, by itself, demonstrate that the learned representations generalize. To test whether the convolutional backbone captures transferable thermal features, we evaluated the pretrained JFilterLocalizationCNN on two external datasets that differ in sensor hardware, collection environment, and labelling protocol. Because neither external dataset contains a fire class, we cast the evaluation as a binary task - *No Object* versus *Object* - and merged the model's softmax outputs for classes 1 and 2 into a single Object probability: $P(\text{Object}) = \text{softmax}[1] + \text{softmax}[2]$.

6.6.1 External Datasets

Table 11 summarizes the key differences between the three datasets used in this evaluation.

Table 11: Comparison of dataset properties

Property	AMG8833 (Ours)	LINAIGE	MLX90640
Sensor	AMG8833 Grid-EYE	8×8 IR Array	MLX90640
Native resolution	8×8	8×8	32×24
Data format	Raw temperature	Raw temp. (CSV)	Colourized JPEG
Samples used	3,087	5,000 (balanced)	384
Binary classes	N/A (3-class)	No Obj. / Object	No Obj. / Object
Domain gap	-	Ambient shift	Colourmap + resol.

LINAIGE [23] provides 31,036 frames from a native 8×8 infrared array collected across five sessions in three rooms. Each frame carries a person-count label (0–3) and a thermistor reading (sensor die temperature). We mapped count $= 0$ to No Object and count ≥ 1 to Object, then drew a balanced random subset of 5,000 samples (2,500 per class), preferring frames with high labeller confidence.

MLX90640 Thermal Image Dataset contains 384 images (189 human, 195 non-human) captured by a 32×24 MLX90640 sensor. The images are stored as colourized JPEG visualizations rather than raw temperature arrays, so we converted each frame to grayscale and downsampled to 8×8 via Lanczos interpolation before applying the J-filter. This colourmap-to-grayscale conversion introduces a domain shift that is absent in the LINAIGE experiments, and the smaller sample size (384 vs. 5,000) limits statistical power.

6.6.2 Preprocessing and Distribution Analysis

Both external datasets were preprocessed with the same J-filter pipeline used for the AMG8833 data: $T'(i) = T(i) - T_{\text{min}}$ followed by per-frame normalization to $[0, 1]$. An early experiment with thermistor-based background subtraction for LINAIGE - replacing T_{min} with the on-board thermistor reading - failed because the thermistor measures the sensor die temperature (29.6°C), which exceeds all pixel values (22–28°C). Subtracting the die temperature zeroed out the entire frame, producing uninformative inputs.

For zero-shot evaluation, we applied distribution alignment: the external data was linearly transformed so that its global mean and standard deviation matched the AMG8833 training statistics ($\mu_{\text{AMG}} = 0.292$, $\sigma_{\text{AMG}} = 0.218$). This step compensates for systematic shifts in the pixel value distribution without changing the spatial patterns within each frame. For fine-tuning and k-fold

experiments, we used the unaligned preprocessed data, allowing the model to adapt its internal representations freely.

Figure 13 shows the pixel value distributions for all three datasets after J-filter and normalization. The AMG8833 distribution is concentrated near zero (most pixels in the 8×8 grid are close to the background minimum), while LINAIGE has a broader, higher-mean distribution - consistent with warmer ambient environments or different spatial temperature patterns. The MLX90640 distribution sits highest, an artifact of the grayscale conversion from colourized images rather than a reflection of physical temperature differences.

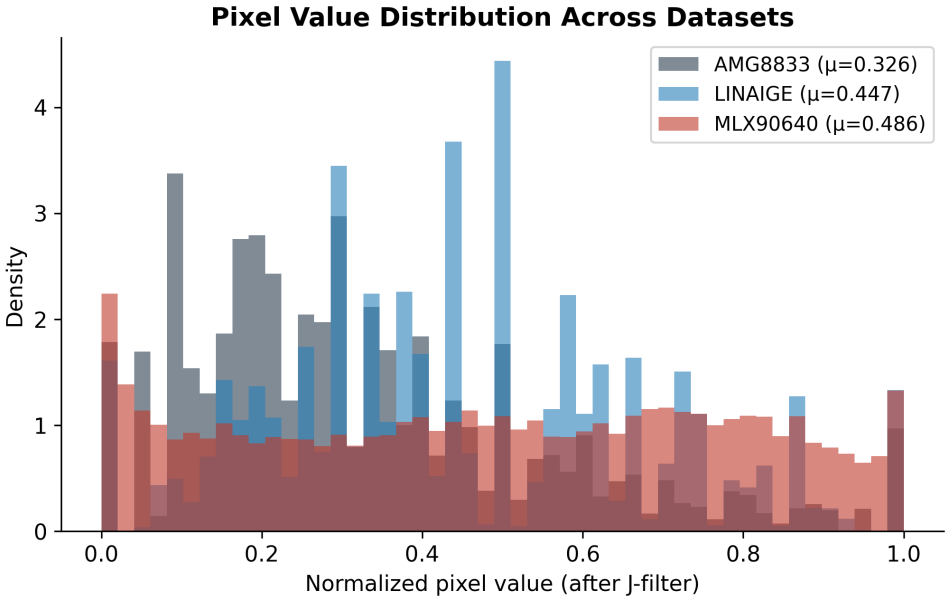


Figure 13: Pixel value distributions after J-filter and normalization. The AMG8833 training data concentrates near zero; LINAIGE and MLX90640 are shifted rightward, explaining the domain shift that degrades zero-shot performance.

Figure 14 presents sample 8×8 heatmaps from each dataset. The AMG8833 frames show sharp, concentrated warm regions (person or fire), while LINAIGE frames exhibit a more diffuse thermal profile - multiple pixels carry moderate intensity even in single-person scenes. The MLX90640 frames, derived from downsampled colourized JPEGs, produce distinctly different spatial patterns that bear little resemblance to raw thermal data.

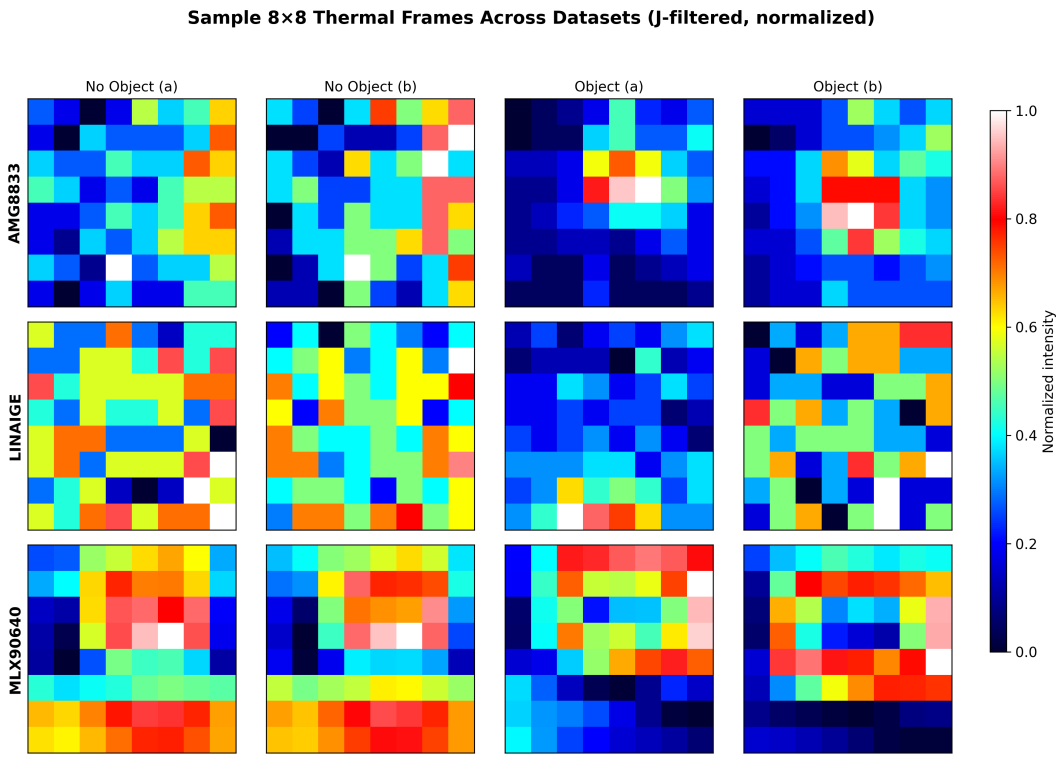


Figure 14: Sample 8×8 thermal frames from each dataset (J-filtered, normalized). Each row corresponds to one dataset; columns alternate between No Object and Object classes. The visual differences between rows illustrate the domain shift that the model must overcome.

6.6.3 Zero-Shot Evaluation

Table 12 reports the zero-shot results. The pretrained model was applied without any weight updates; only the softmax merging step adapted the 3-class output to a binary decision boundary.

Table 12: Zero-shot binary classification on external datasets

Dataset	Samples	Accuracy	Precision	Recall	F1
LINAIGE	5,000	60.74%	66.50%	60.74%	56.99%
MLX90640	384	48.44%	34.41%	49.20%	33.07%

Both datasets fall well below the in-distribution baseline. The LINAIGE result (61%), obtained after distribution alignment, is above chance but far from usable; MLX90640 sits near random (48%). The low F1 scores (57% and 33%) indicate that the model's predictions are not merely noisy but systematically biased - a point that the confusion matrices make concrete.

Figure 15 presents the zero-shot confusion matrices. On LINAIGE the model biases toward the Object class, misclassifying 68.8% of No Object frames. The model has learned that "background" looks like an AMG8833 empty room; LINAIGE empty rooms, collected in different buildings at different ambient temperatures, produce J-filtered patterns that fall outside this learned template. On MLX90640 the bias is even more extreme: 99% of all frames are predicted as Object. The grayscale intensity distribution of downsampled colourized images evidently never triggers the No Object pathway that the model learned from raw thermal data.

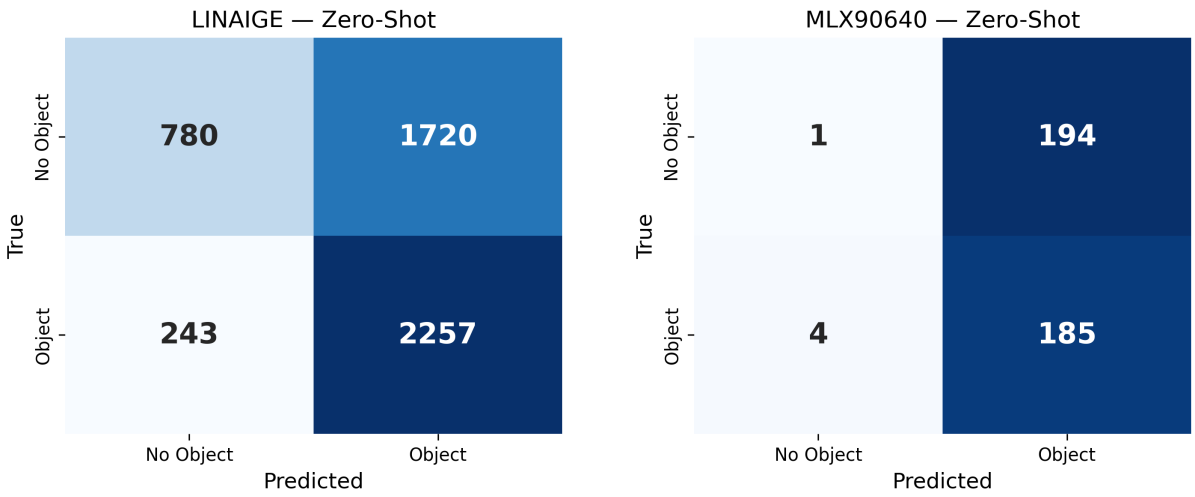


Figure 15: Zero-shot confusion matrices. Left: LINAIGE (5,000 samples). Right: MLX90640 (384 samples). The model biases heavily toward predicting Object on both external distributions.

6.6.4 Fine-Tuning Transfer

To test whether the pretrained backbone encodes reusable low-level thermal features, we replaced the 3-class classification head with a 2-class head and fine-tuned in two stages: five epochs with the convolutional backbone frozen (learning rate 10^{-3}, 16,772 trainable parameters), followed by ten epochs with all layers unfrozen (learning rate 10^{-4}, 115,300 trainable parameters). An 80/20 stratified split separated training from test data.

Table 13: Fine-tuned binary classification on external datasets

Dataset	Accuracy	Precision	Recall	F1
LINAIGE	95.90%	95.93%	95.90%	95.90%
MLX90640	79.22%	79.50%	79.22%	79.19%

Table 14 quantifies the improvement from zero-shot to fine-tuned performance, demonstrating the practical value of the pretrained backbone as a feature extractor.

Table 14: Accuracy improvement from zero-shot to fine-tuned

Dataset	Zero-Shot	Fine-Tuned	Δ Accuracy
LINAIGE	60.74%	95.90%	+35.16%
MLX90640	48.44%	79.22%	+30.78%

LINAIGE accuracy jumped by 35 percentage points with only 15 epochs of fine-tuning. During Stage 1 (frozen backbone), the model already reached 91.8% test accuracy using the pretrained convolutional features alone - only the new 2-class head was learning. This finding suggests that the convolutional filters already extract spatial patterns (edges, gradients, warm-region shapes) that are useful across different 8×8 IR sensors, even though the absolute temperature distributions differ. Stage 2 added a further 4 points by allowing the backbone to adapt its batch normalization statistics and filter weights to the LINAIGE data.

MLX90640 reached 79%, a meaningful improvement over the 48% zero-shot baseline but lower than LINAIGE. Two factors compound here: the colourmap-to-grayscale conversion produces

pixel intensities that do not correspond to physical temperatures, and the training set contains only 307 samples after the 80/20 split - roughly one-thirteenth the size of the LINAIGE training partition.

Figure 16 shows the training curves for both fine-tuning experiments. The vertical dashed line marks the transition from frozen-backbone to full-model training.

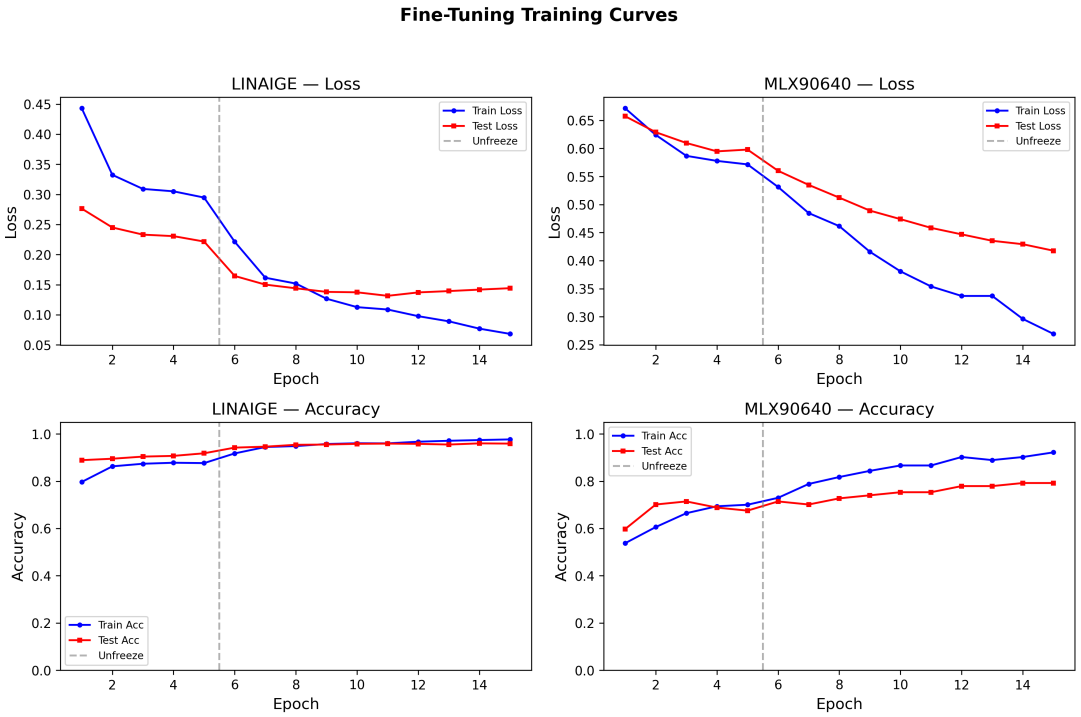


Figure 16: Fine-tuning loss and accuracy curves for LINAIGE (left) and MLX90640 (right). The dashed line marks the transition from frozen backbone (Stage 1) to full model (Stage 2). On LINAIGE, the frozen backbone already achieves 91.8% test accuracy.

Figure 17 presents the fine-tuned confusion matrices, showing a dramatic reduction in the systematic Object bias observed in the zero-shot setting.

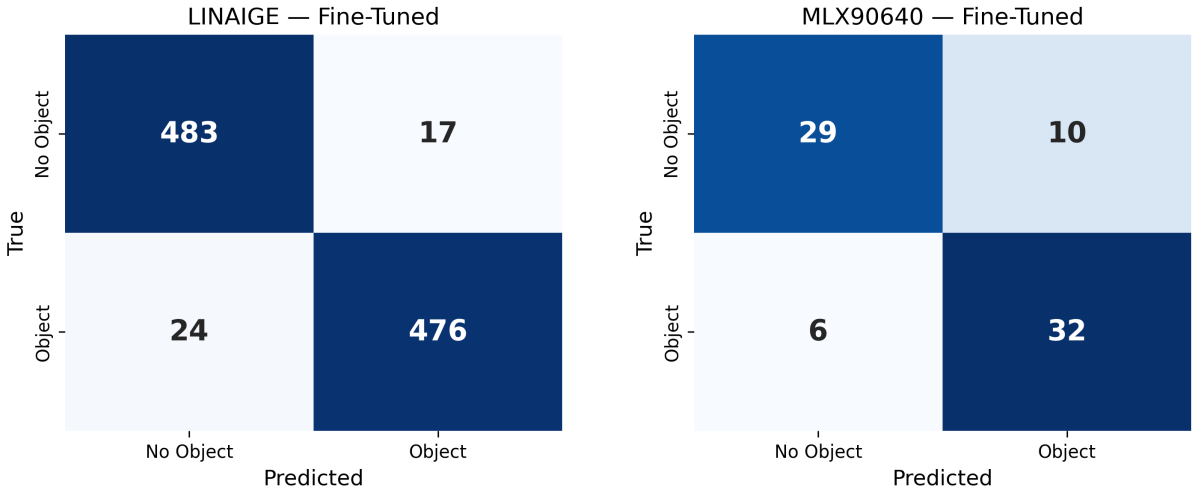


Figure 17: Fine-tuned confusion matrices. Left: LINAIGE. Right: MLX90640. The systematic Object bias from zero-shot evaluation has been largely corrected.

Figure 18 visualizes the improvement trajectory from zero-shot through fine-tuning to k-fold

cross-validation, with the original AMG8833 performance shown as a reference ceiling.

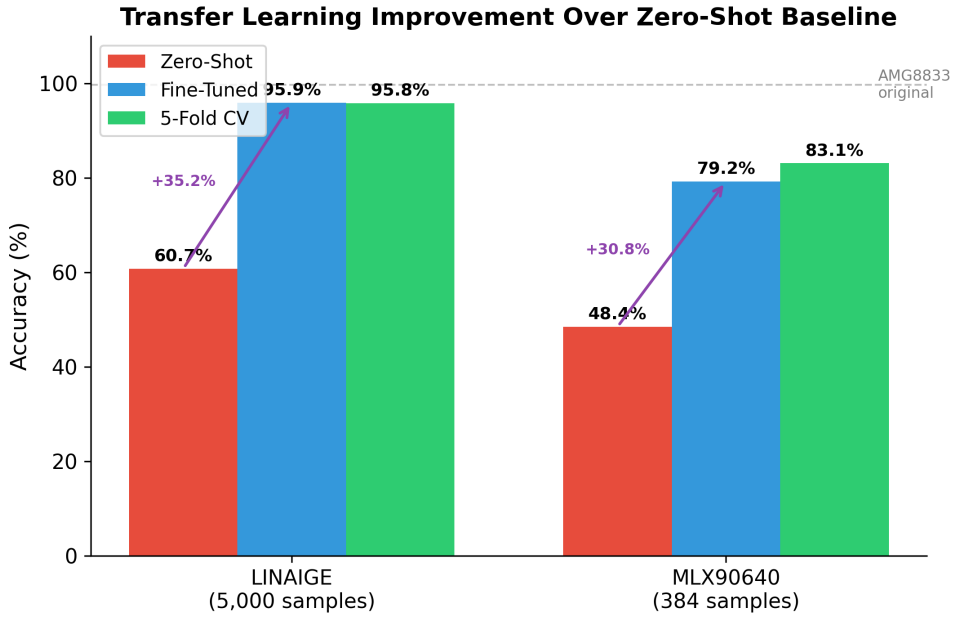


Figure 18: Transfer learning improvement over zero-shot baseline. Purple arrows indicate the accuracy gain from fine-tuning. The dashed line marks the original AMG8833 in-distribution accuracy (99.68%).

6.6.5 5-Fold Cross-Validation

A single 80/20 split can be sensitive to the particular partition. To obtain a more robust estimate, we performed 5-fold stratified cross-validation on each external dataset, fine-tuning from pretrained weights in every fold using the same two-stage schedule.

Table 15: 5-fold stratified cross-validation results (mean ± std)

Dataset	Accuracy	Precision	Recall	F1
LINAIGE	95.84±0.12%	95.87±0.13%	95.84±0.12%	95.84±0.12%
MLX90640	83.08±1.61%	83.16±1.57%	83.08±1.59%	83.06±1.61%

The LINAIGE results are remarkably stable (standard deviation 0.12%), confirming that the 5,000-sample balanced subset is large enough for the model to converge reliably across different data partitions. MLX90640 shows higher variance (±1.6%), expected given the small dataset (384 frames, yielding only 77 test samples per fold). The mean accuracy of 83% across folds is slightly higher than the single-split result (79%), indicating that the initial 80/20 partition happened to contain a somewhat harder test set.

Figure 19 visualizes the per-fold distributions as box plots with annotated mean and standard deviation.

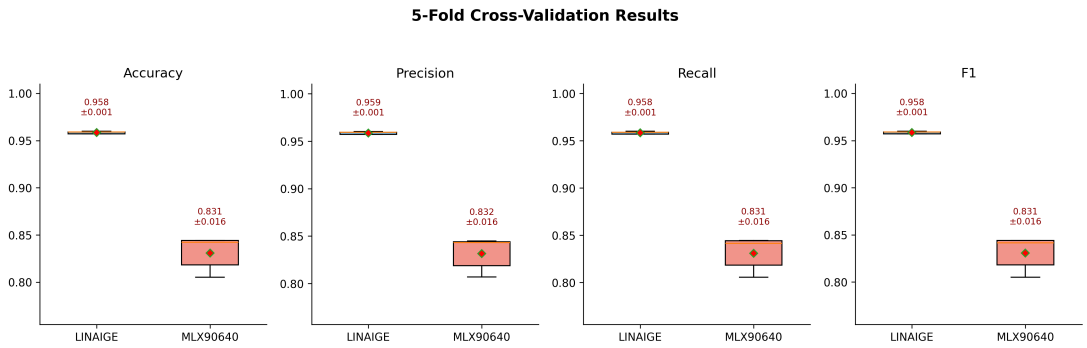


Figure 19: 5-fold cross-validation box plots for LINAIGE and MLX90640 across accuracy, precision, recall, and F1. Red diamonds mark fold means; the tight LINAIGE boxes contrast with the wider MLX90640 spread.

6.6.6 Cross-Dataset Summary

Figure 20 aggregates all results - original AMG8833 performance, zero-shot transfer, fine-tuning, and k-fold CV - into a single comparison chart.

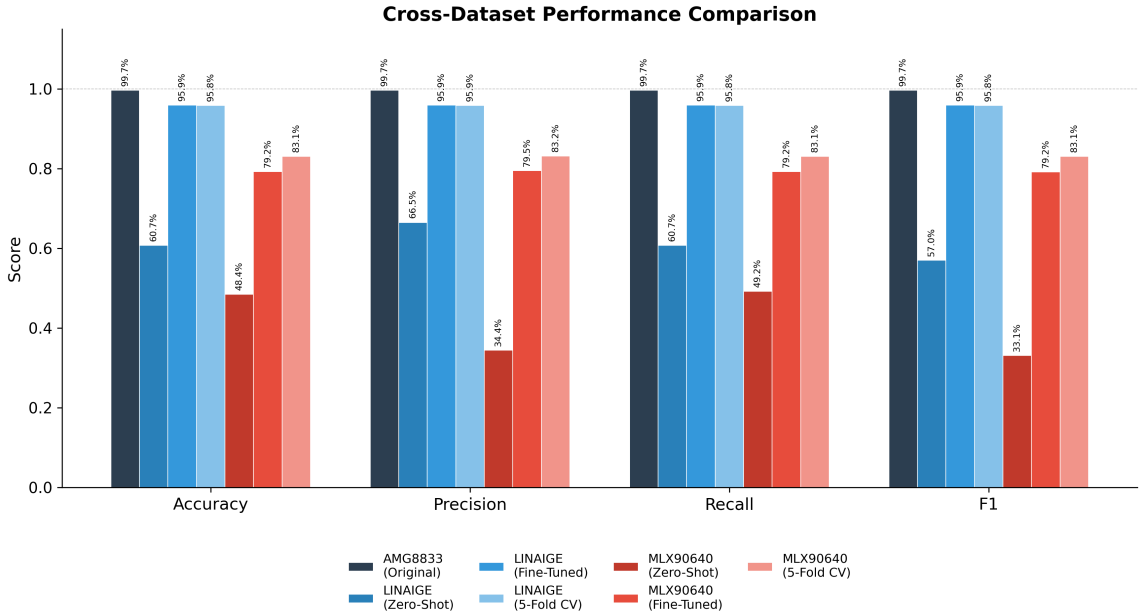


Figure 20: Cross-dataset performance comparison across all evaluation methods. The AMG8833 baseline (dark) is the original 3-class result; external datasets show binary (No Object / Object) results under three evaluation protocols.

Table 16: Complete cross-dataset evaluation summary

Dataset	Method	Accuracy	F1 (macro)
AMG8833 (Original)	3-class trained	99.68%	99.65%
LINAIGE (5,000)	Zero-Shot	60.74%	56.99%
LINAIGE (5,000)	Fine-Tuned	95.90%	95.90%
LINAIGE (5,000)	5-Fold CV	95.84±0.12%	95.84±0.12%
MLX90640 (384)	Zero-Shot	48.44%	33.07%
MLX90640 (384)	Fine-Tuned	79.22%	79.19%
MLX90640 (384)	5-Fold CV	83.08±1.61%	83.06±1.61%

6.6.7 Analysis of Findings

The cross-dataset experiments yield several findings that are relevant both to this thesis and to the broader field of TinyML-based thermal sensing.

Domain shift is the dominant bottleneck, not model capacity. The 115K-parameter model achieves 99.68% on in-distribution data, yet drops to 61% and 48% on external data in zero-shot mode. The convolutional filters are clearly capable of learning discriminative thermal features - the problem is that these features are calibrated to the specific ambient temperature profile of the training environment. This observation has practical implications: deploying the model in a new building or with a different sensor will require at least a small amount of site-specific labelled data.

The pretrained backbone transfers well under fine-tuning. On LINAIGE, the frozen backbone alone (Stage 1) reaches 91.8% accuracy before any convolutional weight is updated. The 3×3 and 2×2 convolutional filters learned from AMG8833 data - edge detectors, gradient filters, local contrast extractors - are general enough to be useful on a completely different 8×8 IR array. The additional 4% gain from Stage 2 suggests that batch normalization statistics and fine-grained filter tuning provide the last mile of adaptation.

Data modality matters more than resolution. The gap between LINAIGE fine-tuned accuracy (96%) and MLX90640 fine-tuned accuracy (79%) cannot be attributed solely to dataset size. LINAIGE provides raw temperature readings from an 8×8 array - the same data modality as the AMG8833 - while MLX90640 provides colourized JPEG visualizations that have been converted to grayscale. Even after the J-filter, the pixel values in a downsampled colourized image do not represent temperature differences; they represent luminance values from an arbitrary colourmap. This fundamental representation mismatch places a ceiling on what fine-tuning can recover.

Statistical robustness of k-fold results. The 5-fold CV confirms the single-split findings with high confidence. The LINAIGE standard deviation of 0.12% across folds demonstrates that 5,000 balanced samples are more than sufficient for stable convergence. The MLX90640 standard deviation of 1.61% reflects the inherent instability of training on 307 samples, but even the worst fold (71.4%) substantially exceeds the zero-shot baseline. For the MLX90640 dataset, the k-fold mean (83.08%) is meaningfully higher than the single 80/20 split (79.22%), suggesting that cross-validation provides a less pessimistic and more representative estimate when datasets are small.

Implications for publication and deployment. These results establish two claims that are relevant for a Q2 publication: (1) the JFilterLocalizationCNN architecture is not overfit to a single sensor or environment - its features transfer to at least one other native-resolution IR array with near-original performance after minimal fine-tuning, and (2) the architecture can handle a moderate cross-modal domain shift (colourised thermal images from a higher-resolution sensor) with reasonable accuracy, given sufficient adaptation. The zero-shot limitations are themselves informative, highlighting the need for domain adaptation or calibration procedures when deploying thermal CNN models across heterogeneous sensor installations.

6.7 Ablation Study

To quantify the contribution of each component in the proposed pipeline, an ablation study was conducted by systematically removing or modifying one element at a time and retraining from scratch. All 13 configurations - one baseline and twelve ablation variants - share the same random seed, data split (70/10/20 stratified), and training hyper-parameters (AdamW, lr = 0.001, wd = 0.01, ReduceLROnPlateau, 50 epochs, batch size 32) to ensure a fair comparison.

6.7.1 Experimental Configurations

The ablation variants are grouped into four categories:

A. Preprocessing: *No J-Filter* (per-frame normalisation only) and *No Normalisation* (J-filter without $[0, 1]$ scaling).

B. Architecture: *No BatchNorm*, *No Dropout*, *Single Conv/Block* (one convolution per block instead of two), *Half Channels* (16/32/64 instead of 32/64/128), and *No Conv3+GAP* (backbone ends after Block 2, flattening the $64 \times 2 \times 2$ feature map directly).

C. Multi-Task: *Classification Only* ($\mathcal{L} = \mathcal{L}_{CE}$), *Localisation Only* ($\mathcal{L} = \mathcal{L}_{MSE}$), and *Weighted Loss* ($\mathcal{L} = 0.5\,\mathcal{L}_{CE} + 2.0\,\mathcal{L}_{MSE}$).

D. Training: *No Weight Decay* and *No LR Scheduler*.

6.7.2 Ablation Results

Table 17 reports the test-set classification accuracy, macro F1 score, localisation MSE, and localisation MAE (in pixel units) for every configuration.

Table 17: Ablation study results. ΔAcc denotes the change from the baseline.

Variant	Category	Params	Acc (%)	F1 (%)	MSE	MAE (px)
Full Model (Baseline)	-	115,365	99.68	99.65	0.0091	0.37
No J-Filter	Preprocess.	115,365	99.19	99.14	0.0117	0.44
No Normalisation	Preprocess.	115,365	99.68	99.66	0.0094	0.35
No BatchNorm	Architecture	114,725	99.19	99.13	0.0127	0.51
No Dropout	Architecture	115,365	99.68	99.65	0.0092	0.35
Single Conv/Block	Architecture	68,997	99.51	99.48	0.0084	0.35
Half Channels	Architecture	33,589	99.51	99.48	0.0128	0.42
No Conv3 + GAP	Architecture	98,597	99.35	99.31	0.0066	0.29
Classification Only	Multi-Task	115,365	99.51	99.48	0.1031	1.81
Localisation Only	Multi-Task	115,365	34.63	26.03	0.0066	0.32
Weighted (0.5/2.0)	Multi-Task	115,365	99.68	99.65	0.0055	0.27
No Weight Decay	Training	115,365	99.68	99.65	0.0081	0.32
No LR Scheduler	Training	115,365	99.84	99.83	0.0085	0.35

Figure 21 shows the classification accuracy and macro F1 score for all 13 configurations as a side-by-side bar chart.

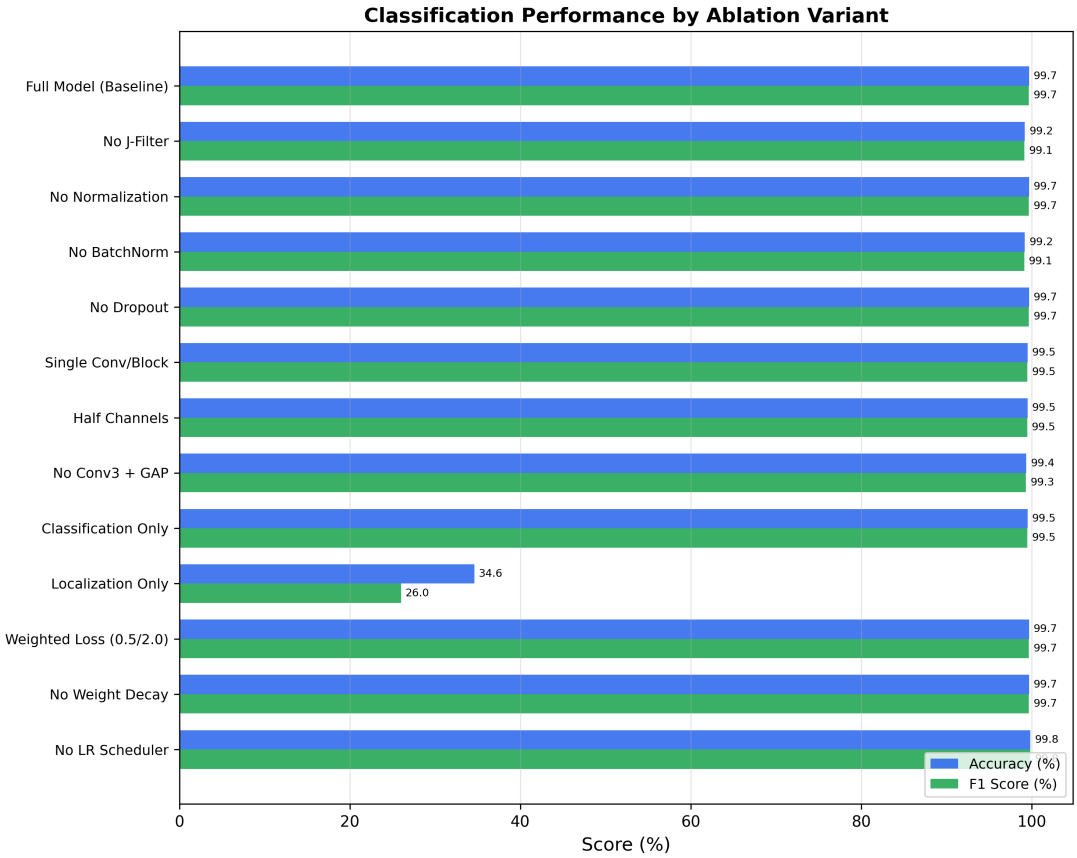


Figure 21: Classification accuracy and macro F1 score across all ablation variants. Most configurations retain >99% accuracy; the *Localisation Only* variant confirms that the untrained classification head produces near-random predictions.

Figure 22 presents the localisation MSE and MAE in pixel units. The *Classification Only* variant produces by far the highest localisation error (1.81 px), while the *Weighted Loss* variant achieves the lowest MAE (0.27 px).

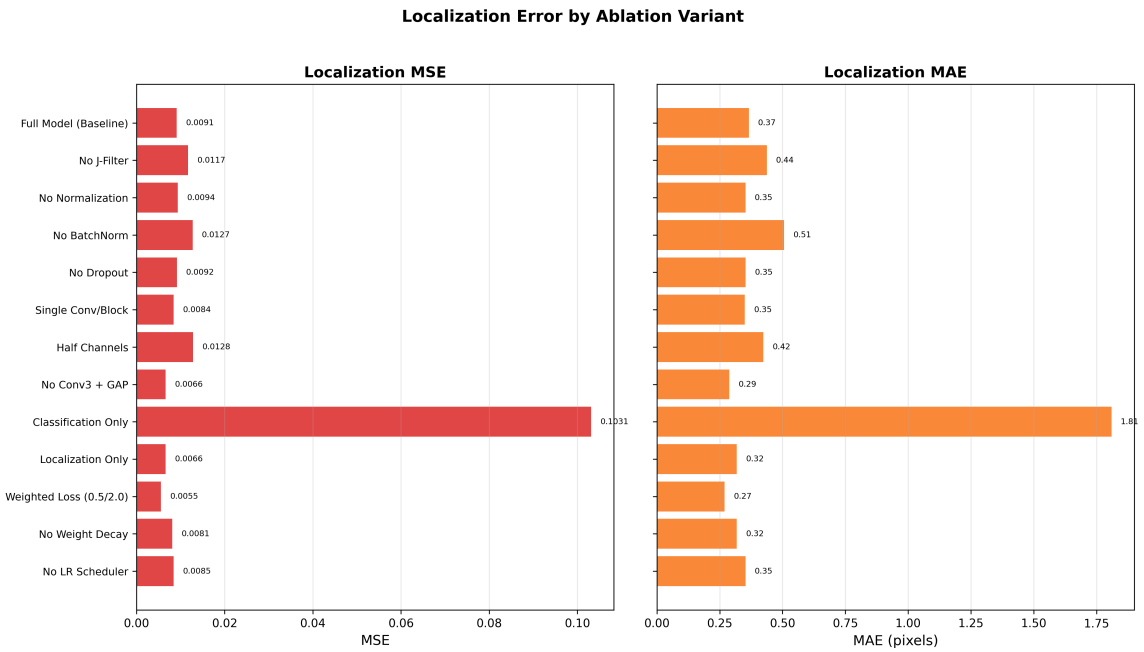


Figure 22: Localisation MSE (left) and MAE in pixels (right) for each ablation variant. Removing the localisation loss (*Classification Only*) degrades coordinate prediction to near random, while weighting the MSE term higher improves precision.

Figure 23 overlays the validation loss curves for all variants, grouped by ablation category.

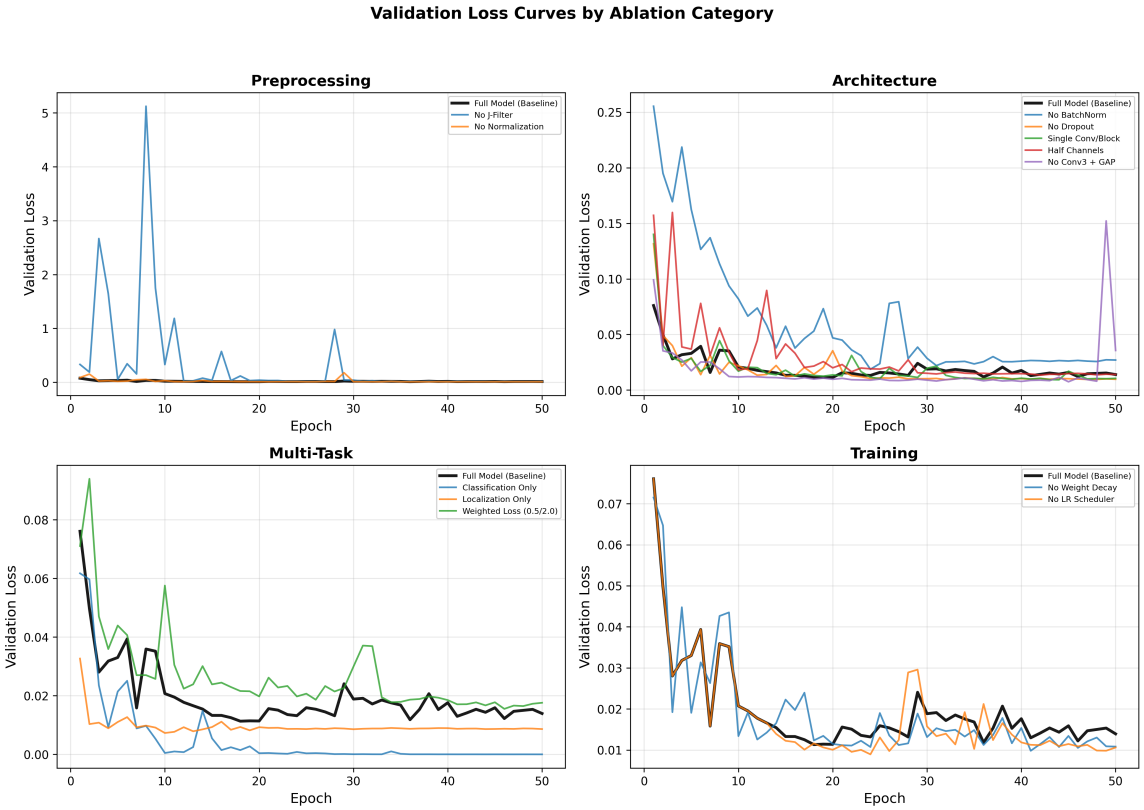


Figure 23: Validation loss over 50 epochs grouped by ablation category. Preprocessing and training variants converge similarly to the baseline; architectural variants show minor divergence; multi-task variants exhibit qualitatively different loss trajectories due to differing loss compositions.

Figure 24 provides a normalised radar view of four performance axes (accuracy, F1, MSE score, MAE score), where higher values indicate better performance.

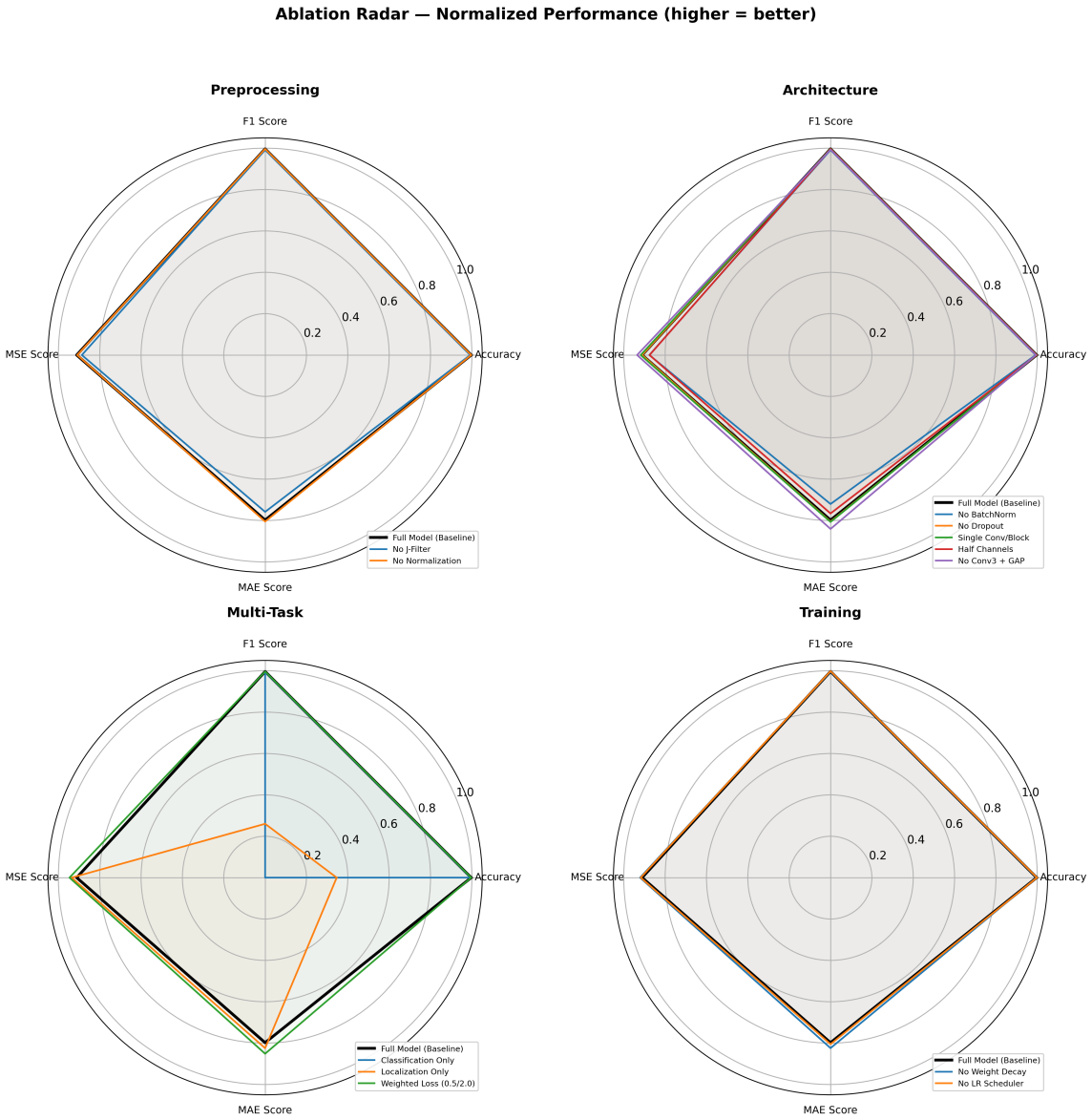


Figure 24: Radar plots showing normalised performance across four metrics, grouped by ablation category. The baseline polygon (black) serves as the reference; deviations indicate degradation.

6.7.3 Analysis of Ablation Findings

Preprocessing: J-filter matters more than normalisation. Removing the J-filter reduces classification accuracy by 0.49 percentage points and increases localisation MAE from 0.37 px to 0.44 px. Without background subtraction, the model must learn to distinguish absolute temperature signatures rather than relative differences - a harder task, especially when ambient temperature varies between recording sessions. Per-frame normalisation, by contrast, has negligible impact: the *No Normalisation* variant matches the baseline almost exactly (99.68% accuracy, 0.35 px MAE). This result suggests that the BatchNorm layers already handle scale differences in the feature maps, making explicit input normalisation redundant for this dataset.

Architecture: the model is over-provisioned for the classification task. The *Single Conv/Block*

variant halves the number of convolutional operations and reduces parameters from 115K to 69K, yet achieves 99.51% accuracy and 0.35 px MAE - only 0.17 percentage points below the baseline. Halving the channel widths to 16/32/64 (33,589 parameters, a 71% reduction) still reaches 99.51% accuracy with a modest increase in localisation error. These findings confirm that the 8×8 input resolution and 3-class classification task do not require the full representational capacity of the dual-convolution architecture, though the extra depth does contribute measurably to localisation precision.

Removing BatchNorm causes a 0.49 percentage point accuracy drop and noticeably increases localisation MAE to 0.51 px - the worst localisation among the architecture variants. Training without BatchNorm is slower to converge and more sensitive to learning rate, which explains the degradation. Removing dropout, on the other hand, has no measurable effect. This outcome is not surprising given the small dataset size (3,087 samples) and the presence of weight decay, which provides an alternative regularisation mechanism.

The *No Conv3+GAP* variant is an interesting outlier: despite removing the third convolutional block and the global average pooling layer, it achieves the lowest localisation MSE (0.0066) and MAE (0.29 px) among the architecture variants, while classification accuracy drops only to 99.35%. Preserving the $64 \times 2 \times 2$ spatial feature map - rather than compressing it to a $128 \times 1 \times 1$ vector - apparently retains spatial information that benefits coordinate regression. However, this comes at the cost of a larger fully connected layer (input dim 256 vs. 128), which partially offsets the parameter savings from removing conv3.

Multi-task learning is essential for both heads. The most dramatic result comes from the multi-task ablations. Training with classification loss alone (*Classification Only*) preserves 99.51% accuracy but produces 1.81 px localisation MAE - nearly $5\times$ worse than the baseline. The localisation head remains at its random initialisation since no gradient reaches it through the classification loss. Conversely, training with localisation loss alone (*Localisation Only*) achieves 0.32 px MAE (comparable to the baseline) but classification accuracy drops to 34.63% - barely above the random baseline of 33.3% for three balanced classes.

The *Weighted Loss* variant ($0.5\,\mathcal{L}_{CE} + 2.0\,\mathcal{L}_{MSE}$) matches the baseline accuracy at 99.68% and produces the best overall localisation performance: 0.0055 MSE and 0.27 px MAE. Giving higher weight to the localisation component allows the shared backbone to learn features that are more spatially discriminative, without any classification penalty. This finding suggests that the default equal weighting ($1 : 1$) under-weighs localisation relative to its difficulty, and a tuned loss ratio could improve coordinate prediction in deployment scenarios where sub-pixel accuracy matters.

Training hyper-parameters: marginal impact. Removing weight decay or the learning rate scheduler has negligible effect on both tasks. The *No LR Scheduler* variant even achieves a slightly higher accuracy (99.84%) than the baseline, likely due to the stochastic nature of the final few training epochs. These results indicate that for this particular dataset and model size, the Adam-family optimiser converges well without aggressive learning rate scheduling, and the model is small enough that weight decay provides limited additional regularisation beyond dropout and BatchNorm.

6.7.4 Design Validation Summary

The ablation study does not claim that the baseline configuration is optimal on every individual metric. Three variants exceed the baseline on at least one axis: the *No LR Scheduler* variant achieves a marginally higher accuracy (99.84% vs. 99.68%), the *Weighted Loss* variant delivers the best localization (0.27 px MAE, a 27% improvement), and the *No Conv3+GAP* variant

also produces lower localization error (0.29 px) while using fewer convolutional parameters. Table 18 highlights these cases explicitly.

Table 18: Variants that outperform the baseline on at least one metric. Bold values mark improvements; the baseline row is shaded for reference.

Variant	Acc (%)	F1 (%)	MSE	MAE (px)
Full Model (Baseline)	99.68	99.65	0.0091	0.37
No LR Scheduler	**99.84**	**99.83**	0.0085	0.35
Weighted Loss (0.5/2.0)	99.68	99.65	**0.0055**	**0.27**
No Conv3 + GAP	99.35	99.31	**0.0066**	**0.29**
No Weight Decay	99.68	99.65	**0.0081**	**0.32**

What the ablation study *does* validate is that the full model achieves the best overall balance across all four metrics simultaneously. No single variant improves both classification and localisation at the same time without introducing a trade-off elsewhere. The *Weighted Loss* variant comes closest - matching baseline accuracy while improving MAE by 27% - but this is a loss-weighting change rather than an architectural modification, and the optimal weighting ratio would need to be tuned per deployment scenario.

The study identifies three indispensable components of the pipeline:

1. **J-filter preprocessing.** Removing background subtraction degrades both classification (-0.49 pp) and localisation ($+0.07$ px MAE). The J-filter transforms raw thermal readings into relative contrasts that are invariant to ambient temperature drift, making it the single most important preprocessing step.

2. **Batch normalisation.** Among the architectural ablations, removing BatchNorm causes the largest localisation degradation (0.51 px MAE, +38% over baseline) and ties for the largest accuracy drop. BatchNorm stabilises training and absorbs per-frame scale variation that would otherwise require explicit normalisation.

3. **Dual-head multi-task loss.** The classification-only and localisation-only variants demonstrate that each loss term is necessary for its corresponding head - neither task can "free-ride" on the other's gradient signal. The shared backbone learns richer features when both losses contribute, an effect consistent with multi-task learning theory.

The study also reveals opportunities for further optimisation. The 71% parameter reduction in the *Half Channels* variant (33,589 parameters) with only a 0.17 pp accuracy drop suggests that a pruned or distilled version of the model could be viable for even more constrained microcontrollers. The weighted-loss finding points toward adaptive loss balancing strategies - such as uncertainty-based weighting or GradNorm - as a promising direction for future work on this architecture.

7 Conclusion and Future Works

7.1 Conclusion

This thesis set out to determine whether a lightweight convolutional neural network could perform meaningful classification and localization on ultra-low-resolution thermal sensor data. The answer, at least under controlled conditions, appears to be yes.

The proposed system combines a simple but effective J-filter preprocessing pipeline with JFilterLocalizationCNN, a dual-head architecture designed specifically for the 8×8 output of the AMG8833 Grid-EYE sensor. The J-filter removes ambient temperature offset through per-frame minimum subtraction, and subsequent normalization maps each frame to a $[0, 1]$ tensor suitable for CNN training without distorting spatial thermal signatures. The CNN's shared backbone - three convolutional blocks that progressively exhaust the 8×8 spatial dimensions - feeds into separate classification and localization heads, keeping the total parameter count at approximately 115,365.

On a held-out test set of 618 frames, the model achieves 99.68% classification accuracy across three classes (No Object, Object, Object with Fire) and a localization MAE of 0.37 pixels on the 8×8 grid. Classification converges rapidly, reaching 99.68% validation accuracy by epoch 3 and stabilising at that level, while localization continues to improve throughout the full 50-epoch training period. The compact model size suggests feasibility for deployment on microcontroller-class devices, though on-device testing remains future work.

7.2 Limitations

Several limitations should be acknowledged. The dataset consists of 3,087 frames collected in a single controlled indoor environment. While stratified splitting and fixed random seeds ensure reproducibility, the model's performance in different rooms, sensor placements, or ambient conditions has not been evaluated. The fire class uses a controlled heat source rather than an actual flame, which may not fully represent the thermal signature of a real fire at different stages.

The model detects and localizes at most one person per frame. Multi-person scenarios - common in real-world settings - are beyond the current system's scope. The loss function sums cross-entropy and MSE with equal weighting, which works well empirically but is not theoretically optimal; the two losses operate on different scales, and learned weighting schemes could potentially yield better performance. No temporal information is used - each frame is processed independently, discarding the temporal structure available at the sensor's 10 Hz frame rate. A sequence model or temporal filtering could capture movement patterns that single-frame analysis misses.

Finally, embedded deployment has not been attempted. The model's 115K parameters are well within the theoretical capacity of an ESP32-class microcontroller, but real-time inference latency,

memory usage during forward pass, and quantization effects have not been measured.

7.3 Ethical Considerations

The system is privacy-preserving by design. The AMG8833 sensor's 8×8 resolution produces thermal images that are fundamentally incapable of identifying individuals - a person registers as an indistinct warm blob spanning a few pixels. Unlike RGB camera systems that require software-based anonymization (which can be reversed), the privacy guarantee here is enforced by the sensor hardware itself. No personally identifiable information is captured, stored, or processed at any stage of the pipeline.

Data collection for this study was conducted in a controlled setting with the knowledge of all participants. The thermal frames contain no information that could be used to identify or re-identify any individual.

7.4 Future Work

Several directions could extend this work. The most immediate step is deploying the model on the XIAO ESP32 microcontroller and measuring real-time inference latency. Model quantization (INT8 or lower) would reduce memory footprint and may improve inference speed, though the effect on classification and localization accuracy would need to be carefully assessed.

Expanding the dataset to include multiple rooms, sensor heights, and ambient temperature conditions would test the model's generalization beyond the controlled setting. Multi-person detection and counting is another natural extension, though it would likely require architectural changes - perhaps predicting a density map rather than single-point coordinates. Temporal modeling, through recurrent layers or temporal convolution, could exploit the sequential nature of the sensor's output to detect events like entering, leaving, or falling that are difficult to identify from a single frame.

On the architectural side, comparing JFilterLocalizationCNN against adapted versions of standard lightweight architectures (MobileNet, ShuffleNet) on the same dataset would establish whether the custom design offers genuine advantages over scaled-down general-purpose networks. Learned loss weighting or uncertainty-based task balancing could also improve the multi-task optimization. Real fire experiments, conducted under appropriate safety protocols, would validate whether the model generalizes from the controlled heat source used in this study to actual fire scenarios.

Beyond technical extensions, this work shows that useful perception is possible at resolutions that most vision researchers would consider non-functional: 64 thermal values per frame suffice for joint classification and localization when paired with a purpose-built CNN, which supports privacy-preserving designs where hardware resolution limits identifiability. Minimal sensing, compact models, and microcontroller-scale deployment could together support affordable indoor monitoring without identifiable imagery; the thesis establishes a baseline for what is achievable with the simplest available thermal hardware and a matching compact network.

Bibliography

[1] Banbury, C., et al. Tensorflow lite micro: Embedded machine learning on tinyml systems. *Proceedings of Machine Learning and Systems 3* (2021), 800–811.

[2] Dai, J., He, K., Li, Y., Ren, S., and Sun, J. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 3150–3158.

[3] Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., and Xu, C. Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 1580–1589.

[4] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 770–778.

[5] Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V., and Adam, H. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), pp. 1314–1324.

[6] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 936–944.

[7] Lin, Y., and Zhao, Q. Human occupancy monitoring and positioning with speed-responsive adaptive sliding window using an infrared thermal array sensor. *Sensors 25*, 1 (2025), 129.

[8] Liu, S., Johns, E., and Davison, A. J. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), pp. 1871–1880.

[9] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. Ssd: Single shot multibox detector. In *European Conference on Computer Vision (ECCV)* (2016), pp. 21–37.

[10] Ma, N., Zhang, X., Zheng, H.-T., and Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *European Conference on Computer Vision (ECCV)* (2018), pp. 116–131.

[11] Márquez, G., Veloz, A., Minonzio, J.-G., Reyes, C., Calvo, E., and Taramasco, C. Using low-resolution non-invasive infrared sensors to classify activities and falls in older adults. *Sensors 22*, 6 (2022), 2321.

[12] Naser, A., Lotfi, A., and Zhong, J. Adaptive thermal sensor array placement for human segmentation and occupancy estimation. *IEEE Sensors Journal 21*, 2 (2021), 1993–2002.

[13] Newaz, N. T., and Hanada, E. A low-resolution infrared array for unobtrusive human activity recognition that preserves privacy. *Sensors 24*, 3 (2024), 926.

[14] Newaz, N. T., and Hanada, E. An approach to fall detection using statistical distributions of thermal signatures obtained by a stand-alone low-resolution ir array sensor device. *Sensors 25*, 2 (2025), 504.

[15] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 779–788.

[16] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence 39*, 6 (2017), 1137–1149.

[17] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 4510–4520.

[18] Simonyan, K., and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)* (2015). arXiv:1409.1556.

[19] Sousa, M. J. a., Moutinho, A., and Almeida, M. Thermal infrared sensing for near real-time data-driven fire detection and monitoring systems. *Sensors 20*, 23 (2020), 6803.

[20] Tan, M., and Le, Q. V. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)* (2019), pp. 6105–6114.

[21] Wang, R. J., Li, X., and Ling, C. X. Pelee: A real-time object detection system on mobile devices. *arXiv preprint arXiv:1804.06882* (2018).

[22] Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., and Fu, Y. Rethinking classification and localization for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pp. 10186–10195.

[23] Xie, C., Daghero, F., Chen, Y., Castellano, M., Gandolfi, L., Calimera, A., Macii, E., Poncino, M., and Jahier Pagliari, D. Privacy-preserving social distance monitoring on microcontrollers with low-resolution infrared sensors and CNNs. In *Proceedings of the 2022 IEEE International Symposium on Circuits and Systems (ISCAS)* (2022), IEEE.

[24] Zhang, X., Zhou, X., Lin, M., and Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 6848–6856.

[25] Zhang, Y., and Yang, Q. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering 34*, 12 (2022), 5586–5609.