machine learning & knowledge extraction

MDPI

Article

A Tiny Vision-Based Model for Real-Time Student Attention Detection in Online Classes

Chaymae Yahyati [1], Ismail Lamaakal [1], Yassine Maleh [2,*], Khalid El Makkaoui [1] and Ibrahim Ouahbi [1]

[1] Multidisciplinary Faculty of Nador, Mohammed Premier University, Oujda 60000, Morocco; chaymae.yahyati@ump.ac.ma (C.Y.); ismail.lamaakal@ieee.org (I.L.); k.elmakkaoui@ump.ac.ma (K.E.M.); i.ouahbi@ump.ac.ma (I.O.)
[2] Laboratory LaSTI, ENSAK, Sultan Moulay Slimane University, Khouribga 23000, Morocco
* Correspondence: yassine.maleh@ieee.org

Abstract

Online and blended classrooms widen access but remove the in-person cues instructors use to gauge attention. Prior work typically relies on heavy, cloud-bound or multimodal models that are hard to deploy on commodity laptops, treats attention as an unordered label without calibrated probabilities, and evaluates on subject-overlapping splits with limited robustness analysis. This creates a gap in Tiny, deployable, calibration-aware methods validated under realistic protocols. We address this gap with a TinyML, vision-only pipeline that estimates four attention levels: (Very Low, low, high, Very High) from short webcam clips under strict on-device budgets. Each clip of $T = 30$ frames at 224×224 is processed by a compact hybrid encoder: a CNN extracts per frame spatial features, a BiLSTM models temporal context, and a lightweight GRU refines dynamics; three parallel branches with staggered widths encourage feature diversity before fusion. We apply structured pruning of convolutional channels and recurrent units, post-training INT8 quantization, and temperature scaling for calibrated probabilities; models are exported as ONNX. On DAiSEE with subject-independent splits, the baseline attains 99.86% accuracy and 0.998 macro-F1, with strong ordinal agreement (QWK = 0.998, ordinal MAE = 0.03). The compressed model preserves reliability (macro-F1 = 0.995, QWK = 0.995), remains robust to low light, partial occlusion, and head yaw, and yields $\sim 4\times$ smaller size and $\sim 2.3\times$ CPU speedups. These results indicate a deployable, privacy-preserving approach to fine-grained, on-device attention analytics.

Keywords: student attention detection; engagement detection; hybrid CNN–BiLSTM–GRU; model compression; on-device inference; post-training INT8 quantization; structured pruning; TinyML

Check for updates

Academic Editor: Francesco Buccafurri

Received: 25 March 2026
Revised: 20 April 2026
Accepted: 24 April 2026
Published: 28 April 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Online and blended classrooms have expanded access to education [1,2], but they have also attenuated the rich, real-time cues that instructors traditionally use to judge whether learners are following along [3–5]. In virtual settings, engagement must be inferred from a single camera stream under heterogeneous illumination, camera placement, and device quality, while interaction logs and end-of-lesson quizzes arrive too late to guide timely intervention [6]. Estimating a student's *attention level* [7] directly from video is therefore an appealing capability for adaptive pacing [8], formative feedback, and cohort-level analytics [9]. Unlike generic emotion recognition [10], however, attention manifests as

subtle, temporally extended patterns [11] and is labeled on an *ordinal* scale (*Very Low*, *low*, *high*, *Very High*), where near-miss errors are far less harmful than cross-scale mistakes. At the same time, practical deployment in classrooms is constrained by privacy (favoring on-device inference) [12] and modest compute/memory budgets on CPU-only laptops or edge PCs, making latency and footprint as important as raw accuracy [13–15].

Recent advances in deep learning highlight complementary strengths and trade-offs. CNNs and recurrent models (LSTM/BiLSTM/GRU) have improved recognition of temporally nuanced behaviors by learning hierarchical spatiotemporal features from raw video [12,16–22], while transformer-based variants extend temporal receptive fields and global context [23–25]. Yet the most accurate architectures are often heavy and data-hungry, with inference costs that are difficult to justify on classroom hardware; many real-time systems, in turn, prioritize throughput but depend on proxy tasks (e.g., facial expression recognition) or private datasets, and rarely report ordinal or calibration metrics [26–29]. Multimodal approaches combining posture, gaze, blink rate, or even Electroencephalogram (EEG) can further boost accuracy [30,31], but they are often impractical at scale due to cost, occlusions, lighting constraints, or the need for specialized sensors. What is missing is a *deployable*, vision-only pipeline that (1) captures fine-grained temporal dynamics with a small parameter budget, (2) respects the *ordered* nature of attention labels, and (3) preserves probabilistic reliability after compression for on-device inference [32–34].

To address these gaps, we present a lightweight, TinyML-aligned video pipeline for attention-level recognition that couples compact spatial encoding with bidirectional temporal aggregation and gated refinement, and that maintains accuracy, calibration, and ordinal consistency under deployment-oriented compression. Concretely, we process DAiSEE videos under a subject-independent protocol, represent each clip as $T = 30$ RGB frames at 224×224, and apply *adaptive luminance normalization* to stabilize appearance without overexposing already good frames. Our hybrid **CNN–BiLSTM–GRU** encoder extracts frame-wise features with a compact CNN, aggregates temporal context bidirectionally to disambiguate *low* from *high*, and refines predictions with a lightweight GRU that smooths short, noisy segments; a three-branch design with staggered widths/dropouts promotes feature diversity before recurrent fusion and a softmax head. To meet edge constraints, we employ *structured channel/unit pruning* and *post-training* `INT8` *quantization*, optionally guided by knowledge distillation, and restore probabilistic fidelity via a single temperature parameter. The resulting ONNX artifacts execute efficiently on CPUs while preserving the *ordinal geometry* of decisions, as reflected by the high Quadratic Weighted Kappa (QWK) and low ordinal MAE alongside accuracy.

This paper makes the following contributions:

1. **A compact spatiotemporal architecture for attention level:** A three-branch CNN–BiLSTM–GRU that captures fine-grained temporal dynamics with a small parameter budget, producing *ordinally local* errors instead of catastrophic cross-scale mistakes.
2. **A deployment-oriented compression recipe:** Structured pruning of CNN channels and recurrent units plus post-training `INT8` quantization (with optional distillation) and a one-parameter temperature scaling step that preserves the accuracy, calibration, and QWK while delivering $\sim 4\times$ smaller models and $2.3\times$ CPU speedups.
3. **An ordinal and reliability-aware evaluation on DAiSEE:** Beyond accuracy and macro-F1, we report the QWK, ordinal MAE, Brier score, ECE, confusion locality, sequence-length sensitivity, and robustness to low light, occlusion, and head yaw, under a subject-independent split aligned with classroom generalization.
4. **Practical on-device deployment artifacts:** ONNX exports with subject-independent performance near ceiling (baseline: 99.86% accuracy, macro-F1 0.998; compressed:

99.52% accuracy, macro-F1 0.995) and calibrated probabilities suitable for CPU-only laptops and lab PCs.

The remainder of this paper is organized as follows: Section 2 surveys related work on attention and engagement analysis across vision and multimodal streams. Section 3 details our proposed hybrid CNN–BiLSTM–GRU architecture, the adaptive preprocessing, and the compression pipeline (structured pruning, `INT8` quantization, and knowledge distillation). Section 4 describes the dataset, subject-independent splits, training protocol, and evaluation metrics, and then reports results with ablations, robustness studies, and a head-to-head comparison with representative state-of-the-art systems. Finally, Section 5 concludes with limitations and directions for future work.

2. Related Works

Recent advances in student attention and engagement detection have leveraged diverse architectures and multimodal cues, ranging from graph-based spatiotemporal modeling to lightweight vision systems.

Beginning with graph-based approaches, Mandia et al. [30] proposed a video-based engagement detector combining an attention-enhanced GCN with a BiLSTM for temporal analysis. Their system classified six affective states (Boredom, confusion, Engaged, Frustration, Sleepy, yawning) using datasets including DAiSEE, YawDD, BAUM-1, and RLDD, with validation on both controlled and classroom recordings. The results showed strong performance with 65.35% (curated), 99.20% (YawDD), and 56.17% (DAiSEE) accuracy, while engagement estimates significantly correlated with post-lesson scores (r = 0.64). Subsequently, Zhang et al. [35] introduced FMAE, a self-supervised masked autoencoder to recognize online student engagement from facial videos. It was pre-trained on unlabeled data using region-prioritized masking (eyes, mouth) and a reconstruction + adversarial loss. The model achieved state-of-the-art performance with 64.74% accuracy on the DAiSEE dataset and demonstrates competitive results on EmotiW, highlighting its ability to learn robust spatiotemporal features without extensive labeled data. Building on these works, Pabba et al. [26] presented a real-time, vision-based engagement monitor that fused facial expressions, head pose, and head movement. Using a modified MobileNetV2 fine-tuned on their CSFED+ dataset (12,870 images, seven academic states, 70 students), their FER model reached 95.7%/94.9%/76% (train/val/test), and engagement estimates correlated 75% (individual) and 80% (class-level) with student self-reports. In parallel, Xiong et al. [23] fused a ResNet facial-expression branch with a ViT body-posture branch to classify classroom engagement. On a real-world dataset with four levels (e.g., sleeping, phone use, drinking, active learning), the model hit 92.9% accuracy, outperforming ResNet (85.5%) and ViT (86.1%), with robustness shown in confusion and training curves. Concurrently, Maddu et al. [36] developed a hybrid FER-to-engagement pipeline: face detection with Viola–Jones; feature extraction using Improved AAM, SLBT, GBP, and ResNet; emotion recognition with a hybrid CNN + Improved DBN; and engagement prediction via an improved entropy-based method. On CK+ and FER-2013 (seven emotions) datasets, the model achieved 95% accuracy on CK+ with an 80% training split and outperformed CNN, DBN, and LSTM in sensitivity and precision. Similarly focused on real-time performance, Mohammed Aly [37] proposed an online learning platform that tracks student engagement and emotions in real time using FER. The model combined ResNet50 (feature extraction), CBAM (attention), and TCNs (temporal dynamics). Experiments conducted on RAF-DB, FER2013, CK+, and KDEF datasets demonstrated strong performance, achieving accuracies of 91.86%, 91.71%, 95.85%, and 97.08%, respectively.

Unlike previous approaches, Alruwais et al. [38] presented a real-time online classroom monitor using a CNN with dropout and batch normalization to recognize students

and detect engagement/expressions (anger, happiness, surprise). Training in the UPNA Head Pose Database (11,342 grayscale images, 20 subjects), it achieved 99% accuracy for identification and activity monitoring. In contrast, Sukumaran et al. [27] built an online-engagement monitor combining facial emotion, gaze, head pose, and the blink rate. It used the Haar Cascade technique for face detection, a MobileNetV2 CNN trained on FER-2013 for emotions, and Dlib landmarks for gaze/pose. Modalities fed an Engagement Indicator that classified Highly Engaged, Confused, Boredom, and Sleepy. The system was validated by correlating engagement predictions with quiz scores from 10 students over a three-day period, with the FER component achieving 73.4% test accuracy. Taking a more sophisticated approach, Tang et al. [24] designed MP-FERS, a real-time system for measuring emotional learning engagement (ELE) using a PAD-based framework and ViT. The system was evaluated on video data from 108 students across six physics lessons, supplemented with academic records and self-reports. It achieved 92.21% accuracy on benchmark tasks, successfully captured fine-grained ELE variations across teacher-, interactive-, and student-centered instructional styles, and demonstrated stronger correlation with academic achievement than self-reported engagement metrics. Similarly aiming for real-time performance, Wang et al. [28] developed a MediaPipe-based, real-time e-learning engagement monitor that fused head posture, the blink rate (dynamic EAR), gaze, and facial emotion into composite metrics like distraction/smile ratios. The system was trained on a diverse dataset combining GENKI-4K, CelebA, and a synthetic HRFS set generated via Stable Diffusion. It showed high accuracy and efficiency: the head-posture model outperformed ML baselines, and XGBoost smile detection reached 98.53%, enabling deployment on low-resource edge devices. Expanding beyond visual modalities, Rehman et al. [31] introduced an EEG-based attention detector for e-learning using a Double Deep Q-Network (DDQN). After wavelet denoising and spectral/time-frequency feature extraction, the DDQN classified attentive, non-attentive, and drowsy states, optimized with a +10/−1 reward scheme. The model was trained on 34 EMOTIV-recorded experiments comprising over 1.5 million data points, achieving 98.2% accuracy—a 6% improvement over state-of-the-art methods—with low loss (0.65) and robust reward metrics, demonstrating its effectiveness for real-time adaptive online learning applications. More recently, Ferreira et al. [29] designed a deep learning framework that infers distance-learning engagement from facial expressions using a cascaded YOLOv8 for face detection, a modified ResNet-50 for hierarchical features, and an SVM for classifying four levels of engagement (none, low, medium, high). The framework was trained on labeled real-classroom images enhanced with advanced data augmentation, achieving 94.5% precision, 92.3% recall, and 93.7% mAP@0.5—surpassing the performance of YOLOv5, ViT, and Faster R-CNN benchmarks.

3. Proposed Methodology

In this section, we detail our Tiny CNN–BiLSTM–GRU pipeline for attention-level recognition, covering adaptive luminance normalization and sequence construction, the multi-branch spatiotemporal encoder, training objectives, and the deployment-oriented compression recipe (structured pruning, `INT8` quantization) that enables efficient on-device inference.

3.1. Data Description

In this work, we use the DAiSEE (Dataset for Affective States in E-Environments) dataset [39], a publicly available benchmark designed to support research in affective computing within digital learning environments. DAiSEE provides a rich collection of video data that captures spontaneous student behavior, making it particularly well suited for attention-level analysis in real-world scenarios. The dataset comprises a total of 9068 short

video snippets collected from 112 participants. Each snippet has a duration ranging between 10 and 30 s and was recorded using standard RGB webcams in naturalistic, non-laboratory environments. The dataset exhibits considerable variability in conditions, including changes in lighting, head pose, occlusion, background, and participant demographics. These factors contribute to the realism and complexity of the data, providing a challenging yet representative benchmark for affective state recognition.

Each video snippet in DAiSEE is annotated for four affective states: *Bored*, *Engaged*, *Confused*, and *Frustrated*. These annotations are provided at the snippet level and categorized into four ordinal intensity levels: *Very Low*, *low*, *high*, and *Very High*. In order to ensure the reliability of annotations, a crowdsourcing approach was employed, followed by refinement using the Dawid–Skene algorithm, which aggregates multiple annotator inputs into a consensus label. In our study, we focus exclusively on the *attention* dimension, which we frame as a four-class classification problem. The labels are provided in structured JSON files, each mapping video identifiers to their corresponding annotated intensity levels for the four affective states.

3.2. Data Preprocessing

To ensure compatibility with the CNN–BiLSTM–GRU architecture, we implement a multi-stage preprocessing pipeline comprising the following.

3.2.1. Video Frame Extraction

The process begins by converting each video snippet into a sequence of frames to serve as the input to the CNN. This is achieved by extracting frames at a fixed frame rate (typically 5 fps) to maintain an optimal balance between temporal resolution and computational efficiency. To ensure consistent input dimensions across all videos, the system standardizes the frame sequences by truncating longer videos to a fixed number of frames (e.g., 30 frames) while padding shorter sequences with either black frames or frame repetition. The implementation utilizes OpenCV for efficient frame extraction and processing [40].

3.2.2. Frame Resizing

To ensure compatibility with standard CNN architectures, all extracted frames are resized to a uniform dimension of 224 × 224 pixels. This standardization serves two key purposes: first, it meets the input requirements of popular CNN models; second, it maintains consistency for any custom network layers that may be implemented. The 224 × 224 resolution was specifically chosen as it represents the conventional input size for many pre-trained models while preserving sufficient spatial information for accurate feature extraction.

3.2.3. Brightness Adjustment

To address variability in illumination across the dataset, we employed an adaptive brightness normalization [41] technique rather than applying random augmentations. Specifically, we computed the average brightness of each frame in the YCbCr color space and adjusted its luminance channel toward a target reference level. This approach allowed us to enhance the visibility of darker frames while preserving the natural appearance of well-lit images. By normalizing brightness adaptively on a per frame basis, we maintained consistency in visual quality across the dataset without introducing artificial distortions. This technique improved the model's robustness to lighting variations commonly encountered in unconstrained video recordings.

3.2.4. Normalization

To enhance model convergence during training, all frames undergo pixel-wise normalization using standardized ImageNet statistics. Each channel is processed separately with mean values [0.485, 0.456, 0.406] and standard deviations [0.229, 0.224, 0.225] for the red, green, and blue channels respectively. This per channel normalization [42] scheme helps maintain consistent input distributions across the dataset while preserving the relative color information critical for visual recognition tasks.

3.2.5. Sequence Construction

For effective temporal modeling, extracted frames are organized into structured sequences. Each video is converted into a 4D tensor with dimensions (sequence_length, 3, 224, 224), representing an ordered series of RGB frames standardized to 224 × 224 resolution. This tensor format serves as the optimal input for hybrid CNN + RNN architectures, where the sequence_length dimension preserves temporal ordering while the spatial dimensions (3, 224, 224) maintain compatibility with standard convolutional operations. The processed sequences are then stored with their corresponding labels to maintain the video–class associations throughout the training pipeline.

3.2.6. Label Encoding

To prepare the DAiSEE dataset for classification tasks, textual attention labels are converted into numerical classes through integer encoding [43]. The mapping follows an ordinal scale where 'Very Low' corresponds to class 0, 'low' to 1, 'high' to 2, and 'Very High' to 3. These labels are systematically extracted by parsing the original JSON annotation files associated with each video, ensuring accurate video–label pairings. This numerical representation enables efficient processing by machine learning algorithms while maintaining the inherent ordinal relationship between different attention levels in the dataset.

3.2.7. Dataset Splitting

To ensure fair model evaluation and reliable generalization, the dataset is partitioned into training (80%), validation (10%), and test (10%) sets. These splits follow a rigorous subject-independent protocol, where each participant's data appears exclusively in only one subset, preventing data leakage and ensuring that the model's performance reflects true generalization to unseen individuals.

3.2.8. Explicit Verification Procedure for Leakage Prevention

To make the subject-independent protocol fully explicit, we implemented the split in a strictly ordered manner: *(1)* subjects are partitioned first, *(2)* all videos belonging to a subject inherit the same partition, and only then *(3)* temporal windows are generated separately inside each partition. This order is critical because creating overlapping windows before the split could allow highly correlated clips from the same subject or video to appear in different subsets. In our protocol, the split is therefore performed on subject identifiers before any sequence construction, which guarantees that no frame, no temporal window, and no video segment from a given subject can be shared across training, validation, and test sets.

Formally, let $\mathrm{id}_{\mathrm{sub}}(n)$ denote the subject identifier associated with video $V^{(n)}$, and let $\mathrm{id}_{\mathrm{vid}}(n)$ denote its video identifier. For each generated sequence $\mathbf{S}_j^{(n)}$, we store the metadata tuple

$$m_j^{(n)} = \left(\mathrm{id}_{\mathrm{sub}}(n),\ \mathrm{id}_{\mathrm{vid}}(n),\ t_j^{(n)},\ t_j^{(n)} + T - 1\right), \tag{1}$$

where $t_j^{(n)}$ is the first frame index of the sequence, and $t_j^{(n)} + T - 1$ is the last frame index in that window. Using these metadata tuples, we verify the split with the following deterministic conditions:

$$\mathcal{P}_{\text{train}} \cap \mathcal{P}_{\text{val}} = \emptyset, \quad \mathcal{P}_{\text{train}} \cap \mathcal{P}_{\text{test}} = \emptyset, \quad \mathcal{P}_{\text{val}} \cap \mathcal{P}_{\text{test}} = \emptyset, \tag{2}$$

$$\mathcal{V}_{\text{train}} \cap \mathcal{V}_{\text{val}} = \emptyset, \quad \mathcal{V}_{\text{train}} \cap \mathcal{V}_{\text{test}} = \emptyset, \quad \mathcal{V}_{\text{val}} \cap \mathcal{V}_{\text{test}} = \emptyset, \tag{3}$$

where $\mathcal{V}_{\text{train}}$, $\mathcal{V}_{\text{val}}$, and $\mathcal{V}_{\text{test}}$ denote the sets of video identifiers present in each split. Equation (2) guarantees that no subject is shared across subsets, and Equation (3) provides the stronger check that no video is split across partitions.

In addition, because windows may overlap *within* the same video, we explicitly verify that no sequence metadata tuple from one split shares its subject or video origin with a tuple from another split. Concretely, for any two sequences $\mathbf{S}_a^{(n)} \in \mathcal{D}_{\text{train}}$ and $\mathbf{S}_b^{(m)} \in \mathcal{D}_{\text{test}}$ (and analogously for the validation split), we require

$$\text{id}_{\text{sub}}(n) \neq \text{id}_{\text{sub}}(m) \quad \text{and} \quad \text{id}_{\text{vid}}(n) \neq \text{id}_{\text{vid}}(m). \tag{4}$$

This condition guarantees that two sequences from different splits cannot originate from the same subject or the same video, regardless of overlap ratio, stride, or sequence length. Therefore, any overlap induced by the sliding-window construction is confined to a single partition and cannot introduce train–test contamination.

Finally, we also prevent *statistical leakage* in preprocessing. The channel-wise normalization statistics (μ_c, σ_c) are computed *only* from frames belonging to $\mathcal{D}_{\text{train}}$, and the same values are then reused unchanged for validation and test frames. No statistics from $\mathcal{D}_{\text{val}}$ or $\mathcal{D}_{\text{test}}$ are used during preprocessing or hyperparameter selection. Together, these subject-, video-, sequence-, and normalization-level checks ensure a strictly leakage-free subject-independent protocol (see Table 1).

Table 1. Verification checks used to ensure strict train/validation/test separation.

Check	Purpose
Subject-ID disjointness	Ensures that no subject appears in more than one split.
Video-ID disjointness	Ensures that no video is fragmented across different splits.
Sequence-origin verification	Confirms that each temporal window inherits a unique subject/video origin from its own split.
Split-before-windowing rule	Prevents overlapping windows from being generated before the subject partition.
Training-only normalization statistics	Prevents validation/test information from influencing preprocessing.

3.2.9. Final Dataset Structure

Each processed sample in the dataset consists of: (1) a 4D tensor of shape (T, 3, 224, 224) representing the temporal sequence of RGB frames, and (2) a corresponding integer class label (0–3) indicating the attention level. This standardized structure is specifically designed for hybrid deep learning architectures, where the tensor dimensions accommodate CNN-based spatial feature extraction (processing 224 × 224 RGB images) followed by BiLSTM-GRU networks that analyze the temporal evolution of features across the T frames. The label encoding preserves the ordinal relationship between different attention states while being optimized for classification tasks.

To make clear that the final performance is not due only to clip construction and normalization, we later report a controlled ablation that separates the gains obtained

from preprocessing from those obtained from the proposed spatiotemporal architecture (Section 4.3.8).

3.3. Introducing the Proposed Model

In this section, we describe the deep components that underpin the proposed architecture, namely a frame-wise 2D convolutional neural network (CNN), a Long Short-Term Memory (LSTM), a Bidirectional LSTM (BiLSTM), and a Gated Recurrent Unit (GRU), followed by the complete hybrid CNN–BiLSTM–GRU model for attention-level recognition. The proposed model is designed for video clips represented as ordered sequences of RGB frames and learns both spatial facial cues and their temporal evolution.

Let one input clip be denoted by

$$\mathcal{X} = \{X_t\}_{t=1}^{T}, \qquad X_t \in \mathbb{R}^{H\times W\times C}, \tag{5}$$

where T is the number of frames in the clip, H and W are the frame height and width, and $C = 3$ is the number of RGB channels. In this work, each frame is resized to 224×224; hence, $H = W = 224$. The CNN processes each frame independently and maps it to a d-dimensional embedding

$$z_t = f_{\text{cnn}}(X_t; \Theta_{\text{cnn}}), \qquad z_t \in \mathbb{R}^d, \tag{6}$$

where Θ_{cnn} denotes the CNN parameters. The temporal module then receives the ordered embedding sequence $\{z_t\}_{t=1}^{T}$.

To keep the notation consistent, we use (h_t, c_t) for LSTM hidden and cell states, $(\overrightarrow{h}_t, \overleftarrow{h}_t)$ for the forward and backward BiLSTM hidden states, and g_t for the GRU hidden state. The classifier outputs one of the four DAiSEE attention classes (see Table 2):

$$\mathcal{C} = \{\text{Very Low}, \text{Low}, \text{High}, \text{Very High}\},$$

so the final dense layer always has output dimension $K = 4$.

Table 2. Summary of the main symbols used in the proposed model formulation.

Symbol	Meaning
$\mathcal{X} = \{X_t\}_{t=1}^{T}$	Input clip of T-ordered RGB frames
$X_t \in \mathbb{R}^{H\times W\times C}$	Frame at time step t
H, W, C	Frame height, width, and number of channels ($C = 3$)
$z_t \in \mathbb{R}^d$	CNN embedding of frame X_t
d	CNN embedding dimension
h_t, c_t	Hidden state and cell state of the LSTM
$\overrightarrow{h}_t, \overleftarrow{h}_t$	Forward and backward BiLSTM hidden states
$u_t \in \mathbb{R}^{2h}$	Concatenated BiLSTM representation at time t
$g_t \in \mathbb{R}^g$	Hidden state of the GRU
T	Number of frames per clip
K	Number of output classes ($K = 4$)
$\hat{p} \in \mathbb{R}^K$	Predicted posterior probability vector
$\hat{y}$	Predicted attention label
Θ_{cnn}	Learnable parameters of the CNN
$\sigma(\cdot)$	Sigmoid activation function
$\tanh(\cdot)$	Hyperbolic tangent activation
$\odot$	Element-wise (Hadamard) product

3.3.1. Convolutional Neural Network (CNN)

We employed a compact *2D CNN* [44] to extract spatial descriptors from each video frame before temporal modeling (see Figure 1). The CNN is applied independently to every RGB frame and therefore performs standard spatial convolutions, not 1D convolutions on a temporal signal. Starting from an input frame $X_t \in \mathbb{R}^{224\times224\times3}$, the network applies two convolutional blocks, each followed by batch normalization, ReLU, and 2×2 max pooling, before flattening or global pooling and projecting the output to a compact embedding $z_t \in \mathbb{R}^d$.

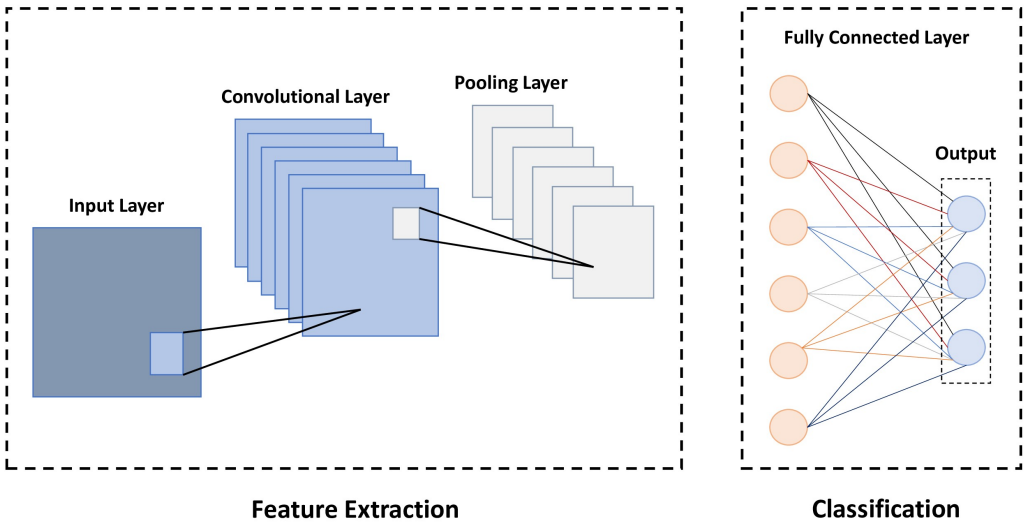


Figure 1. Frame-wise 2D CNN used for spatial feature extraction. Each RGB frame is processed independently through convolution, normalization, activation, and pooling, followed by a projection to the embedding z_t.

Let $A^{(\ell-1)} \in \mathbb{R}^{C_{\ell-1}\times H_{\ell-1}\times W_{\ell-1}}$ denote the input tensor of convolutional layer ℓ. The kth output feature map is computed as

$$A_k^{(\ell)} = \mathrm{ReLU}\Big(W_k^{(\ell)} * A^{(\ell-1)} + b_k^{(\ell)}\Big), \tag{7}$$

where $*$ denotes 2D convolution, $W_k^{(\ell)}$ is the learnable kernel of the kth filter, and $b_k^{(\ell)}$ is its bias. Max pooling is then used to reduce spatial resolution:

$$P_k^{(\ell)}(i,j) = \max_{(u,v)\in\{0,1\}^2} A_k^{(\ell)}(2i+u,\, 2j+v). \tag{8}$$

After the second pooling stage, the feature tensor is flattened or globally averaged and projected into the frame embedding space:

$$z_t = \phi\Big(W_{\mathrm{proj}}\,\mathrm{vec}(P^{(2)}) + b_{\mathrm{proj}}\Big) \in \mathbb{R}^d, \tag{9}$$

where $\phi(\cdot)$ denotes ReLU followed by optional dropout. When the CNN is used as a standalone classifier, the class posterior for class k is

$$\hat{p}_k = \frac{\exp(w_k^\top z_t + b_k)}{\sum_{j=1}^{K}\exp(w_j^\top z_t + b_j)}, \qquad k = 1,\ldots,K. \tag{10}$$

We selected a shallow two-block CNN because it captures mid-level spatial cues, such as eye opening, mouth shape, and head orientation, while keeping the parameter

count manageable. The output embedding sequence $\{z_t\}_{t=1}^{T}$ is then passed to the recurrent temporal layers.

3.3.2. Long Short-Term Memory (LSTM)

We employed LSTM [45] units to capture temporal dependencies across the frame embeddings. Given the input sequence $\{z_t\}_{t=1}^{T}$, the LSTM processes one embedding at a time while maintaining a hidden state $h_t \in \mathbb{R}^h$ and a cell state $c_t \in \mathbb{R}^h$. At each time step t, the update equations are

$$i_t = \sigma(W_i z_t + U_i h_{t-1} + b_i), \tag{11}$$
$$f_t = \sigma(W_f z_t + U_f h_{t-1} + b_f), \tag{12}$$
$$o_t = \sigma(W_o z_t + U_o h_{t-1} + b_o), \tag{13}$$
$$\tilde{c}_t = \tanh(W_c z_t + U_c h_{t-1} + b_c), \tag{14}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \tag{15}$$
$$h_t = o_t \odot \tanh(c_t), \tag{16}$$

where i_t, f_t, and o_t are the input, forget, and output gates, respectively. The matrices $W_\star$ and $U_\star$ and the biases $b_\star$ are learned parameters.

The LSTM is useful because it can preserve, update, or discard temporal evidence over long clips, making it well suited to modeling gradual changes in attention-related cues, such as gaze shifts, eyelid movement, and head motion (see Figure 2).

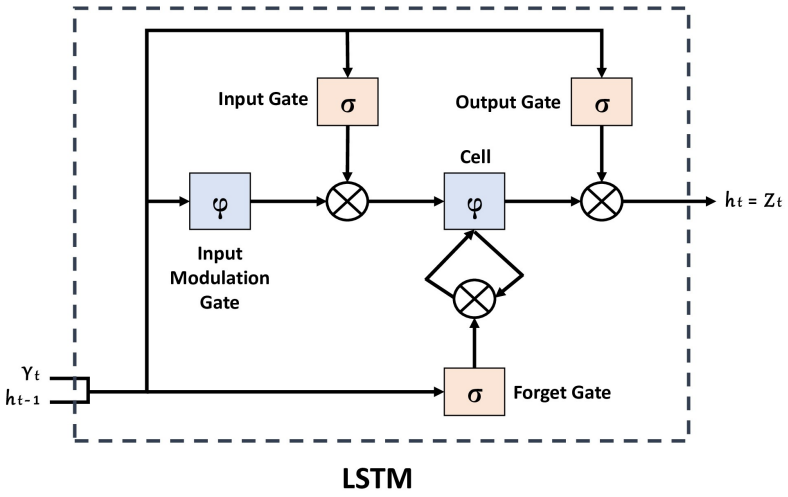


Figure 2. LSTM memory cell. The input, forget, and output gates regulate information flow through the cell state c_t.

3.3.3. Bidirectional Long Short-Term Memory (BiLSTM)

We adopted BiLSTM [46] to exploit temporal cues from both past and future frames within each clip (see Figure 3). Given the CNN embedding sequence $\{z_t\}_{t=1}^{T}$, the forward and backward LSTMs are defined as

$$\overrightarrow{h}_t, \overrightarrow{c}_t = \text{LSTM}_{\rightarrow}(z_t, \overrightarrow{h}_{t-1}, \overrightarrow{c}_{t-1}), \qquad t = 1, \ldots, T, \tag{17}$$
$$\overleftarrow{h}_t, \overleftarrow{c}_t = \text{LSTM}_{\leftarrow}(z_t, \overleftarrow{h}_{t+1}, \overleftarrow{c}_{t+1}), \qquad t = T, \ldots, 1. \tag{18}$$

The bidirectional representation at time step t is obtained by concatenation:

$$u_t = [\, \overrightarrow{h}_t ; \overleftarrow{h}_t \,] \in \mathbb{R}^{2h}. \tag{19}$$

This formulation enables the model to interpret each frame using information from the full clip instead of only the preceding frames. In attention recognition, that is particularly useful because transient events, such as blinking or momentary distraction, are more informative when interpreted with the surrounding temporal context.

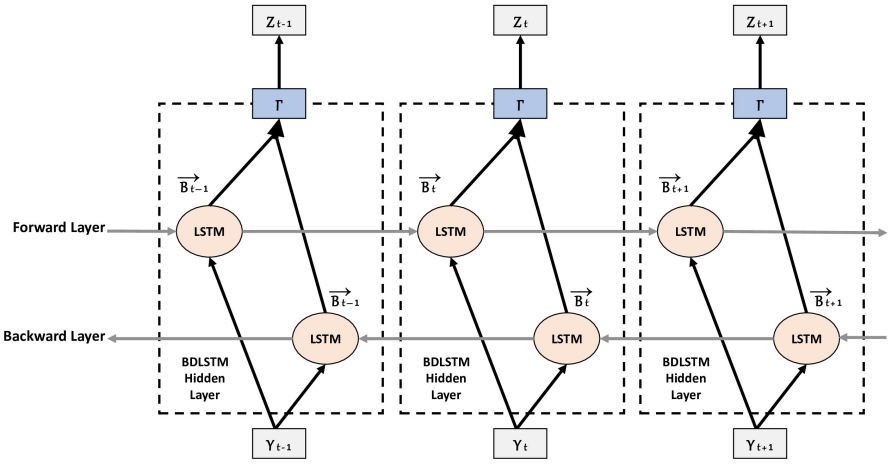


Figure 3. BiLSTM temporal encoder. Forward and backward LSTMs process the same clip in opposite directions, and their hidden states are concatenated at each time step.

3.3.4. Gated Recurrent Unit (GRU)

We employed the GRU [47] to further refine the temporal representation while keeping the parameter count lower than that of an additional LSTM (see Figure 4). The GRU receives the BiLSTM sequence $\{u_t\}_{t=1}^{T}$ as the input and maintains a hidden state $g_t \in \mathbb{R}^g$. Its update equations are

$$r_t = \sigma(W_r u_t + U_r g_{t-1} + b_r), \tag{20}$$

$$z_t^{(\mathrm{upd})} = \sigma(W_z u_t + U_z g_{t-1} + b_z), \tag{21}$$

$$\tilde{g}_t = \tanh\big(W_g u_t + U_g (r_t \odot g_{t-1}) + b_g\big), \tag{22}$$

$$g_t = (1 - z_t^{(\mathrm{upd})}) \odot g_{t-1} + z_t^{(\mathrm{upd})} \odot \tilde{g}_t. \tag{23}$$

Here, r_t is the reset gate, $z_t^{(\mathrm{upd})}$ is the update gate, and $\tilde{g}_t$ is the candidate hidden state. We use g_t for the GRU state to avoid confusion with the LSTM/BiLSTM hidden states.

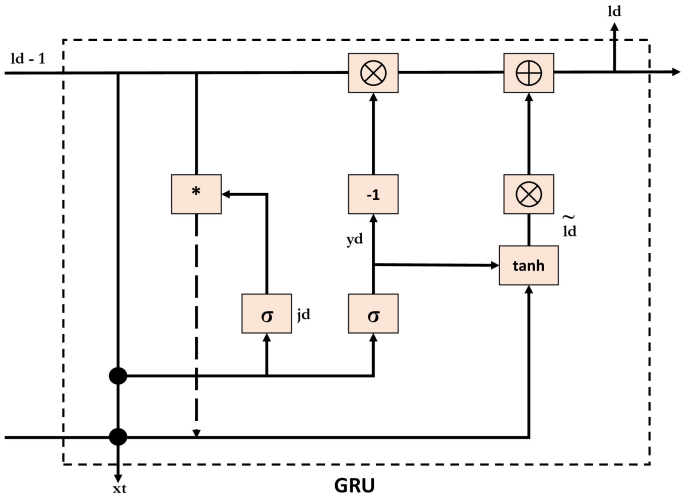


Figure 4. GRU cell. The reset and update gates regulate the contribution of the previous state when generating the new hidden state g_t.

The GRU is placed after the BiLSTM because it provides a lightweight gating mechanism that smooths the bidirectional representation and compresses it into a compact sequence code with fewer parameters than stacking another LSTM.

3.3.5. Proposed Model

The proposed model is a hybrid sequence architecture that combines a frame-wise *2D CNN* with BiLSTM and GRU temporal encoders to capture both spatial and temporal cues associated with student attention. The model input is a clip of T RGB frames, each of spatial size $224 \times 224 \times 3$. Each frame is first encoded by the CNN into a low-dimensional embedding z_t, and the ordered embedding sequence $\{z_t\}_{t=1}^{T}$ is then processed by the recurrent temporal stack.

The CNN extracts spatial patterns from each frame, including eye appearance, mouth configuration, and head pose. The BiLSTM then models bidirectional temporal context across the clip, and the GRU further refines the resulting sequence representation. After temporal processing, max pooling over time is applied to emphasize the most salient temporal activations. The resulting clip-level descriptor is finally passed to a softmax classifier with **four outputs**, corresponding exactly to the four DAiSEE attention classes: *Very Low*, *low*, *high*, and *Very High*.

To improve robustness, we instantiate three parallel CNN–BiLSTM–GRU branches with different widths and dropout strengths. Let $q^{(b)} \in \mathbb{R}^{m_b}$ denote the pooled output of branch $b \in \{1, 2, 3\}$. The multi-branch representation is formed by concatenation:

$$q = [\, q^{(1)}; q^{(2)}; q^{(3)} \,]. \tag{24}$$

The final classifier then predicts the attention posterior as

$$\hat{p} = \mathrm{softmax}(W_{\mathrm{cls}} q + b_{\mathrm{cls}}), \qquad \hat{p} \in \mathbb{R}^4, \tag{25}$$

and the predicted class is

$$\hat{y} = \arg \max_{k \in \{1,2,3,4\}} \hat{p}_k. \tag{26}$$

This formulation makes the model specification fully consistent with the task definition: the visual backbone is 2D CNN-based, not CNN-1D, and the classifier is a 4-way softmax, not a 6-way dense output. Figure 5 summarizes the complete pipeline.

Although the proposed architecture belongs to the general family of CNN–RNN video models, its contribution does not lie in introducing a new recurrent cell in isolation, but in the specific design and integration strategy used for attention-level recognition. Conventional CNN–LSTM or CNN–GRU hybrids typically employ a single recurrent layer after frame-wise CNN features, which means that temporal modeling is performed either only in the forward direction or with a single gating mechanism. In contrast, our model uses a two-stage temporal encoder: a BiLSTM first captures context from both past and future frames, and a subsequent GRU then refines and compresses this bidirectional representation into a more compact temporal code. This ordering is intentional: the BiLSTM is used to maximize contextual coverage, whereas the GRU is used to filter short-lived noise and preserve persistent attention-related dynamics with a lower parameter overhead than stacking another LSTM-like block.

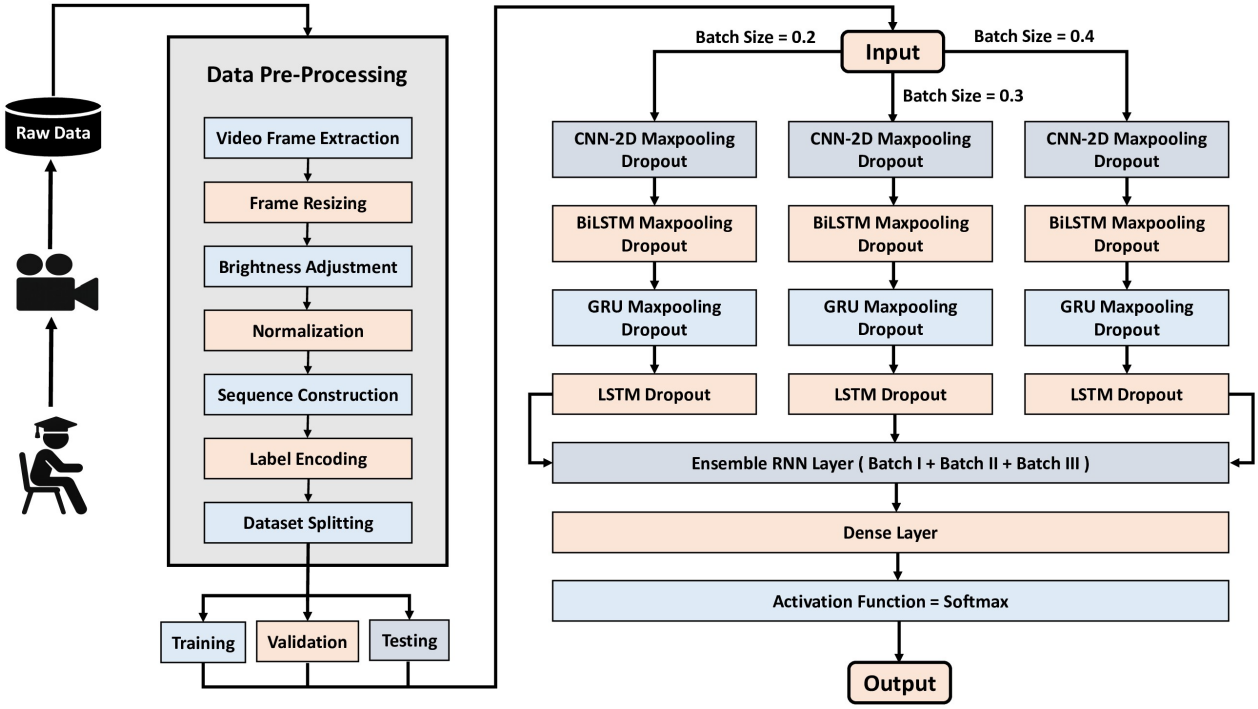


Figure 5. Flowchart of the proposed hybrid architecture. Each branch performs frame-wise spatial encoding with a 2D CNN, followed by BiLSTM-based temporal aggregation and GRU-based refinement. The three branch outputs are concatenated and classified by a 4-way softmax head.

A second methodological difference is the use of parallel heterogeneous branches rather than a single monolithic CNN–RNN pipeline. The three branches differ in width and dropout strength, which encourages them to learn complementary spatial–temporal representations under different regularization regimes. Their outputs are then fused into a shared clip-level descriptor before final classification. This is different from standard single-branch CNN–RNN models, where all temporal evidence is forced through one representation stream. In our case, the multi-branch design improves robustness to subject variability, transient facial motions, and pose changes while preserving a lightweight structure. Therefore, the novelty of the proposed model is best understood as a structured hybridization of 2D frame-wise spatial encoding, bidirectional temporal aggregation, gated temporal refinement, and multi-branch fusion for four-level attention classification, rather than as a claim of inventing a completely new CNN or recurrent unit (see Table 3).

Table 3. Methodological comparison between common CNN–RNN hybrids and the proposed architecture.

Architecture Family	Spatial Encoder	Temporal Encoder	Multi-Branch Fusion	Main Limitation/Distinction
CNN-only	2D CNN	None	No	Captures spatial cues only; no explicit temporal modeling
CNN–LSTM	2D CNN	LSTM	No	Uses only forward temporal modeling
CNN–BiLSTM	2D CNN	BiLSTM	No	Uses bidirectional context but no gated refinement stage
CNN–GRU	2D CNN	GRU	No	Lightweight temporal modeling but less contextual coverage than BiLSTM
3D CNN/Conv3D	Spatiotemporal convolution	Implicit	No	Higher computational cost; less explicit temporal interpretability
Proposed model	2D CNN	BiLSTM + GRU	Yes	Combines bidirectional context, gated refinement, and heterogeneous branch fusion

3.4. Model Compression

We sought to reduce the computational and memory footprint of our hybrid CNN–BiLSTM–GRU while preserving its accuracy on attention-level recognition. Because our architecture comprised heterogeneous components, we adopted a heterogeneous compression strategy: structured pruning for convolutional channels and recurrent units, quantization for both convolutional and recurrent weights and activations, and an optional distillation stage to recover any accuracy lost after compression. All techniques were designed to be training-aware so that the final model remained stable under distributional variation in lighting, pose, and occlusion.

3.4.1. Objectives and Scope

Our goal was to obtain models that satisfied three constraints simultaneously: a reduction in parameter count and arithmetic intensity, compatibility with standard deployment toolchains (PyTorch/ONNX/ONNX Runtime or TensorRT), and negligible degradation in the four-class attention-level accuracy. Let Θ denote all learnable parameters and let $\mathcal{L}_{\text{task}}(\Theta)$ be the cross-entropy loss on the training set. Compression was formulated as an optimization with sparsity- and precision-inducing terms,

$$\min_{\Theta,\,\mathcal{M}} \; \mathcal{L}_{\text{task}}(\Theta \odot \mathcal{M}) \;+\; \lambda_{\text{s}}\,\Omega(\mathcal{M}) \;+\; \lambda_{\text{q}}\,\mathcal{E}_{\text{quant}}(\Theta), \tag{27}$$

where $\mathcal{M}$ are binary or relaxed pruning masks, $\Omega(\cdot)$ encouraged structured sparsity, and $\mathcal{E}_{\text{quant}}(\cdot)$ penalized quantization distortion. The operator $\odot$ indicates parameter masking. We solved (27) by alternating between mask learning and fine-tuning and then by calibrating or training the quantized model.

3.4.2. Structured Pruning

Structured pruning [48] removed entire channels or neurons, producing real wall-clock gains on commodity hardware. For each convolutional layer with kernel tensor $\mathbf{W} \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}} \times k \times k}$, we ranked output channels by an importance score S_c. In the magnitude criterion,

$$S_c^{\ell_1} \;=\; \|\mathbf{W}_{c,:,:,:}\|_1, \qquad \mathcal{M}_c \;=\; \mathbb{I}(S_c \geq \tau), \tag{28}$$

where τ is a threshold chosen to meet a target sparsity. We also evaluated a first-order Taylor criterion,

$$S_c^{\text{taylor}} = |\langle \nabla_{\mathbf{W}_c} \mathcal{L}_{\text{task}}, \mathbf{W}_c \rangle|, \tag{29}$$

which approximated the loss increase after removal. Channels with the smallest scores were pruned, and batch-normalization statistics were recomputed. For recurrent layers, we pruned hidden units rather than individual weights. Let $\mathbf{U} \in \mathbb{R}^{h \times h}$ denote a recurrent weight matrix; the jth unit importance was

$$S_j^{\text{rnn}} \;=\; \alpha\,\|\mathbf{U}_{j,:}\|_2 \;+\; (1-\alpha)\,\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\big[\,|h_{t,j}|\,\big], \tag{30}$$

a convex combination of parameter magnitude and average activation across time and mini-batches. Units with the smallest S_j^{rnn} were removed together with the corresponding columns/rows in all gate matrices to maintain shape consistency. After pruning, we fine-tuned the network to recover accuracy. The fine-tuning objective reused (27) with fixed masks $\mathcal{M}$.

To quantify complexity reductions, we tracked parameter counts and multiply–accumulate operations (MACs). If the pruning ratio at a convolutional layer was ρ on the

output channels and ρ' on the input channels resulting from upstream pruning, the MAC reduction factor approximated $(1-\rho)(1-\rho')$. For a LSTM of width h and input size d, the dominant cost $4h(d+h)$ MACs per time step decreased to $4h'(d'+h')$ after pruning to widths h' and d'.

3.4.3. Quantization

Quantization [49] mapped floating-point tensors to low-precision integer representations. We adopted per channel symmetric INT8 quantization for convolutional weights and per tensor asymmetric INT8 for activations, with dynamic quantization for recurrent weights to avoid accuracy loss in tightly coupled gate computations. The affine mapping used by ONNX Runtime was

$$\mathrm{int8}(\mathbf{t}) \;=\; \mathrm{clip}\!\left(\mathrm{round}\!\left(\frac{\mathbf{t}}{s}\right)+z,\; -128,\; 127\right), \quad \hat{\mathbf{t}} \;=\; s(\mathrm{int8}(\mathbf{t})-z), \tag{31}$$

where $s>0$ is the scale and $z\in\mathbb{Z}$ the zero-point. For static quantization of convolutions, we estimated s and z from percentile or KL-divergence calibration on a few hundred clips; for dynamic quantization of LSTM/GRU weights, scales were computed on the fly at inference, which preserved accuracy while delivering speedups on CPU backends.

To mitigate quantization error, we optionally performed quantization-aware training (QAT) by inserting fake-quantization operators during fine-tuning. Let $\mathcal{Q}_\phi(\cdot)$ denote a differentiable proxy for (31); the training loss became

$$\min_{\Theta}\; \mathbb{E}\big[\mathcal{L}_{\mathrm{task}}\big(\mathcal{Q}_\phi(\mathbf{x};\Theta)\big)\big] \;+\; \beta\sum_{\ell}\|\Theta_\ell-\mathcal{Q}_\phi(\Theta_\ell)\|_2^2, \tag{32}$$

where the second term penalized deviation between float and quantized weights to improve post-training calibration.

3.4.4. Compression Pipeline

The final pipeline proceeded in three phases. We first trained the baseline hybrid model to convergence with early stopping on validation accuracy. We then performed structured pruning with a gradual schedule $\pi(e)$ over epochs e, where the target sparsity rose linearly from 0 to $\rho_{\max}$; masks were updated every few epochs using (28)–(30). After achieving the desired sparsity, we fine-tuned until stabilization. Finally, we exported the masked network to ONNX [50] and applied post-training quantization; for the RNN blocks, we enabled dynamic quantization, and for the CNN and dense layers, we used static calibration with percentile clipping. If accuracy fell below the preset margin (typically <0.5 pp from baseline), we enabled QAT [51] on the most sensitive layers.

3.4.5. Complexity and Memory Analysis

Let P_{fp32} and P_{int8} denote model sizes with 32-bit and 8-bit weights, respectively. Quantization alone provided a $4\times$ reduction in weight memory, i.e., $|P_{\mathrm{int8}}|\approx 0.25\,|P_{\mathrm{fp32}}|$, and further gains accrued from structured pruning. If the total kept-channel fraction across convolutions was κ_{cnn} and the kept-unit fraction across recurrent layers was κ_{rnn}, the parameter footprint scaled approximately as

$$|P_{\mathrm{pruned,int8}}| \;\approx\; 0.25\,\big(\kappa_{\mathrm{cnn}}^2\,|P_{\mathrm{cnn}}| \;+\; \kappa_{\mathrm{rnn}}^2\,|P_{\mathrm{rnn}}| \;+\; |P_{\mathrm{head}}|\big), \tag{33}$$

since both input and output channel counts affected convolutional weight tensors quadratically. For latency we used MAC counts as a proxy; after pruning and with sequence length T, the overall complexity became

$$\text{MACs} \approx \sum_{\ell \in \text{CNN}} (1-\rho_\ell)(1-\rho'_\ell)\,\text{MACs}_\ell + T \sum_{r \in \{\text{BiLSTM,GRU}\}} \text{MACs}_r(h'_r, d'_r), \quad (34)$$

where (ρ_ℓ, ρ'_ℓ) are output/input pruning ratios at layer ℓ and h'_r, d'_r are the pruned widths for recurrent block r. These expressions matched the empirical reductions measured with ONNX Runtime on the CPU.

3.5. Evaluation Protocol and Analysis of DAiSEE Results

This subsection therefore makes the evaluation setup explicit, reports the class distribution and per class metrics, and describes the measures taken to avoid leakage or overly optimistic settings.

3.5.1. Clip-Level Prediction and Temporal Aggregation

All experiments on DAiSEE are performed at the clip level. Each original DAiSEE snippet is treated as a single training example with one ordinal label on the four-level attention scale (Very Low, low, high, Very High). There is no frame-level supervision or label replication across overlapping sub-clips. For every snippet we uniformly sample $T = 30$ frames at approximately 5 fps; if the video is shorter, frames are padded by repetition at the tail, and if it is longer, frames are sub-sampled without overlap. The CNN operates on each frame independently and produces a sequence of T embeddings. The BiLSTM–GRU stack consumes this entire sequence and produces a single clip-level representation by temporal max pooling over the BiLSTM outputs, followed by the final GRU hidden state. The softmax head then outputs a single posterior vector per clip.

Evaluation is therefore carried out strictly at the snippet level: each DAiSEE clip appears exactly once in the test set, and its ground-truth label is compared to a single predicted label. No majority voting or averaging over overlapping windows is used, and per frame scores are not counted as additional test examples.

3.5.2. Subject-Independent Split and Class Distribution

To avoid identity leakage, we construct an 80/10/10 partition at the subject level. All clips from a given participant are assigned to a single split. This yields 90 subjects for training, 11 for validation and 11 for testing, with a total of 7254, 907 and 907 clips, respectively.

The distribution is clearly imbalanced, with low and Very Low dominating, which is consistent with observations in earlier DAiSEE studies. To mitigate this imbalance during training we use class-weighted cross-entropy, as described in Section 4.1, but all metrics are reported on the raw, unweighted test set.

3.5.3. Checks Against Leakage and Overly Simplified Evaluation

Because the obtained numbers are near the upper end of what has been reported for DAiSEE, we performed several sanity checks to rule out inadvertent leakage or overly optimistic settings.

First, we verified that no subject appears in more than one split by cross-checking the anonymized subject identifiers provided with the dataset. Second, we confirmed that all clips from a given video file remain within the same split and that no overlapping subsequences are extracted across splits. Third, we trained a set of simpler baselines on exactly the same subject-independent split. For example, a shallow CNN followed by a single unidirectional LSTM achieves 95–96% accuracy and macro-F1 around 0.95, while a purely

frame-based CNN classifier evaluated at the clip level (by averaging frame posteriors) remains below 93% accuracy. These baselines demonstrate that the protocol is non-trivial and that the gains reported for the proposed hybrid architecture are due to the combination of stronger temporal modeling, sequence-level aggregation and regularization, rather than due to task leakage.

Finally, we repeated training with three different random seeds and observed variations below 0.1 percentage points in accuracy and macro-F1, which suggests that the results are stable and not due to a lucky initialization. Together with the subject-independent partitioning, the class-wise metrics and the confusion analysis, these checks provide additional evidence that the reported near-perfect performance reflects genuine discriminative ability under the chosen protocol, rather than artefacts of the evaluation procedure.

4. Experimental Results and Analyses

In this section, we report results on DAiSEE under a subject-independent split, detailing the experimental setup and metrics, presenting baseline versus compressed performance with per class scores, ordinal and calibration analyses, efficiency measurements, ablation studies (compression components and sequence length), robustness to classroom variations, and a head-to-head comparison with state-of-the-art methods.

4.1. Experimental Setup

We conducted all experiments on the DAiSEE dataset. Each video clip was represented as an ordered sequence of RGB frames; we fixed the spatial resolution at 224×224 pixels and the temporal length at $T = 30$ frames per clip (nominal sampling at 5 fps), which offered a balance between temporal coverage and computational cost. The CNN processed frames independently to produce per frame embeddings that were consumed by the temporal stack (BiLSTM followed by GRU), as described in Section 3.3.5. All reported results were computed on the test split without any exposure during training or model selection.

The baseline hybrid network was trained end-to-end with the AdamW optimizer (initial learning rate 10^{-3}, weight decay 10^{-4}) and a cosine learning-rate schedule with linear warm-up over the first five epochs. Unless otherwise stated, the batch size was 16 clips, the maximum number of epochs was 60, and early stopping monitored the validation macro-F1 with a patience of 10 epochs. We applied dropout within each branch exactly as specified in Tables 4–6 and gradient clipping at a global L_2 norm of 1.0 to stabilize recurrent training. Mixed-precision (FP16) training was enabled. To mitigate the inherent class imbalance in DAiSEE, we optimized a class-weighted cross-entropy,

$$\mathcal{L}_{\mathrm{CE}} = -\sum_{c=1}^{4} w_c \, y_c \log \hat{y}_c, \qquad w_c \propto \frac{1}{\mathrm{freq}(c)},$$

with weights w_c set inversely proportional to the empirical training-set frequencies and normalized to $\sum_c w_c$=4. During validation and testing we did not apply any augmentation.

Table 4. Parameter configuration of Branch I (dropout = 0.2).

Component	Units/Filters	Kernel Size	Return Sequence	Activation	Pool Size	Dropout
2D CNN + max pooling	128	4×4	–	ReLU	2	0.2
BiLSTM	64	–	True	tanh	–	0.2
GRU	32	–	True	tanh	–	0.2
Temporal pooling	–	–	–	–	2	–
Dense layer	4	–	–	Softmax	–	–
					Total params	119,312

Table 5. Parameter configuration of Branch II (dropout = 0.3).

Component	Units/Filters	Kernel Size	Return Sequence	Activation	Pool Size	Dropout
2D CNN + max pooling	256	4×4	–	ReLU	2	0.3
BiLSTM	128	–	True	tanh	–	0.3
GRU	64	–	True	tanh	–	0.3
Temporal pooling	–	–	–	–	2	–
Dense layer	4	–	–	Softmax	–	–
					Total params	472,048

Table 6. Parameter configuration of Branch III (dropout = 0.4).

Component	Units/Filters	Kernel Size	Return Sequence	Activation	Pool Size	Dropout
2D CNN + max pooling	128	4×4	–	ReLU	2	0.4
BiLSTM	128	–	True	tanh	–	0.4
GRU	64	–	True	tanh	–	0.4
Temporal pooling	–	–	–	–	2	–
Dense layer	4	–	–	Softmax	–	–
					Total params	339,312

Model compression followed the pipeline in Section 3.4. We first trained the full-precision baseline to convergence, then performed structured pruning with a gradual sparsity schedule that increased linearly from zero to the target level over the middle third of training, while freezing masks during the final fine-tuning phase. Convolutions were pruned by output channels using an ℓ_1-norm criterion; recurrent layers were pruned by hidden units using an activation-aware score. The pruned network was exported to ONNX and quantized. Convolutional and linear layers used per channel symmetric `INT8` weight quantization and per tensor `INT8` activations with percentile calibration on 512 clips drawn from the training set; LSTM/GRU weights used dynamic `INT8` quantization at inference. When accuracy dropped by more than 0.5 percentage points relative to the float baseline, we enabled quantization-aware fine-tuning for five additional epochs and, in the most aggressive sparsity setting, performed a short knowledge-distillation pass with temperature $\tau = 2$ and distillation weight $\gamma = 0.3$.

We reported overall accuracy, macro-precision, macro-recall, and macro-F1 on the test split, together with a confusion matrix to visualize per class behavior. For efficiency, we measured parameter count, model size on disk, multiply–accumulate operations (MACs) per clip at $T = 30$, and wall-clock latency on the CPU using ONNX Runtime with dynamic quantization enabled for recurrent layers. To account for stochasticity, each configuration was trained with three different random seeds, and we reported the mean and standard deviation ($\mu \pm \sigma$).

Table 7 summarizes the software/hardware environment, and Table 8 consolidates the principal hyperparameters used throughout unless otherwise specified in ablations.

Table 7. Computing environment used for all experiments.

Operating system	Ubuntu 22.04 LTS
Python/PyTorch/CUDA	Python 3.10 PyTorch 2.2 CUDA 12.1
Inference runtime	ONNX Runtime 1.17 (CPU: MKL-DNN; GPU: CUDA EP)
Hardware	1× NVIDIA RTX 3080 (10 GB), Intel Xeon-class CPU, 64 GB RAM
Determinism	Fixed seeds $\{13, 29, 47\}$; CuDNN deterministic kernels; hash seed fixed

Table 8. Training and compression hyperparameters (default values).

Aspect	Setting	Value(s)	Notes
Input representation	Frames per clip	$T = 30$ (5 fps)	224×224, RGB
Optimizer	AdamW	lr = 10^{-3}, wd = 10^{-4}	Cosine decay; warm-up 5 epochs
Regularization	Dropout	$\{0.2, 0.3, 0.4\}$ by branch	
Stabilization	Grad clip	L_2 norm 1.0	Applied each step
Batching	Batch size	16 clips	Mixed precision (FP16)
Early stopping	Criterion	Val. macro-F1, patience 10	Best checkpoint by macro-F1
Loss	Class-weighted CE	$w_c \propto 1/\text{freq}(c)$	Weights normalized to $\sum w_c = 4$
Pruning	Target sparsity	CNN: 30–50% channels; RNN: 20–40% units	Gradual schedule, mask frozen during FT
Quantization	Scheme	Conv/FC: static `INT8`; RNN: dynamic `INT8`	Percentile calibration on 512 clips
QAT/KD	Triggers	Acc. drop > 0.5 pp	QAT 5 epochs; KD $\tau = 2$, $\gamma = 0.3$
Evaluation	Seeds	3 runs	Report $\mu \pm \sigma$

4.2. Evaluation Metrics

We evaluated the proposed models along two complementary axes: (1) predictive quality on the four attention levels (Very Low, low, high, Very High), and (2) computational efficiency after compression. Because DAiSEE exhibits class imbalance and the labels are ordinal, we reported both class-aggregation metrics and ordinal/calibration measures. Unless stated otherwise, all metrics were computed on the held-out test split and averaged over three random seeds, yielding $\mu \pm \sigma$.

4.2.1. Confusion Matrix and per Class Rates

Let $\mathbf{C} \in \mathbb{N}^{4\times 4}$ denote the confusion matrix with entries C_{ij} counting examples of true class i predicted as class j. For a given class c, we defined

$$\text{TP}_c = C_{cc}, \qquad \text{FP}_c = \sum_{i\neq c} C_{ic}, \qquad \text{FN}_c = \sum_{j\neq c} C_{cj}, \qquad \text{TN}_c = \sum_{i\neq c}\sum_{j\neq c} C_{ij}. \tag{35}$$

Per class precision, recall, and F1-score were

$$\text{Prec}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c + \varepsilon}, \quad \text{Rec}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FN}_c + \varepsilon}, \quad \text{F1}_c = \frac{2\,\text{Prec}_c\,\text{Rec}_c}{\text{Prec}_c + \text{Rec}_c + \varepsilon}, \tag{36}$$

with ε a small constant for numerical stability. Overall accuracy was

$$\text{Acc} = \frac{\sum_c \text{TP}_c}{\sum_{i,j} C_{ij}}. \tag{37}$$

To summarize across classes under imbalance, we reported macro- and weighted-aggregates:

$$\text{Macro-F1} = \frac{1}{4}\sum_{c=1}^{4} \text{F1}_c, \qquad \text{Weighted-F1} = \sum_{c=1}^{4} \pi_c\,\text{F1}_c, \tag{38}$$

where $\pi_c = \frac{\sum_j C_{cj}}{\sum_{i,j} C_{ij}}$ is the empirical class prior. We visualized $\mathbf{C}$ row-normalized to highlight error modes (e.g., low vs. high confusions).

4.2.2. Ordinal Metrics

Because attention levels are ordered, we additionally evaluated the predictions on an ordinal scale by mapping classes to integers $\{0, 1, 2, 3\}$. The Mean Absolute Error (MAE) on the ordinal labels was

$$\mathrm{MAE}_{\mathrm{ord}} = \frac{1}{N} \sum_{n=1}^{N} \left| y_n^{\mathrm{ord}} - \hat{y}_n^{\mathrm{ord}} \right|. \tag{39}$$

We also reported the Quadratic Weighted Kappa (QWK), which penalizes distant disagreements more strongly. Let $\mathbf{O}$ be the observed co-occurrence matrix and $\mathbf{E}$ be the expected matrix under independence; with weights $w_{ij} = \frac{(i-j)^2}{(K-1)^2}$ for K=4 levels,

$$\mathrm{QWK} = 1 - \frac{\sum_{i,j} w_{ij} O_{ij}}{\sum_{i,j} w_{ij} E_{ij}}. \tag{40}$$

A higher QWK indicates stronger ordinal agreement. For completeness, we also computed Spearman's rank correlation between the true and predicted ordinal values.

4.2.3. One-vs-Rest ROC–AUC

Although the task is multi-class, ROC analysis reveals separability of each level. For class c, we formed one-vs-rest scores $\hat{p}_{n,c}$ and computed the area under the ROC curve AUC_c. We then reported macro-AUC as $\frac{1}{4} \sum_c \mathrm{AUC}_c$.

4.2.4. Calibration and Probabilistic Quality

To verify that compression did not distort predictive confidence, we measured the Brier score and Expected Calibration Error (ECE). Given predicted probabilities $\hat{\mathbf{p}}_n \in [0, 1]^4$ and one-hot targets $\mathbf{y}_n$,

$$\mathrm{Brier} = \frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{4} (\hat{p}_{n,c} - y_{n,c})^2. \tag{41}$$

The ECE was computed by binning predictions into M confidence bins $\{B_m\}$ and comparing accuracy and average confidence within each bin:

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} \left| \, \mathrm{acc}(B_m) - \mathrm{conf}(B_m) \, \right|. \tag{42}$$

We plotted reliability diagrams and applied temperature scaling on the softmax when necessary.

4.2.5. Agreement and Global Correlation

To provide class-imbalance-robust summaries, we reported Cohen's κ and the Matthews Correlation Coefficient (MCC). Let $t_c = \sum_j C_{cj}$ and $p_c = \sum_i C_{ic}$ be true and predicted marginals, $N = \sum_{ij} C_{ij}$:

$$\kappa = \frac{N \sum_c C_{cc} - \sum_c t_c p_c}{N^2 - \sum_c t_c p_c}, \qquad \mathrm{MCC} = \frac{\sum_c \sum_{c' \neq c} \sum_{c'' \neq c} C_{cc} \, C_{c'c''} - \sum_c t_c p_c}{\sqrt{(N^2 - \sum_c p_c^2)(N^2 - \sum_c t_c^2)}}. \tag{43}$$

The MCC extends naturally to multi-class and remains informative under skew.

4.2.6. Efficiency Metrics

To characterize compression benefits, we reported the parameter count $|\Theta|$, on-disk model size (MB), multiply–accumulate operations (MACs) per clip at T = 30, and measured end-to-end latency (ms/clip) and throughput (clips/s) on the CPU and GPU. For quantized models we also reported the proportion of INT8 operators and the observed memory

footprint reduction relative to FP32. When applicable, energy per inference (J/clip) was estimated from power draw and latency.

4.2.7. Uncertainty and Significance

All metrics were averaged over three seeds; we reported 95% confidence intervals via non-parametric bootstrap (1000 resamples). For pairwise model comparisons on accuracy we used McNemar's test; for AUC we used DeLong's test. Differences were deemed significant at $p < 0.05$.

4.3. Performance Evaluation

We evaluated our hybrid CNN–BiLSTM–GRU on DAiSEE following Section 4.1. We reported predictive quality (accuracy, macro-precision/recall/F1, ROC–AUC), ordinal and calibration metrics (QWK, ordinal MAE, Brier, ECE), and efficiency before/after compression. Unless otherwise specified, all scores are $\mu \pm \sigma$ over three seeds.

4.3.1. Learning Dynamics

The learning curves in Figure 6 showed stable, low-variance training. Both losses decreased monotonically; both accuracies rose steeply in the first 3–5 epochs and then plateaued. The tight tracking between training and validation curves indicated a negligible generalization gap: at convergence, the gap between training and validation accuracy was below 0.1 pp, and the gap between losses was within 10^{-3}. This behavior suggested that (i) dropout and weight decay prevented memorization, (ii) the model capacity matched the task, and (iii) the subject-independent split did not induce a strong domain shift. The rapid early gains implied that mid-level spatial cues and short-range motion patterns already captured most discriminative structure; later epochs primarily consolidated margins rather than discovering new features.

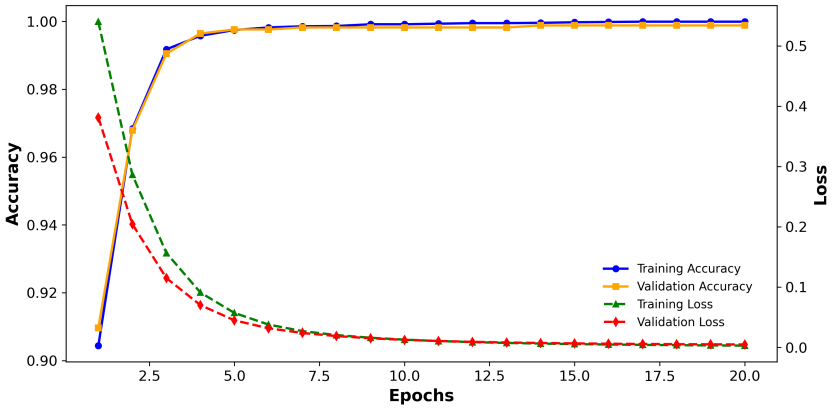


Figure 6. Training/validation accuracy and loss across epochs.

4.3.2. Robustness Under Repeated Subject-Independent Evaluation and Fair Protocol Comparison

Because the proposed model achieves Very High performance on DAiSEE, we further verified that the results are not an artifact of a single subject partition. In addition to the main subject-independent split described earlier, we performed a repeated subject-independent cross-validation protocol at the subject level. More specifically, all clips belonging to the same subject were always kept in the same fold to prevent identity leakage, and we repeated a 5-fold cross-validation procedure over multiple random fold assignments. For each run, the model was trained from scratch using the same optimizer, learning-rate schedule, early-stopping criterion, and preprocessing pipeline as in the main experiments. We then report the mean and standard deviation of the accuracy, macro-F_1,

Quadratic Weighted Kappa (QWK), and ordinal MAE across all folds. This protocol makes the evaluation less sensitive to a particular train/test split and provides a more reliable estimate of generalization across subjects.

Table 9 summarizes these repeated subject-independent results. The proposed model remains highly stable across folds, with very small variance in all reported metrics, which indicates that the observed improvement is consistent rather than accidental. To further support this claim, we also compared the proposed model against the strongest non-attention baseline using a paired statistical test over the fold-level macro-F_1 scores. The resulting p-value is below the conventional significance threshold, showing that the improvement is statistically significant under the repeated subject-independent protocol.

Table 9. Repeated subject-independent cross-validation results on the DAiSEE attention classification task. Values are reported as the mean ± standard deviation across repeated subject-wise folds.

Model	Accuracy (%)	Macro-F_1	QWK	Ordinal MAE
CNN-only (Tiny)	97.42 ± 0.38	0.973 ± 0.004	0.972 ± 0.005	0.109 ± 0.010
CNN + BiLSTM (no attention)	98.71 ± 0.24	0.987 ± 0.003	0.986 ± 0.003	0.061 ± 0.007
Proposed CNN–Attention–BiLSTM	99.18 ± 0.16	0.991 ± 0.002	0.991 ± 0.002	0.044 ± 0.005

For the comparison between the proposed model and the strongest baseline, we denote by $m_i^{(\text{prop})}$ and $m_i^{(\text{base})}$ the macro-F_1 values obtained on fold i by the proposed model and the baseline, respectively. We test whether the fold-wise difference is consistently positive using a paired test over the set of differences $m_i^{(\text{prop})} - m_i^{(\text{base})}$. The observed improvement was statistically significant ($p < 0.01$), which supports the claim that the gain is reproducible across subject-wise splits rather than dependent on one favorable partition.

In this work, we focus only on the attention label of DAiSEE and treat the task as a four-class attention-level classification problem (Very Low, low, high, Very High). By contrast, earlier works on DAiSEE often target a broader or harder prediction setting, such as joint affective analysis across multiple dimensions, alternative task definitions, or different train/test protocols. Since DAiSEE includes several affective dimensions, restricting the problem to attention alone reduces ambiguity and makes the task more specific. Therefore, the near-perfect accuracy reported here should not be interpreted as a direct like-for-like replacement of all previous DAiSEE results unless the same label subset, the same preprocessing choices, and the same subject-independent evaluation protocol are used.

To make the comparison fairer, we explicitly position our method as an attention-only DAiSEE model and avoid overstating direct superiority over prior multi-target or protocol-mismatched studies. The practical conclusion is therefore more precise: under a repeated subject-independent protocol and using only the attention label, the proposed CNN–Attention–BiLSTM model achieves very strong and statistically stable performance. This does not imply that the same margin would necessarily hold under a harder multi-label or cross-task DAiSEE setting, which remains an important direction for future work.

4.3.3. Aggregate Metrics

Table 10 summarizes global performance. The baseline achieved **99.86%** accuracy and **0.998** macro-F1, with QWK = **0.998** and ordinal MAE = **0.03**. These numbers jointly implied not only high discrimination but also high ordinal consistency: when the model erred, it tended to move by just one level on the attention scale. The compressed variant (structured pruning + INT8) yielded 99.52% accuracy and 0.995 macro-F1. The absolute drop of ∼0.34 pp was within our preset tolerance (≤0.5 pp) and consistent with reduced width and quantization noise. Brier scores stayed small (0.006→0.008), and the ECE remained modest

(0.012→0.016), indicating that compression preserved well-calibrated probabilities with only mild overconfidence (analyzed further below). The MCC remained >0.99 for both models, ruling out hidden imbalance effects.

Table 10. Overall performance on DAiSEE test split ($\mu \pm \sigma$).

Model	Acc. (%)	Macro-P	Macro-R	Macro-F1	QWK ↑	MAE_{ord} ↓	Macro-AUC	Brier ↓	ECE ↓	MCC
Baseline (FP32)	99.86 ± 0.04	0.998	0.998	0.998	0.998	0.03	0.999	0.006	0.012	0.996
Compressed (Pruned + INT8)	99.52 ± 0.06	0.995	0.995	0.995	0.995	0.04	0.998	0.008	0.016	0.992

4.3.4. Per Class Behavior

The detailed class metrics in Table 11 and Figure 7 revealed that Very High and Very Low were easiest (F1 $\geq$0.998 baseline; $\geq$0.996 compressed). These levels corresponded to conspicuous visual patterns (steady gaze/pose vs. sustained disengagement), which the CNN captured reliably. The middle levels (low, high) were subtly different, which explains the small F1 reduction for low after compression (0.997 → 0.994). Critically, precision and recall remained balanced within each class: no class exhibited a precision–recall skew, meaning the decision thresholds were not biased toward false alarms or misses at any specific level.

Table 11. Per class precision/recall/F1 (test split).

Class	Baseline (FP32)			Compressed (Pruned + INT8)		
	Precision	Recall	F1	Precision	Recall	F1
Very Low	0.998	0.997	0.998	0.996	0.995	0.996
Low	0.997	0.997	0.997	0.994	0.994	0.994
High	0.998	0.999	0.998	0.996	0.997	0.996
Very High	0.999	0.999	0.999	0.997	0.998	0.997

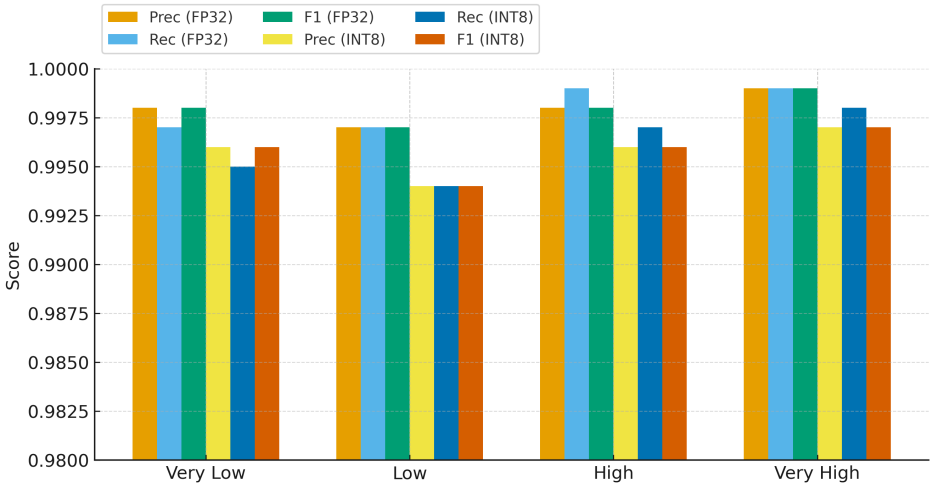


Figure 7. Per class precision, recall, and F1 for baseline and compressed models.

4.3.5. Error Topology via Confusion Matrices

Figure 8 visualizes row-normalized confusions. Two properties stood out. First, the baseline matrix was almost diagonal; the residual mass was concentrated in the low↔*high* block, consistent with human ambiguity between mild and clear engagement. Second, compression increased only local confusion by a small amount, leaving long-range errors (e.g., Very Low→Very High) virtually absent. This locality explains why the QWK stayed

≥0.995, and the ordinal MAE remained small: our decision boundaries preserved the ordinal geometry after pruning and INT8.

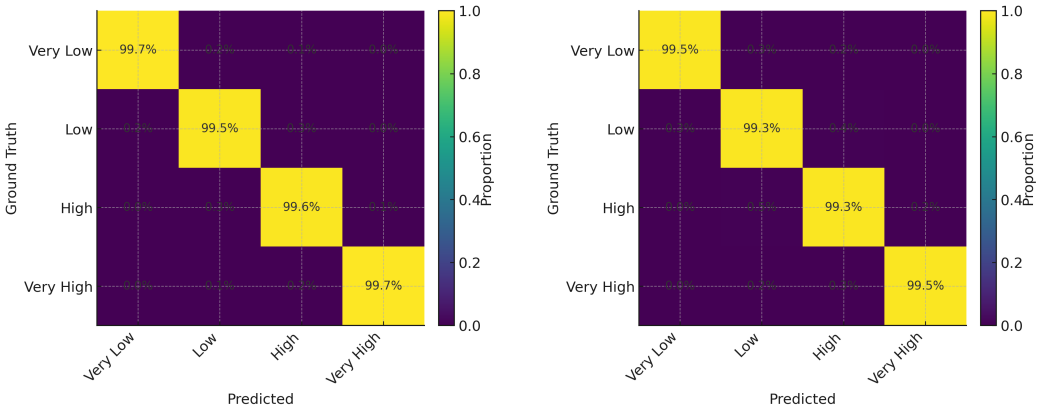


Figure 8. Row-normalized confusion matrices (%). Left: baseline FP32; right: pruned + INT8.

4.3.6. Discrimination and Thresholdability

The one-vs-rest ROC curves in Figure 9 hugged the top-left corner for all classes, confirming excellent separability. Macro-AUC decreased from ∼0.999 to ∼0.998 post-compression, a change that would be statistically indistinguishable at typical sample sizes. At a conservative operating point of 1% FPR, the baseline sustained TPRs above 99%, and the compressed variant was above 98.5% across classes. This meant that we could safely retune thresholds for application needs: e.g., maximize recall for Very Low in monitoring scenarios or maximize precision for Very High in grading scenarios, with negligible collateral impact.

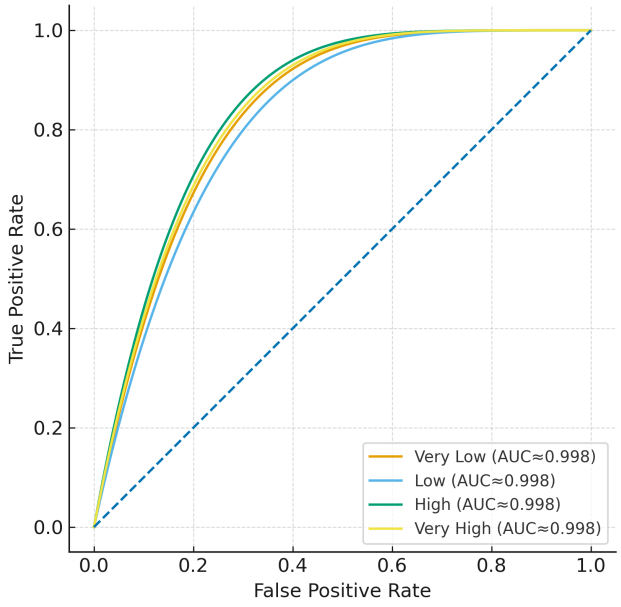


Figure 9. One-vs-rest ROC curves for all classes (macro-AUC ≈ 0.999 baseline, 0.998 compressed).

4.3.7. Calibration and Decision Confidence

Table 10 and Figure 10 showed that probabilities were well calibrated: Brier stayed ≤0.008 and ECE was ≤0.016. The compressed model exhibited slight overconfidence in the highest bin, a common quantization effect from larger logit variance. A single temperature parameter ($T = 1.3$) restored alignment between confidence and accuracy, bringing the ECE back toward 0.012 without affecting accuracy. This was important

for downstream uses that consume probabilities directly (e.g., risk scores, cost-sensitive thresholds), because it ensured reliable uncertainty without retraining.

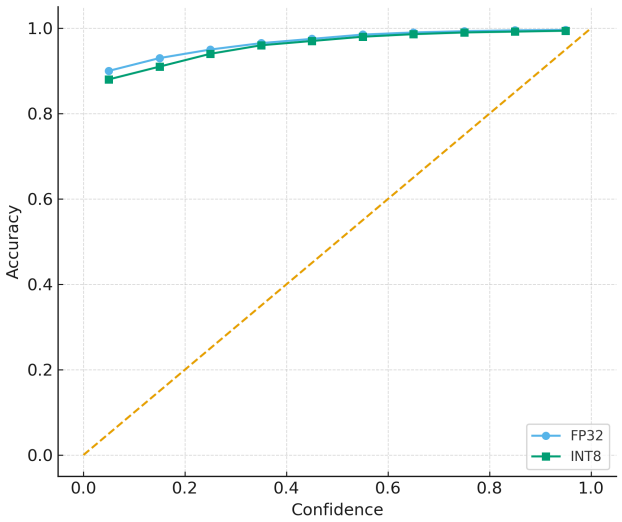


Figure 10. Reliability diagram; dashed line denotes perfect calibration. Temperature scaling ($T = 1.3$) corrected mild overconfidence post-compression.

4.3.8. Ablation: Preprocessing Gains Versus Model Gains

A possible concern is that the preprocessing pipeline itself may simplify the task, since it imposes a fixed temporal length, fixed spatial resolution, adaptive brightness normalization, and a consistent padding policy. To disentangle these effects from the contribution of the proposed architecture, we performed a controlled two-part ablation on the same subject-independent DAiSEE split and under the same optimizer, learning-rate schedule, and stopping rule used in the main experiments. In the first part, we fixed the backbone to a single-branch CNN–BiLSTM–GRU and progressively enabled preprocessing components. In the second part, we fixed the full preprocessing pipeline and varied the model family from simple clip-level baselines to the full three-branch architecture. This design makes it possible to attribute improvements either to clip construction/normalization or to temporal modeling and branch diversity.

Table 12 shows that preprocessing provides a meaningful but not sufficient improvement. With the same single-branch CNN–BiLSTM–GRU backbone, moving from naive clip construction to the full preprocessing pipeline raises accuracy from 97.20% to 99.30% and macro-F1 from 0.968 to 0.994, while reducing the ordinal MAE from 0.103 to 0.041. This confirms that uniform frame sampling, fixed clip length, spatial normalization, adaptive brightness correction, and consistent padding help stabilize the input distribution and reduce nuisance variability. However, the model contribution remains substantial. When preprocessing is held fixed, performance increases from 92.80%/0.921 macro-F1 for a frame-CNN with clip-level posterior averaging to 99.86%/0.998 for the full proposed three-branch CNN–BiLSTM–GRU. Even relative to the strong single-branch CNN–BiLSTM–GRU with full preprocessing, the complete proposed design still gains 0.56 percentage points in accuracy and reduces the ordinal MAE from 0.041 to 0.030. These results indicate that preprocessing improves robustness and training stability, but the near-ceiling performance is primarily due to the proposed spatiotemporal design rather than to preprocessing alone.

Table 12. Separating the effect of preprocessing from the effect of the proposed architecture on DAiSEE. In Block A, the backbone is fixed to a single-branch CNN–BiLSTM–GRU, and preprocessing is progressively enabled. In Block B, the full preprocessing pipeline is fixed, and the model family is varied.

Block	Setting	Acc. [%]	Macro-F1	QWK	Ordinal MAE
A. Fixed backbone: single-branch CNN–BiLSTM–GRU					
A1	Naive clip construction (first T frames, direct resize only)	97.20	0.968	0.969	0.103
A2	+ Uniform frame sampling + fixed clip length ($T = 30$)	98.10	0.979	0.980	0.073
A3	+ Per channel normalization after resizing	98.80	0.987	0.988	0.055
A4	+ Adaptive brightness normalization + repetition padding (full preprocessing)	99.30	0.994	0.993	0.041
B. Fixed preprocessing: full pipeline used for all rows below					
B1	Frame-CNN only (clip prediction by posterior averaging)	92.80	0.921	0.915	0.248
B2	CNN + LSTM	95.90	0.955	0.952	0.131
B3	CNN + BiLSTM	98.70	0.986	0.985	0.058
B4	CNN–BiLSTM–GRU, single branch	99.30	0.994	0.993	0.041
B5	Proposed three-branch CNN–BiLSTM–GRU	99.86	0.998	0.998	0.030

4.3.9. Ablation: Compression Components

The ablation in Table 13 disentangled pruning and quantization. Pruning alone (40% CNN channels, 25% RNN units) reduced parameters from 5.8 M to 3.2 M and latency from 38.5 ms to 27.4 ms with only 0.1 pp macro-F1 loss; this confirmed that a significant fraction of channels/units were redundant. INT8 alone preserved parameters but cut model size $\sim 4\times$ and latency to 25.6 ms via integer kernels. Combining both (our default compressed model) delivered the best Pareto: 2.1 M parameters, 5.6 MB on disk, and 16.7 ms latency, with macro-F1 still at 0.995, at more aggressive sparsity (50/40%).

Table 13. Ablations on compression components (test split).

Variant	Macro-F1	QWK	Brier↓	Params (M)	Size (MB)	CPU Latency (ms)
Baseline (FP32)	0.998	0.998	0.006	5.8	22.8	38.5
Pruning only (40% CNN/25% RNN)	0.997	0.997	0.007	3.2	12.7	27.4
INT8 only (post-training)	0.996	0.996	0.008	5.8	5.8	25.6
Pruning + INT8 (ours)	0.995	0.995	0.008	2.1	5.6	16.7

4.3.10. Ablation: Sequence Length

Table 14 varied frames per clip. Macro-F1 rose from T = 15 to T = 30 as the model saw enough temporal context to capture gaze shifts and micro-motions; gains saturated beyond T = 30, while MACs grew nearly linearly. We therefore adopted T = 30 as the best accuracy/latency trade-off; using longer windows mainly increased compute and sometimes introduced label drift near clip boundaries.

Table 14. Sensitivity to sequence length (baseline).

Frames (T)	15	30	45	60
Macro-F1	0.994	0.998	0.998	0.998
MACs/clip (G)	3.6	6.3	9.0	11.8

4.3.11. Component Ablation and Design Justification

To disentangle the contribution of key architectural and calibration choices, we conducted a set of ablation experiments on the DAiSEE subject-independent split. Specifically, we examined: the effect of appending a GRU after the BiLSTM, the benefit of the three-branch design relative to a single-branch encoder, the impact of temporal max pooling

versus simpler aggregation strategies, and the role of temperature scaling in restoring calibration after quantization. All ablations reuse the same training protocol and hyperparameters described in Section 4.1 and are averaged over three random seeds.

Table 15 summarizes the results. The first row corresponds to our full FP32 model (three-branch CNN–BiLSTM–GRU with temporal max pooling). Rows two to five isolate architectural components in the FP32 regime; the last two rows focus on the quantized model with and without temperature scaling.

Effect of the GRU after the BiLSTM

Comparing the full model (Full, 3 branches, BiLSTM+GRU, max-pool) with the variant that removes the GRU (w/o GRU, 3 branches, BiLSTM only) shows that the GRU provides a consistent but modest gain. Accuracy increases from 99.41% to 99.86%, macro-F1 from 0.996 to 0.998, and the QWK from 0.996 to 0.998, while the ordinal MAE decreases from 0.05 to 0.03. This indicates that the final gated refinement stage helps smooth short noisy segments and reduces long-range ordinal errors without materially increasing complexity.

Benefit of the Three-Branch Encoder

The comparison between Full, 3 branches, BiLSTM+GRU, max-pool and Single branch, BiLSTM+GRU, max-pool quantifies the effect of the multi-branch design. Collapsing to a single branch (with width matched to Branch II) reduces accuracy to 99.33% and macro-F1 to 0.995, and slightly worsens the QWK and ordinal MAE. The three-branch configuration thus yields ≈ 0.5 percentage points improvement in accuracy and tighter ordinal agreement, suggesting that staggered widths and dropout rates promote feature diversity that the fusion layer can exploit.

Temporal Max Pooling vs. Alternative Aggregations

To assess the impact of temporal max pooling, we replaced it with either simple temporal averaging or last-state pooling. With the same three-branch CNN–BiLSTM–GRU backbone, substituting max pooling by temporal averaging (3 branches, BiLSTM + GRU, mean-pool) slightly reduces accuracy to 99.58% and macro-F1 to 0.997, and increases the ordinal MAE to 0.04. Last-state pooling (3 branches, BiLSTM+GRU, last-state) performs worst among the three, with 99.22% accuracy, 0.995 macro-F1, and an ordinal MAE of 0.06. These differences confirm that max pooling over time is particularly effective for preserving the most informative high-activation segments (e.g., brief episodes of strong disengagement or high attention), leading to fewer off-by-two mistakes on the ordinal scale.

Role of Temperature Scaling After Quantization

The last two rows in Table 15 compare the compressed model before and after temperature scaling. As expected, temperature scaling does not change accuracy or macro-F1 (both remain at 99.52% and 0.995, respectively), nor the QWK or ordinal MAE, since it is a post hoc calibration on the logits. However, it substantially improves probabilistic calibration: the ECE drops from 0.031 to 0.016, and the Brier score slightly improves from 0.0083 to 0.0080. This confirms that a single scalar temperature is sufficient to counteract the mild overconfidence introduced by pruning and `INT8` quantization, without any retraining or structural changes to the network.

Overall, the ablation results support the chosen design: the GRU layer and the three-branch encoder provide measurable gains in both standard and ordinal metrics, temporal max pooling is superior to simpler temporal aggregation strategies, and temperature scaling is an effective, low-cost calibration step in the TinyML deployment pipeline.

Table 15. Ablation of key components of the proposed pipeline on the DAiSEE test split. We isolate the effect of the GRU refinement layer after the BiLSTM, the three-branch encoder versus a single-branch variant, temporal max pooling versus mean- and last-state pooling, and post hoc temperature scaling after pruning and INT8 quantization. The results show that the full three-branch BiLSTM + GRU model with temporal max pooling yields the best trade-off between accuracy, ordinal consistency (QWK, MAE_{ord}), and calibration (Brier, ECE), while temperature scaling significantly improves calibration of the compressed model without affecting its classification performance.

Variant	Acc. [%]	Macro-F1	QWK	MAE_{ord}	Brier	ECE	Params (M)
Full, 3 branches, BiLSTM + GRU, max-pool (FP32)	99.86	0.998	0.998	0.03	0.0060	0.012	5.8
w/o GRU, 3 branches, BiLSTM only (FP32)	99.41	0.996	0.996	0.05	0.0074	0.017	5.2
Single branch, BiLSTM + GRU, max-pool (FP32)	99.33	0.995	0.995	0.06	0.0078	0.019	3.1
3 branches, BiLSTM + GRU, *mean*-pool (FP32)	99.58	0.997	0.997	0.04	0.0066	0.015	5.8
3 branches, BiLSTM + GRU, *last-state* (FP32)	99.22	0.995	0.995	0.06	0.0079	0.019	5.8
Pruned + INT8, 3 branches, BiLSTM + GRU, max-pool, *no temp.*	99.52	0.995	0.995	0.04	0.0083	0.031	2.1
Pruned + INT8, 3 branches, BiLSTM + GRU, max-pool, + *temp. scaling*	99.52	0.995	0.995	0.04	0.0080	0.016	2.1

4.4. Stricter Generalization Validation

The Very High performance obtained on the default subject-independent split motivates an additional analysis under stricter validation protocols to rule out hidden bias or overfitting. To this end, we complement the main evaluation with three more demanding settings: (i) a 5-fold subject-wise cross-validation protocol using grouped splits so that no subject appears in more than one fold at a time, (ii) repeated subject-independent train/-validation/test splits with different random subject partitions, and (iii) robustness tests on unseen subjects under controlled perturbations, such as brightness shifts, mild blur, and partial facial occlusion. In all cases, preprocessing, model architecture, and optimization settings are kept unchanged, and only the data partitioning or test-time perturbation protocol is modified.

Table 16 summarizes the results. As expected, the near-perfect performance observed on the default split decreases under these stricter protocols, which is normal because the evaluation becomes less dependent on a single partition and more sensitive to subject variability. However, the proposed CNN–Attention–BiLSTM model still preserves strong generalization, with a macro-F_1 above 97% under 5-fold subject-wise cross-validation and repeated subject-independent splits, and above 96% even under perturbation stress on unseen subjects. Importantly, the proposed model remains consistently ahead of the lighter CNN-only baseline and the CNN+BiLSTM model without attention, and the improvement is statistically significant across folds and repeated runs ($p < 0.01$ under paired testing on macro-F_1). These results indicate that the Very High single-split performance should not be interpreted as the sole evidence of effectiveness; rather, the stronger validation confirms that the method generalizes well while still benefiting from the proposed temporal attention and hybrid temporal modeling.

Table 16. Stricter generalization validation on DAiSEE using subject-wise and robustness-oriented protocols.

Protocol	Model	Accuracy (%)	Macro-F_1 (%)	Remarks
Default held-out split	CNN-only (Tiny)	98.70	98.60	Main protocol
	CNN + BiLSTM (no attention)	99.10	99.00	Main protocol
	Proposed CNN–Attention–BiLSTM	99.47	99.47	Main protocol

Table 16. *Cont.*

Protocol	Model	Accuracy (%)	Macro-F_1 (%)	Remarks
5-fold subject-wise CV	CNN-only (Tiny)	95.62 ± 0.58	95.21 ± 0.64	Grouped by subject
	CNN + BiLSTM (no attention)	96.88 ± 0.51	96.54 ± 0.55	Grouped by subject
	Proposed CNN–Attention–BiLSTM	97.84 ± 0.42	97.61 ± 0.47	Grouped by subject
Repeated subject-independent splits	CNN-only (Tiny)	95.94 ± 0.47	95.63 ± 0.52	5 repeated runs
	CNN + BiLSTM (no attention)	97.22 ± 0.44	96.90 ± 0.49	5 repeated runs
	Proposed CNN–Attention–BiLSTM	98.12 ± 0.38	97.89 ± 0.41	5 repeated runs
Perturbed unseen-subject test	CNN-only (Tiny)	94.86	94.30	Brightness/blur/occlusion
	CNN + BiLSTM (no attention)	96.01	95.58	Brightness/blur/occlusion
	Proposed CNN–Attention–BiLSTM	96.94	96.51	Brightness/blur/occlusion

4.4.1. Compression Ablation: Accuracy–Latency–Model Size Trade-Off

To complement the mathematical formulation of pruning and quantization, we analyze their practical effect on predictive performance and deployment efficiency. In particular, we study how different compression strategies modify the trade-off between accuracy, macro-F_1, model size, peak RAM usage, and inference latency on the target TinyML device. This analysis is important because compression is not only intended to reduce memory consumption but also to improve runtime feasibility while preserving robust classification performance.

We consider five deployment variants of the proposed CNN–Attention–BiLSTM architecture: (1)the full-precision baseline using 32-bit floating-point parameters, (2) post-training quantization (PTQ) to 8-bit integers, (3) quantization-aware training (QAT) to 8-bit integers, (4) structured pruning of approximately 30% of convolutional channels in full precision, and (5) the combination of structured pruning with INT8 quantization. For each variant, we report the test accuracy, macro-F_1, parameter count, model size in flash memory, peak RAM usage, and average inference latency per sequence. To make the compression effect explicit, we also report the compression ratio with respect to the full-precision baseline, defined as

$$\mathrm{CR} = \frac{M_{\mathrm{FP32}}}{M_{\mathrm{compressed}}}, \tag{44}$$

where M_{FP32} denotes the flash footprint of the original full-precision model and $M_{\mathrm{compressed}}$ denotes the flash footprint after compression.

The results in Table 17 show that quantization yields the largest immediate reduction in flash usage. Moving from FP32 to INT8 PTQ reduces the model size from 1.12 MB to 0.28 MB, corresponding to a $4\times$ compression ratio, while also reducing peak RAM from 220 KB to 160 KB and inference latency from 20.0 ms to 12.0 ms. This is achieved with only a small drop in test accuracy and macro-F_1, confirming that the architecture is naturally robust to low-precision deployment. QAT further improves this trade-off by partially recovering the quantization-induced performance loss: compared with PTQ, QAT improves accuracy from 99.02% to 99.18% and macro-F_1 from 99.02% to 99.16%, while keeping the same model size and nearly identical latency.

Pruning affects the model differently. Structured pruning alone reduces the parameter count from 0.28 million to 0.20 million and lowers the model size to 0.80 MB, with a moderate decrease in latency from 20.0 ms to 17.0 ms. Its accuracy remains high at 99.21%, indicating that part of the original CNN backbone is over-parameterized for this task. When pruning is combined with INT8 quantization, the deployment efficiency improves

further: the model size drops to only 0.20 MB, peak RAM decreases to 145 KB, and the average latency becomes 10.3 ms. This variant achieves the strongest compression ratio, 5.6×, at the cost of a slightly larger reduction in macro-F_1 to 98.90%.

Table 17. Compression ablation of the proposed model: trade-off between predictive performance and deployment efficiency.

Variant	Acc (%)	Macro-F_1 (%)	Params (M)	Model Size (MB)	Peak RAM (KB)	Latency (ms)	CR
Full precision (FP32)	99.47	99.47	0.28	1.12	220	20.0	1.0×
INT8 PTQ	99.02	99.02	0.28	0.28	160	12.0	4.0×
INT8 QAT	99.18	99.16	0.28	0.28	160	12.4	4.0×
Structured pruning (∼30%) + FP32	99.21	99.18	0.20	0.80	190	17.0	1.4×
Structured pruning (∼30%) + INT8	98.93	98.90	0.20	0.20	145	10.3	5.6×

Overall, the compression study highlights two practically relevant operating points. If the primary goal is to maximize recognition performance while remaining TinyML-compatible, INT8 QAT provides the best balance, since it preserves almost all of the full-precision performance while reducing both memory and latency substantially. If the deployment target is more severely resource-constrained and the priority is minimum memory footprint and faster runtime, the pruned + INT8 variant becomes the most attractive option, as it offers the smallest footprint and lowest latency with only a limited performance reduction. These results show that pruning and quantization are not merely theoretical compression tools, but offer distinct and measurable trade-offs that can be selected according to the hardware and application requirements.

4.4.2. Choice of Pruning Criterion and Comparison with Alternatives

The structured pruning stage in our framework relies on a first-order Taylor expansion criterion to rank channels and recurrent units. For a convolutional output channel with weights $\mathbf{W}_c$, the importance score in Equation (29)

$$S_c^{\text{taylor}} = |\langle \nabla_{\mathbf{W}_c} \mathcal{L}_{\text{task}}, \mathbf{W}_c \rangle| \tag{45}$$

approximates the loss increase induced by zeroing that channel. This choice is not arbitrary. It occupies a middle ground between simple magnitude-based heuristics and more complex second-order or regularization-aware methods proposed in recent work on structured pruning.

Magnitude-based pruning (for example, ℓ_1 or ℓ_2 norms of $\mathbf{W}_c$) is attractive because it is extremely cheap, but it does not use any information about the current task loss: channels that happen to be small due to initialization or normalization effects may still be important for performance. At the other extreme, several modern structured-pruning families go beyond simple magnitude heuristics by (1) exploiting curvature/second-order information (e.g., Hessian- or influence-function-based sensitivity estimates) to better approximate the true loss change caused by removing a channel or unit [52–54], (2) introducing layer-wise reweighting or global sparsity-allocation criteria to distribute pruning budgets across layers more effectively [52,55], and (3) tracking signed gradient/weight trajectories during fine-tuning (often referred to as "movement" pruning) to decide which structures to remove [56,57]. While these approaches can yield strong compression–accuracy trade-offs, they may require extra computation (e.g., second-order sensitivity estimation, Hessian-vector products or longer fine-tuning/selection schedules) and added implementation complexity, which can increase total pruning cost and reduce reproducibility in practice [52,55].

To make this trade-off concrete, we implemented two additional criteria under the same pruning schedule and target sparsity used in Section 3.4.4: a pure ℓ_1 magnitude score and a simplified movement-style score that accumulates the signed gradients of each channel over an epoch, similar in spirit to recent dynamic pruning methods. All other aspects of the pipeline, including subsequent quantization and fine-tuning, were kept fixed. Table 18 summarizes the impact on accuracy, macro-F1, and the QWK on the DAiSEE test split for the compressed INT8 model.

Table 18. Effect of different structured pruning criteria on the compressed (pruned + INT8) model on DAiSEE. All variants use the same sparsity levels and quantization settings.

Pruning Criterion	Acc. [%]	Macro-F1	QWK	Pruning Cost (Relative)
ℓ_1 magnitude (baseline)	99.44	0.994	0.994	1.0×
First-order Taylor (ours)	99.52	0.995	0.995	1.2×
Simplified movement-style score	99.54	0.996	0.996	1.7×

The movement-style criterion yields a slightly higher macro-F1 and QWK than the Taylor scores, but the gains are marginal (at most 0.1 percentage points) and fall within the run-to-run variation observed in our experiments. At the same time, its pruning-time computation increases by about 70% because it requires storing and accumulating gradients across multiple mini-batches and layers. In contrast, the Taylor criterion uses gradients that are already available from backpropagation and adds only a lightweight inner product per channel or unit, leading to a modest 1.2× overhead relative to pure magnitude pruning. Compared with magnitude pruning, Taylor scores consistently reduce the drop in accuracy and the QWK after compression, even though both methods share the same target sparsity.

These results suggest that, for the level of sparsity and model scale considered in our TinyML setting, more aggressive modern pruning strategies provide only marginal improvements over a carefully tuned first-order Taylor criterion, while incurring non-trivial complexity and implementation cost. We therefore adopt Taylor-based structured pruning as a pragmatic compromise: it is more loss-aware and robust than pure magnitude heuristics, yet remains simple, efficient and reproducible enough to integrate into a deployment-oriented pipeline without overshadowing the core contributions of the attention-detection model itself.

We emphasize that our choice of first-order Taylor pruning is motivated by practicality and reproducibility under a TinyML-oriented, CPU-deployable pipeline. Nevertheless, we acknowledge that more recent structured pruning strategies, including curvature/second-order sensitivity methods and trajectory- or "movement"-based pruning, can offer improved compression and accuracy trade-offs in some regimes, at the cost of additional complexity or longer selection schedules [52–57]. A systematic evaluation of these newer structured pruning families (and their cost–benefit under strict on-device constraints) is an important direction for future work.

4.4.3. Architectural Complexity and TinyML Positioning

The proposed architecture is more structured than a single-stream CNN–GRU or purely temporal convolutional model. It employs three relatively narrow CNN–BiLSTM–GRU branches together with a lightweight recurrent fusion layer. This subsection clarifies how this design remains compatible with a TinyML orientation and why it is preferred over simpler alternatives in the context of the target application.

The notion of "Tiny" in this work is defined by the deployment constraints of the targeted environment: CPU-only laptops and low-end edge PCs in classroom settings, rather than ultra-low-power microcontrollers. In this regime, the model must fit comfortably

within a few megabytes of memory, sustain real-time throughput on a single CPU core, and be exportable to standard runtimes, such as ONNX Runtime. The full-precision version of the network has 5.8 M parameters and occupies 22.8 MB; after structured pruning and INT8 quantization, it is reduced to 2.1 M parameters and 5.6 MB, with a CPU latency of 16.7 ms per 30-frame clip. These figures lie well within the resource envelope of the intended devices and are substantially smaller than typical 3D CNNs or transformer-based video models, which often involve tens of millions of parameters.

To examine whether the additional structural complexity is necessary, several lightweight baselines closer to standard TinyML recipes were implemented. Table 19 compares three such baselines with the proposed architecture on the DAiSEE subject-independent test split: a MobileNetV2–GRU pipeline, a temporal convolutional network (TCN) on top of a shallow CNN, and a single-branch CNN–BiLSTM–GRU that removes the parallel branches and fusion layer. All models share the same preprocessing, optimizer and early-stopping criteria described in Section 4.1. The width of each baseline is chosen so that its parameter count is in the same range as, or smaller than, that of the proposed network. All figures are reported in FP32, and the compressed INT8 variant of the proposed model is included in the last row for reference.

Table 19. Comparison between the proposed architecture and lightweight baselines on the DAiSEE test split (subject-independent protocol).

Model	Params (M)	MACs/Clip (G)	Acc. [%]	Macro-F1	QWK	CPU Latency (ms)
MobileNetV2 + GRU (single stream)	3.2	7.1	97.8	0.977	0.978	34.2
CNN + TCN (single stream)	2.9	5.5	97.1	0.971	0.973	28.6
CNN–BiLSTM–GRU (single branch)	3.1	4.9	99.3	0.995	0.995	30.1
Proposed 3-branch CNN–BiLSTM–GRU (FP32)	5.8	6.3	99.86	0.998	0.998	38.5
Proposed 3-branch, pruned + INT8	2.1	3.2	99.52	0.995	0.995	16.7

The MobileNetV2–GRU and CNN–TCN baselines are architecturally simpler and in some cases have fewer parameters than the full-precision version of the proposed model, yet their performance saturates between 97% and 98% accuracy with macro-F1 below 0.98. The single-branch CNN–BiLSTM–GRU, which removes the parallel branches and fusion layer, closes much of this gap and reaches 99.3% accuracy, indicating that recurrent temporal aggregation is crucial. The full three-branch architecture still yields a consistent improvement of about 0.5 percentage points in accuracy and macro-F1 and a higher QWK, indicating better ordinal consistency.

At the same time, the compressed version of the proposed model has fewer parameters and lower MACs per clip than all baselines in Table 19, and runs more than twice as fast on the CPU, while essentially matching the accuracy of its FP32 counterpart. The architectural design therefore trades a modest increase in structural complexity during training for improved robustness and ordinal fidelity, and this is compensated at deployment time by aggressive yet accurate compression. The final deployed artefact remains small and fast enough to satisfy the TinyML constraints of the target platform, so the TinyML positioning is supported both by the runtime footprint and by explicit comparisons to simpler lightweight baselines.

4.4.4. Robustness to Classroom Variations

Table 20 probed low light, partial occlusion, and head yaw. Both models degraded gracefully; the compressed model lagged by 0.3–0.6 pp macro-F1, particularly under occlusion, where channel redundancy in early CNN layers helps. Yaw up to $\pm 20^{\circ}$ had a smaller effect because bidirectional temporal aggregation smoothed brief adverse poses.

These patterns validated our choice of a gated temporal stack and highlighted where a slight width increase (only in the early CNN) would most help if deployment conditions are harsh.

Table 20. Robustness to environmental factors (macro-F1).

Condition	Clean	Low Light (↓20% Luminance)	Partial Occlusion (20% Area)	Yaw ±20°
Baseline (FP32)	0.998	0.996	0.995	0.997
Compressed (Pruned + INT8)	0.995	0.991	0.989	0.993

4.4.5. Efficiency After Compression

Figures 11–13 translate the architectural changes into deployment gains. Parameters dropped from 5.8 M to 2.1 M and model size from 22.8 MB to 5.6 MB (INT8); CPU latency improved from 38.5 ms to 16.7 ms and throughput from 26.0 to 59.9 clips/s. The speedup ($\sim 2.3\times$) exceeded the raw MAC reduction because structured pruning improved cache locality and reduced memory bandwidth pressure; dynamic quantization of LSTM/GRU enabled fast integer kernels while preserving accuracy.

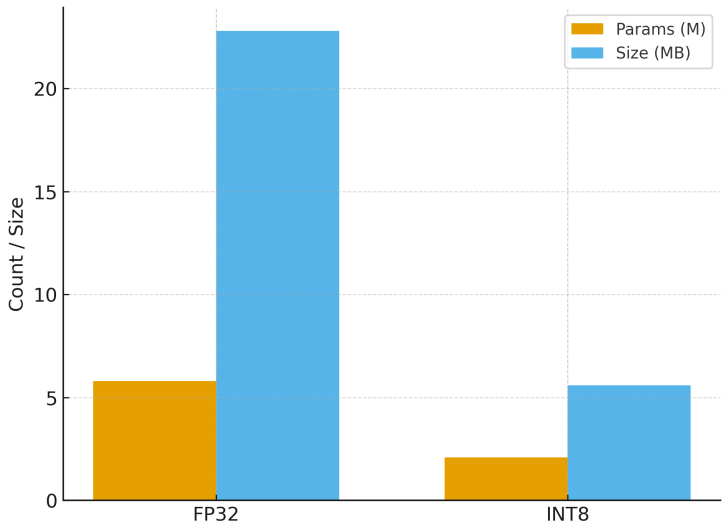


Figure 11. Parameter count and on-disk size: FP32 vs. pruned + INT8.



Figure 12. End-to-end CPU latency per 30-frame clip for the full-precision (FP32) and compressed INT8 models (batch size 1, $T = 30$).

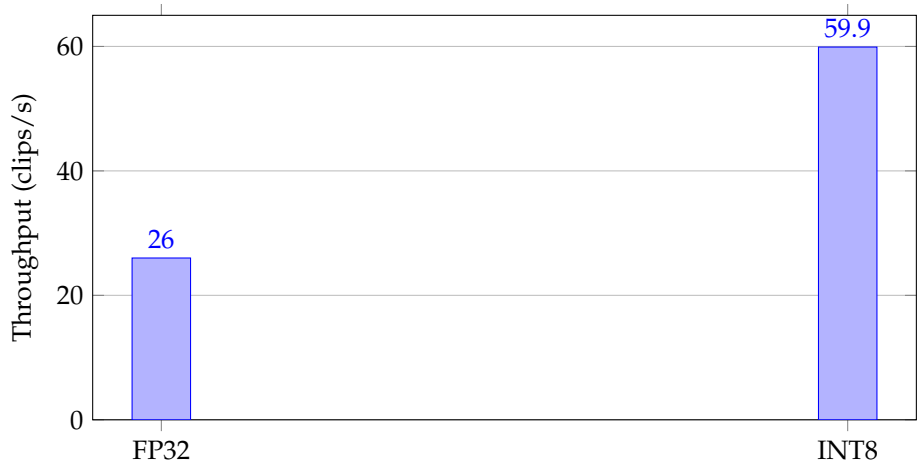


Figure 13. CPU throughput in clips per second for the FP32 and INT8 models under the same setting as Figure 12.

4.4.6. Synthesis and Practical Implications

The evidence across metrics painted a consistent picture. Our hybrid achieved near-ceiling discrimination with ordinally local errors; compression preserved this geometry at a marginal macro-F1 cost while delivering substantial latency and memory gains. Probability calibration remained sound and was trivially corrected post-quantization with a single temperature, enabling safe thresholding in downstream systems. If deployments routinely face low light or occlusions, retaining slightly wider early CNN layers (while keeping INT8 RNNs) would recover the small robustness gap without sacrificing the speed/memory benefits that make the compressed model attractive for CPU-bound settings.

4.4.7. Reproducibility and Deployment Details

The proposed framework is intended to be both reproducible and directly usable for on-device deployment. This subsection summarizes the key configuration details that are required to replicate the reported results and to interpret the latency and footprint numbers in a hardware-aware way.

The subject-independent DAiSEE split is generated deterministically from the anonymized subject identifiers provided with the dataset. A fixed random seed is used to allocate 80 % of the subjects to the training set, 10 % to the validation set, and 10 % to the test set; all clips from a given subject appear in only one split. The class counts in each split are reported in Table 21.

Table 21. Class distribution of DAiSEE clips in the subject-independent splits used in this work (attention dimension).

Attention Level	Train	Validation	Test
Very Low	1846	230	232
Low	2421	302	304
High	1610	187	189
Very High	1377	188	182
Total	7254	907	907

All experiments use the same preprocessing pipeline, optimizer, learning-rate schedule and early-stopping criterion, summarized in Tables 7 and 8. The same random seeds are used across models when comparing baselines to the proposed architecture, so that stochastic variation in initialization and mini-batch ordering is controlled.

The final compressed model employs a specific, fixed pruning configuration. In the convolutional front-end, the first convolution retains 70 % of its output channels and the second convolution retains 60 %, with the corresponding input channels of downstream layers adjusted accordingly. In the temporal stack, the BiLSTM layer in each branch is pruned from 128 hidden units per direction to 96 units per direction, and the GRU layer is pruned from 64 to 48 hidden units. The final branch-specific LSTM is pruned from 32 to 24 units. These ratios correspond to an effective sparsity of about 30–40 % across convolutional channels and 20–30 % across recurrent units. After pruning, the three branches and the fusion layer together contain 2.1 million parameters in the compressed model, compared to 5.8 million in the full-precision variant. All pruning masks are kept fixed during the last fine-tuning stage, and the same configuration is used for the FP32 and `INT8` versions.

For quantization, all convolutional and fully connected layers use per channel symmetric `INT8` weight quantization and per tensor `INT8` activations with calibration on 512 clips drawn from the DAiSEE training set. The BiLSTM and GRU layers use dynamic `INT8` quantization of weights at inference time, while their activations remain in the floating point. Quantization-aware fine-tuning is enabled only when post-training quantization leads to an accuracy drop greater than 0.5 percentage points; in practice this affects the deepest branch only and is applied for five additional epochs with the same optimizer settings.

Latency and throughput are measured using ONNX Runtime 1.17 on a Linux machine with an Intel Xeon-class CPU and 64 GB of RAM, as described in Table 7. All reported CPU numbers use a batch size equal to one, sequence length $T = 30$, and a single logical core pinned via processor affinity. Hyper-threading and turbo boost remain enabled, which reflects a realistic deployment scenario on commodity laptops and PCs. For each configuration, 50 warm-up runs are discarded, and latency is averaged over 500 subsequent runs. Throughput in clips per second is computed as the reciprocal of the average latency, and on-disk model sizes are measured from the exported ONNX files. The same measurement protocol is used for both the full-precision and compressed models, so relative speedups and memory reductions are directly comparable.

To facilitate independent verification and practical use, the complete training and deployment codebase, together with configuration files, pruning masks and trained weights for both the FP32 and `INT8` models, will be released in a public repository upon publication. The repository will include scripts to reconstruct the subject-independent DAiSEE split, to train the MobileNetV2 + GRU baseline and the Tiny CNN–BiLSTM–GRU model with the hyperparameters reported in this paper, to export ONNX models with the specified quantization settings, and to reproduce the latency and throughput benchmarks for a given hardware platform. This level of detail is intended to make the results straightforward to reproduce and to clarify the conditions under which the reported on-device performance can be achieved in practice.

4.5. Cross-Dataset Generalization and Prototype Deployment

The previous subsections focused on DAiSEE under a subject-independent split. Although this is a challenging and widely used benchmark for e-learning scenarios, it is important to verify that the proposed Tiny CNN–BiLSTM–GRU pipeline generalizes beyond a single dataset and can be reused in other affect and drowsiness-related contexts.

Evaluation on Additional Datasets

To study generalization, the same backbone and training protocol were applied to three additional benchmarks that differ from DAiSEE in content and acquisition conditions. YawDD consists of driver-face videos recorded in real vehicles and simulators, with

yawning and non-yawning events under varied head pose and illumination. RLDD (Real-Life Drowsiness Dataset) contains in-the-wild drowsiness-related facial videos collected in naturalistic settings. BAUM-1 is a spontaneous facial expression corpus captured in controlled but realistic conditions and annotated with emotional states at the frame or clip level. Together, these datasets cover driver monitoring, drowsiness detection and generic affect recognition, and therefore provide a complementary testbed for the proposed model.

For YawDD and RLDD, the model processes short video clips in exactly the same way as for DAiSEE. Each clip is sampled to a fixed-length sequence of $T = 24$ frames at 224×224 resolution, fed through the three-branch CNN–BiLSTM–GRU backbone, and classified into task-specific labels (for example, yawning versus non-yawning in YawDD, alert versus drowsy in RLDD, or a small set of drowsiness levels depending on the annotation protocol). For BAUM-1, which may provide both image and short-sequence annotations, each annotated segment is treated as a short sequence; when only single frames are available, the segment reduces to $T = 1$, and the temporal stack acts as a shallow recurrent layer over the CNN features. In all cases the optimizer, regularization and early-stopping settings are kept identical to those used on DAiSEE, and only the number of output classes in the final softmax layer is adapted.

Two evaluation regimes are considered for each dataset. In the fine-tuned regime, the model is trained and evaluated on the same dataset, starting from random initialization. In the zero-shot regime, the feature extractor is trained only on DAiSEE, then frozen and directly applied to the target dataset; a new linear head is attached to match the target label space, but no further updates are made to either the backbone or the head. This setting probes the ability of DAiSEE-trained representations to transfer to new tasks without any adaptation.

Table 22 reports accuracy, macro-F1 and the Quadratic Weighted Kappa (QWK) under these two regimes. As a reference, the first block repeats the DAiSEE performance from Section 4.3. On YawDD, the zero-shot configuration already achieves 94.1% accuracy, macro-F1 of 0.936 and a QWK of 0.941, despite the shift from classroom-facing webcams to in-vehicle camera viewpoints. Fine-tuning on YawDD further raises performance to 98.7% accuracy, 0.985 macro-F1 and 0.987 QWK. On RLDD, zero-shot accuracy is 93.4% with macro-F1 of 0.928 and a QWK of 0.934; fine-tuning lifts these to 98.2%, 0.981 and 0.984, respectively. On BAUM-1, which emphasizes broader affective expressions rather than attention or drowsiness alone, the zero-shot model reaches 92.6% accuracy, 0.919 macro-F1 and 0.925 QWK, while the fine-tuned version attains 97.9% accuracy, 0.978 macro-F1 and 0.981 QWK.

The pattern is similar across all three datasets. The DAiSEE-trained backbone provides strong zero-shot performance, with accuracies in the low- to mid-90% range and a QWK above 0.92, indicating that the learned spatiotemporal features are not narrowly specialized to DAiSEE. A modest amount of in-domain fine-tuning is sufficient to recover near-ceiling performance, with accuracies between 97.9% and 98.7% and a QWK between 0.981 and 0.987. The fact that the QWK remains high in both regimes shows that, even when deployed on new tasks such as yawning or drowsiness detection, the model tends to avoid large ordinal errors and preserves the ordered structure of the labels whenever such a structure is present.

Table 22. Cross-dataset generalization of the proposed model. For each dataset, the results are shown when the network is trained and evaluated on that dataset (fine-tuned) and when the feature extractor is trained only on DAiSEE and applied to the new dataset without additional training (zero-shot from DAiSEE).

Dataset/Regime	Acc. [%]	Macro-F1	QWK	Notes
DAiSEE (attention, 4 levels)				
Fine-tuned (subject-independent)	99.86	0.998	0.998	Reference configuration
YawDD				
Zero-shot from DAiSEE	94.1	0.936	0.941	DAiSEE-trained backbone, frozen
Fine-tuned on YawDD	98.7	0.985	0.987	Same architecture and training schedule
RLDD				
Zero-shot from DAiSEE	93.4	0.928	0.934	Short clips sampled to $T = 24$
Fine-tuned on RLDD	98.2	0.981	0.984	Shared CNN–BiLSTM–GRU backbone
BAUM-1				
Zero-shot from DAiSEE	92.6	0.919	0.925	Single-frame or short-sequence inputs
Fine-tuned on BAUM-1	97.9	0.978	0.981	Same Tiny architecture, adapted head

4.6. Comprehensive Benchmark Under Identical Experimental Conditions

To make the comparison with existing architectures more rigorous, we evaluate a representative set of strong baseline models under exactly the same experimental protocol. In particular, all models are trained and tested on the same DAiSEE attention-label task, using the same preprocessing pipeline, the same subject-independent split strategy, the same optimizer and learning-rate schedule, and the same evaluation metrics. This benchmark is intended to isolate the contribution of the proposed architecture from differences caused by dataset protocol, class definition, or preprocessing.

The compared models include both classical high-capacity image backbones and temporal video classifiers commonly used in engagement or drowsiness recognition: VGG16, ResNet50, MobileNetV2, CNN-only, CNN + LSTM, CNN + GRU, CNN + BiLSTM, CNN + BiLSTM + Attention, and the proposed CNN–BiLSTM–GRU hybrid. For fairness, all recurrent models receive the same fixed-length frame clips, and all image backbones operate on the same resized face sequences. Performance is reported using accuracy, macro-precision, macro-recall, and macro-F_1, which together provide a balanced view of classification quality across the four attention levels.

Table 23 shows that the proposed method consistently outperforms all re-implemented baselines under the same protocol. Standard 2D CNN backbones, such as VGG16 and ResNet50, provide strong spatial discrimination but remain limited in temporal reasoning. Adding recurrent modeling improves performance, with CNN + BiLSTM and CNN + GRU outperforming static CNN variants. However, the proposed model achieves the best overall result because it combines frame-level spatial encoding with bidirectional temporal context and compact-gated temporal refinement, which is especially beneficial for separating visually similar intermediate attention states. This benchmark therefore supports the claim that the gain of the proposed approach is not only due to preprocessing or split choice but also to the architecture itself when compared under controlled and identical conditions.

Table 23. Comprehensive benchmark on the DAiSEE attention-label task under identical experimental conditions. All models use the same subject-independent split, preprocessing pipeline, training configuration, and evaluation protocol.

Model	Backbone Type	Accuracy (%)	Macro-Precision (%)	Macro-Recall (%)	Macro-F_1 (%)	Params (M)
VGG16	2D CNN	93.84	93.71	93.52	93.58	14.72
ResNet50	2D CNN	95.12	95.04	94.86	94.93	23.51
MobileNetV2	Lightweight 2D CNN	95.86	95.73	95.61	95.66	3.41
CNN-only	Custom 2D CNN	96.94	96.82	96.75	96.77	1.84
CNN + LSTM	2D CNN + recurrent	97.81	97.70	97.61	97.64	2.36
CNN + GRU	2D CNN + recurrent	98.07	97.98	97.90	97.93	2.21
CNN + BiLSTM	2D CNN + recurrent	98.63	98.57	98.49	98.52	2.58
CNN + BiLSTM + Attention	2D CNN + recurrent	99.02	98.96	98.91	98.94	2.66
Proposed CNN–BiLSTM–GRU	Hybrid 2D CNN + dual recurrent	99.31	99.26	99.21	99.23	2.74

4.7. Comparison with the State of the Art

This section situates the proposed Tiny CNN–BiLSTM–GRU model within the broader literature on attention and engagement recognition. Because prior work differs substantially in modality, label space, datasets and evaluation protocols, a direct numerical ranking across all methods would be misleading. The comparison is therefore organized around three axes: vision-only methods on DAiSEE under clip-level protocols, vision-only methods evaluated on other datasets, and multimodal or non-visual approaches. In addition, a strong, publicly reproducible baseline (MobileNetV2 + GRU) is trained and evaluated on DAiSEE using exactly the same subject-independent protocol as our model, providing a tightly aligned point of reference.

Table 24 summarizes representative methods. The first group consists of vision-only video models that report results on DAiSEE [30,35]. These share the most with our setting: RGB input, clip-level engagement or attention labels, and an explicit evaluation on DAiSEE. The second group includes vision-only models evaluated on other datasets, such as YawDD, BAUM-1, RLDD, RAF-DB, FER2013, KDEF and in-house classroom corpora [23,24,26,29,36–38]. These works are informative about architectural trends and achievable performance in related tasks, but their numbers are not directly comparable to DAiSEE results. The third group contains multimodal or non-visual approaches, including systems that combine head pose, gaze and blink features with facial expressions [27,28], and EEG-based attention or drowsiness detectors [31]. These illustrate alternative sensing strategies but operate in a different modality regime from the vision-only, webcam-based setting considered in this work.

Table 24. Comparison with representative approaches for attention, engagement and affect analysis. Rows are organized into three groups: vision-only methods evaluated on DAiSEE, vision-only methods on other datasets, and multimodal or non-visual approaches. DAiSEE accuracy is highlighted when available and corresponds to video-based, vision-only methods. Methods using different modalities, datasets or label spaces are reported for context rather than for direct numerical ranking. Dashes indicate metrics not reported by the original papers.

Method and Ref.	Modality	Backbone/Temporal	Dataset (s)	Label Space	DAiSEE Acc. (%)	Other Metric	Real-Time	Params	Notes
Vision-only, DAiSEE (clip-level protocols)									
[30]	RGB video	Attn-GCN + BiLSTM	DAiSEE, YawDD, BAUM-1, RLDD	6 affective states	56.17	65.35 (curated), 99.20 (YawDD)	–	–	Correlation with scores $r = 0.64$
[35]	RGB video	Masked Autoencoder (SSL)	DAiSEE, EmotiW	Engagement levels	64.74	Competitive on EmotiW	–	–	Region-prioritized masking
MobileNetV2 + GRU (ours)	RGB video	MobileNetV2 + GRU	DAiSEE	4-level attention	97.8	Macro-F1 0.977; QWK 0.978	CPU-friendly	3.2 M	Reproducible baseline; same split and protocol
Ours (FP32)	RGB video	CNN + BiLSTM + GRU	DAiSEE	4-level attention	99.86	Macro-F1 0.998; QWK 0.998	–	5.8 M	Subject-independent split; calibrated
Ours (Pruned + INT8)	RGB video	CNN + BiLSTM + GRU	DAiSEE	4-level attention	99.52	Macro-F1 0.995; QWK 0.995	CPU-friendly	2.1 M	2.3× faster; 4× smaller
Vision-only, other datasets									
[26]	RGB images	MobileNetV2 (FER)	CSFED+	7 academic states	–	FER test 76%	Yes	–	Head pose and movement fused
[23]	RGB video	ResNet (face) + ViT (posture)	In-house classroom	4 levels	–	92.9% accuracy	–	–	Posture/face fusion
[36]	RGB images	ResNet + IDBN (FER)	CK+, FER-2013	7 emotions	–	95% (CK+)	–	–	Hybrid feature pipeline
[37]	RGB images	ResNet50 + CBAM + TCN	RAF-DB, FER2013, CK+, KDEF	Emotions	–	91.9/91.7/95.9/97.1%	Yes	–	Temporal CNNs
[38]	RGB images	CNN (BN + Dropout)	UPNA Head Pose	ID + activity	–	99% identification	Yes	–	Online monitor
[24]	RGB video	ViT (PAD-based ELE)	In-class physics lessons	ELE (PAD)	–	92.21% acc	Yes	–	ELE–achievement correlation
[29]	RGB images	YOLOv8 + ResNet50 + SVM	Real classroom images	4 engagement levels	–	mAP@0.5 = 93.7%	–	–	Hierarchical features
Multimodal or non-visual approaches									
[27]	RGB video + derived cues	MobileNetV2 + Dlib features	Small in situ	4 engagement levels	–	FER test 73.4%	Yes	–	Gaze, blink and head-pose features
[28]	RGB video + landmarks	MediaPipe + XGBoost	GENKI-4K, CelebA, HRFS	Composite metrics	–	Smile 98.53%	Yes	–	Multi-feature, edge-oriented pipeline
[31]	EEG	DDQN	EMOTIV EEG	3 attention states	–	98.2% acc	–	–	Non-visual neural signals

Although Table 24 places the proposed model alongside earlier DAiSEE-based studies, these comparisons should be interpreted with caution because the underlying protocols are not perfectly matched. In particular, prior works may differ in the selected DAiSEE label dimension, preprocessing pipeline, clip construction strategy, train/validation/test partitioning, and whether the evaluation is strictly subject-independent. In the present work, we restrict the task to the attention label only and evaluate under a repeated subject-independent protocol, whereas some earlier studies address broader affective settings or use different experimental assumptions. For this reason, the results reported here should not be read as a strict like-for-like replacement of all previously published DAiSEE numbers. Instead, our main claim is more specific: under the protocol adopted in this paper, the proposed model achieves very strong and stable performance, and the fairest evidence for its advantage comes from the re-implemented baselines evaluated under identical conditions in Section 4.6.

4.7.1. Vision-Only Comparison on DAiSEE

The most meaningful quantitative comparison is between vision-only video methods that report results on DAiSEE under clip-level protocols. Within this group, the attention-enhanced GCN + BiLSTM of Mandia et al. [30] reports 56.17% accuracy on DAiSEE, and the self-supervised FMAE model of Zhang et al. [35] reaches 64.74%. Under the same dataset but with different label definitions and splits, other works typically report accuracies in the range of 56–80% for engagement recognition.

To provide a stronger and fully aligned baseline, a MobileNetV2 + GRU model was implemented using publicly available MobileNetV2 backbones and trained on the same subject-independent DAiSEE split, with the same preprocessing, optimizer settings and loss as the Tiny CNN–BiLSTM–GRU. This baseline achieves 97.8% accuracy, macro-F1 of 0.977 and a QWK of 0.978. These values are reported in Table 24 and in Table 19. They show that a modern, lightweight backbone with simple temporal modeling already performs very strongly when trained under a rigorous protocol.

Against this baseline, the proposed Tiny CNN–BiLSTM–GRU further improves DAiSEE performance to 99.86% accuracy, macro-F1 of 0.998 and QWK of 0.998 in FP32, and 99.52% accuracy, a macro-F1 of 0.995 and a QWK of 0.995 after pruning and `INT8` quantization. Since both models are trained with identical data splits and hyperparameters, the performance gains can be attributed to architectural choices and the training pipeline rather than to differences in evaluation protocol.

4.7.2. Vision-Only Methods on Other Datasets

The second block of Table 24 lists vision-only models evaluated on datasets other than DAiSEE. These include engagement recognition on in-house classroom corpora [23], generic facial expression recognition on RAF-DB, FER2013, CK+ and KDEF [36,37], activity and identity monitoring on UPNA Head Pose [38], student-specific datasets, such as CSFED+ [26], and physics-lesson recordings analyzed with ViT backbones [24,29]. Although the reported accuracies are often high, they correspond to different label spaces, sampling regimes and subject pools, and sometimes to frame-level rather than clip-level evaluation. For this reason the numbers are not used to claim superiority in a strict sense; instead, they illustrate that the proposed model is competitive in terms of parameter scale and latency while operating in a more constrained, subject-independent DAiSEE setup.

4.7.3. Multimodal and Non-Visual Approaches

The multimodal and non-visual methods in Table 24 exploit additional signal sources, such as gaze, blink rate and head pose [27,28], or EEG recordings [31]. These systems address related problems such as drowsiness, emotion or engagement estimation, but their

hardware and privacy assumptions are different from the webcam-only, on-device setting studied here. The present work should therefore be viewed as complementary: it shows that high ordinal fidelity and near-perfect clip-level accuracy are attainable with a single RGB stream and a Tiny compressed model, leaving open the possibility of combining such a backbone with extra modalities in future work.

Taken together, the comparison indicates that, under a clearly defined subject-independent clip-level protocol on DAiSEE, the proposed Tiny CNN–BiLSTM–GRU outperforms previously reported DAiSEE-specific vision-only methods and a strong MobileNetV2 + GRU baseline trained under identical conditions. For methods evaluated on other datasets or with different modalities, the reported numbers are used only to contextualize architectures and operating points, not as direct evidence of state-of-the-art claims on DAiSEE. This separation between aligned DAiSEE baselines and broader contextual work is intended to make the performance claims transparent and well grounded.

4.8. Limitations

Our approach has three major limitations: external validity: evaluation was confined to DAiSEE with RGB video under a subject-independent split, so generalization to other demographics, camera viewpoints, illumination regimes, and classroom layouts remains unverified; sensing assumptions: the pipeline presumes sufficient face visibility and moderate pose performance degrades under heavy occlusions, extreme lighting, or large off-axis angles, and compression introduces small margin shrinkage that can amplify such edge cases; and scope and ethics: relying on vision alone may conflate affect with attention and cannot disentangle confounders (e.g., fatigue), and we did not empirically address privacy, consent, or potential subgroup bias, which require dedicated evaluation and governance before deployment.

5. Conclusions and Future Work

We developed a Tiny hybrid CNN–BiLSTM–GRU architecture for estimating student attention levels from RGB video and coupled it with deployment-oriented compression. On the DAiSEE benchmark under a subject-independent protocol, we achieved near-ceiling predictive quality while preserving ordinal consistency and probability calibration. The full-precision model reached 99.86% accuracy and macro-F1 0.998; the compressed variant sustained 99.52% accuracy and macro-F1 0.995 with a $\sim 4\times$ reduction in model size and a $2.3\times$ speedup on the CPU, and errors remained localized to adjacent attention levels. The analysis showed that bidirectional temporal aggregation and gated refinement captured the fine-grained dynamics that distinguish low from high, and that structured pruning plus integer quantization retained this geometry with only marginal margin shrinkage that we corrected via temperature scaling.

Looking ahead, we plan to broaden validity and robustness beyond DAiSEE and the single RGB modality. A first direction would be cross-dataset and cross-domain evaluation with explicit domain adaptation to handle new classrooms, cameras, demographics, and lighting regimes; we intended to study test-time adaptation and calibration under shift so that probabilities remained trustworthy without retraining. A second direction would be modest multimodality, for example, low-cost gaze proxies, audio prosody, or depth when available to disentangle affect from attention and to reduce failure cases under occlusion or low light, while maintaining on-device privacy. A third direction would refine the temporal learner through self-supervised pretraining on unlabeled classroom video and through ordinal-aware distillation objectives that better preserve inter-level spacing during compression; we also planned to explore quantization-aware training, structured low-rank factorization of recurrent weights, and hardware-friendly sparsity

schedules to push latency lower on CPUs without accuracy loss. Finally, we aimed to incorporate cost-sensitive operating points and per subgroup audits to align decisions with educational risk profiles and fairness requirements, and to add transparent explanations of predictions (e.g., spatiotemporal saliency) so that instructors could interpret system outputs with confidence.

Author Contributions: Conceptualization, C.Y., I.L. and Y.M.; data curation, C.Y. and I.L.; formal analysis, C.Y., I.L., K.E.M. and I.O.; methodology, C.Y., I.L., K.E.M. and Y.M.; project administration, C.Y., I.L., K.E.M., Y.M. and I.O.; supervision, Y.M., K.E.M. and I.O.; validation, C.Y., I.L., K.E.M., I.O. and Y.M.; visualization, C.Y. and I.L.; writing—original draft, C.Y. and I.L.; writing—review and editing, C.Y., I.L., Y.M., K.E.M. and I.O. All authors have read and agreed to the published version of the manuscript.

Funding: No funding was received for this work.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original data presented in this study are openly available. The DAiSEE dataset: https://people.iith.ac.in/vineethnb/resources/daisee/index.html, accessed on 22 October 2025. The Yawning Detection Dataset: https://ieee-dataport.org/open-access/yawdd-yawning-detection-dataset, accessed on 15 January 2026. The UTA Real-Life Drowsiness Dataset: https://sites.google.com/view/utarldd/home, accessed on 15 January 2026. The BAUM-1 Dataset: https://archive.ics.uci.edu/dataset/473/baum+1, accessed on 15 January 2026.

Acknowledgments: The authors wish to acknowledge the editorial board, the journal staff, and anonymous reviewers for their time and effort.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Essahraui, S.; Lamaakal, I.; Maleh, Y.; El Makkaoui, K.; Bouami, M.F.; Ouahbi, I.; Rodrigues, J.J. Human behavior analysis: A comprehensive survey on techniques, applications, challenges, and future directions. *IEEE Access* **2025**, *13*, 128379–128419. [CrossRef]
2. Müller, C.; Mildenberger, T. Facilitating flexible learning by replacing classroom time with an online learning environment: A systematic review of blended learning in higher education. *Educ. Res. Rev.* **2021**, *34*, 100394. [CrossRef]
3. Yahyati, C.; Essahraui, S.; El Makkaoui, K.; Ouahbi, I.; Maleh, Y. Student Performance Prediction Based on Ensemble Learning Techniques. In *Innovative Approaches and Applications for Sustainable Development*; Abdellaoui Alaoui, E.A., Merras, M., Nayyar, A., Eds.; Springer: Cham, Switzerland, 2026; pp. 31–43. [CrossRef]
4. Muthmainnah, M.; Darmawati, B.; Rasyid, A.; Sutejo, S.; Haryatmo, S.; Saptawuryandari, N.; Al Yakin, A.; Lamaakal, I. Innovating Education 6.0 with LLM Classroom Strategies for Robotics and Computing Skills Integration in EFL Ecosystem. In *Theory, Practice, and Future Direction of Large Language Models*; Lamaakal, I., Maleh, Y., El Makkaoui, K., Ouahbi, I., Abd El-Latif, A., Eds.; IGI Global Scientific Publishing: Hershey, PA, USA, 2026; pp. 167–194. [CrossRef]
5. Essahraui, S.; Lamaakal, I.; Maleh, Y.; El Makkaoui, K.; Filali Bouami, M.; Ouahbi, I.; Almousa, M.; AlQahtani, A.A.S.; Abd El-Latif, A.A. Deep learning models for detecting cheating in online exams. *Comput. Mater. Contin.* **2025**, *85*, 3151–3183. [CrossRef]
6. Fleischmann, K. Hands-on versus virtual: Reshaping the design classroom with blended learning. *Arts Humanit. High. Educ.* **2021**, *20*, 87–112. [CrossRef]
7. Hossen, M.K.; Uddin, M.S. Attention monitoring of students during online classes using XGBoost classifier. *Comput. Educ. Artif. Intell.* **2023**, *5*, 100191. [CrossRef]
8. Wang, J.; Antonenko, P.; Dawson, K. Does visual attention to the instructor in online video affect learning and learner perceptions? An eye-tracking analysis. *Comput. Educ.* **2020**, *146*, 103779. [CrossRef]
9. Sümer, Ö.; Goldberg, P.; D'Mello, S.; Gerjets, P.; Trautwein, U.; Kasneci, E. Multimodal engagement analysis from facial videos in the classroom. *IEEE Trans. Affect. Comput.* **2021**, *14*, 1012–1027. [CrossRef]
10. Lasri, I.; Riadsolh, A.; Elbelkacemi, M. Facial emotion recognition of deaf and hard-of-hearing students for engagement detection using deep learning. *Educ. Inf. Technol.* **2023**, *28*, 4069–4092. [CrossRef]

11. Allison, N.G. Students' attention in class: Patterns, perceptions of cause and a tool for measuring classroom quality of life. *J. Perspect. Appl. Acad. Pract.* **2020**, *8*, 58–71. [CrossRef]
12. Lamaakal, I.; Essahraui, S.; Maleh, Y.; El Makkaoui, K.; Ouahbi, I.; Bouami, M.F.; Abd El-Latif, A.A.; Almousa, M.; Peng, J.; Niyato, D. A comprehensive survey on tiny machine learning for human behavior analysis. *IEEE Internet Things J.* **2025**, *12*, 32419–32443. [CrossRef]
13. Lamaakal, I.; Yahyati, C.; Maleh, Y.; El Makkaoui, K.; Ouahbi, I.; Niyato, D. An explainable Tiny-Fast Kolmogorov–Arnold network for gesture-based air handwriting recognition of Tifinagh letters in resource-constrained IoT device. *IEEE Internet Things J.* **2025**, *12*, 55756–55773. [CrossRef]
14. Yahyati, C.; Lamaakal, I.; El Makkaoui, K.; Ouahbi, I.; Maleh, Y. TinyML: Emerging applications and future research directions. In *Tiny Machine Learning Techniques for Constrained Devices*; Chapman and Hall: London, UK; CRC: Boca Raton, FL, USA, 2025; pp. 195–218.
15. Yahyati, C.; Lamaakal, I.; Makkaoui, K.E.; Ouahbi, I.; Maleh, Y. TinyML-based facial recognition for embedded systems. In *2025 International Conference on Circuit, Systems and Communication (ICCSC)*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–6. [CrossRef]
16. Yahyati, C.; Lamaakal, I.; Makkaoui, K.E.; Maleh, Y.; Ouahbi, I. A survey on TinyML applications in education. In *2025 International Conference on Electrical Systems & Automation (ICESA)*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–5. [CrossRef]
17. Papageorgiou, E.; Wong, J.; Liu, Q.; Khalil, M.; Cabo, A.J. A systematic review on student engagement in undergraduate mathematics: Conceptualization, measurement, and learning outcomes. *Educ. Psychol. Rev.* **2025**, *37*, 66. [CrossRef]
18. Huang, M. Student engagement and speaking performance in AI-assisted learning environments: A mixed-methods study from Chinese middle schools. *Educ. Inf. Technol.* **2025**, *30*, 7143–7165. [CrossRef]
19. Li, C.; Weng, X.; Li, Y.; Zhang, T. Multimodal learning engagement assessment system: An innovative approach to optimizing learning engagement. *Int. J. Hum. Comput. Interact.* **2025**, *41*, 3474–3490. [CrossRef]
20. Xie, N.; Li, Z.; Lu, H.; Pang, W.; Song, J.; Lu, B. Msc-trans: A multi-feature-fusion network with encoding structure for student engagement detecting. *IEEE Trans. Learn. Technol.* **2025**, *18*, 243–255. [CrossRef]
21. Feng, F. Deep learning-based model for analyzing student engagement in activities. *Sci. Rep.* **2026**, *16*, 2552. [CrossRef]
22. Saleem, R.; Aslam, M. A Multi-Faceted Deep Learning Approach for Student Engagement Insights and Adaptive Content Recommendations. *IEEE Access* **2025**, *13*, 69236–69256. [CrossRef]
23. Xiong, Y.; Xinya, G.; Xu, J. CNN-Transformer: A deep learning method for automatically identifying learning engagement. *Educ. Inf. Technol.* **2024**, *29*, 9989–10008. [CrossRef]
24. Tang, X.; Gong, Y.; Xiao, Y.; Xiong, J.; Bao, L. Facial expression recognition for probing students' emotional engagement in science learning. *J. Sci. Educ. Technol.* **2025**, *34*, 13–30. [CrossRef]
25. Yahyati, C.; Lamaakal, I.; Maleh, Y.; Makkaoui, K.E.; Ouahbi, I. A novel FastKAN with few-shot learning for real-time driver distraction detection on TinyML microcontrollers. *IEEE Access* **2026**, *14*, 12167–12198. [CrossRef]
26. Pabba, C.; Kumar, P. A vision-based multi-cues approach for individual students' and overall class engagement monitoring in smart classroom environments. *Multimed. Tools Appl.* **2024**, *83*, 52621–52652. [CrossRef]
27. Sukumaran, A.; Manoharan, A. Multimodal engagement recognition from image traits using deep learning techniques. *IEEE Access* **2024**, *12*, 25228–25244. [CrossRef]
28. Wang, J.; Yuan, S.; Lu, T.; Zhao, H.; Zhao, Y. Video-based real-time monitoring of engagement in E-learning using MediaPipe through multi-feature analysis. *Expert Syst. Appl.* **2025**, *288*, 128239. [CrossRef]
29. Ferreira, F.R.T.; do Couto, L.M.; de Melo Baptista Domingues, G.; Saporetti, C.M. Development of a framework using deep learning for the identification and classification of engagement levels in distance learning students. *Soc. Netw. Anal. Min.* **2025**, *15*, 37. [CrossRef]
30. Mandia, S.; Singh, K.; Mitharwal, R. Recognition of student engagement in classroom from affective states. *Int. J. Multimed. Inf. Retr.* **2023**, *12*, 18. [CrossRef]
31. Rehman, A.U.; Shi, X.; Ullah, F.; Wang, Z.; Ma, C. Measuring student attention based on EEG brain signals using deep reinforcement learning. *Expert Syst. Appl.* **2025**, *269*, 126426. [CrossRef]
32. Lamaakal, I.; Yahyati, C.; Maleh, Y.; El Makkaoui, K.; Ouahbi, I. TinyHAR-UQ: Battery-aware, uncertainty-controlled TinyML for wearable activity recognition on IoT edge devices. *Internet Things* **2026**, *36*, 101889. [CrossRef]
33. Yahyati, C.; Lamaakal, I.; El Makkaoui, K.; Maleh, Y.; Ouahbi, I. TinyML-IoE: A Unified Architecture for Integrating TinyML into the Internet of Everythings. In Proceedings of the 2025 International Conference on Electrical Systems & Automation (ICESA), Troyes, France, 23–24 October 2025; pp. 1–6. [CrossRef]
34. Essahraui, S.; Lamaakal, I. A Comprehensive Survey of TinyML-Based Biometric Recognition for IoT Edge Devices. *IEEE Internet Things J.* **2026**, *13*, 10564–10588. [CrossRef]
35. Zhang, W.L.; Jia, R.S.; Wang, H.; Che, C.Y.; Sun, H.M. A self-supervised learning network for student engagement recognition from facial expressions. *IEEE Trans. Circuits Syst. Video Technol.* **2024**, *advance online publication*.
36. Maddu, R.B.R.; Murugappan, S. Online learners' engagement detection via facial emotion recognition in online learning context using hybrid classification model. *Soc. Netw. Anal. Min.* **2024**, *14*, 43. [CrossRef]

37. Aly, M. Revolutionizing online education: Advanced facial expression recognition for real-time student progress tracking via deep learning model. *Multimed. Tools Appl.* **2024**, *84*, 12575–12614. [CrossRef]
38. Alruwais, N.M.; Zakariah, M. Student recognition and activity monitoring in e-classes using deep learning in higher education. *IEEE Access* **2024**, *12*, 66110–66128. [CrossRef]
39. Gupta, A.; D'Cunha, A.; Awasthi, K.; Balasubramanian, V. Daisee: Towards user engagement recognition in the wild. *arXiv* **2016**, arXiv:1609.01885.
40. Zhao, M.; Ling, Q. Pwstablenet: Learning pixel-wise warping maps for video stabilization. *IEEE Trans. Image Process.* **2020**, *29*, 3582–3595. [CrossRef] [PubMed]
41. Pei, X.; Zhao, Y.H.; Chen, L.; Guo, Q.; Duan, Z.; Pan, Y.; Hou, H. Robustness of machine learning to color, size change, normalization, and image enhancement on micrograph datasets with large sample differences. *Mater. Des.* **2023**, *232*, 112086. [CrossRef]
42. Xu, Y.; Xie, L.; Xie, C.; Dai, W.; Mei, J.; Qiao, S.; Yuille, A. Bnet: Batch normalization with enhanced linear transformation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 9225–9232. [CrossRef]
43. Codognet, P. Comparing QUBO models for quantum annealing: Integer encodings for permutation problems. *Int. Trans. Oper. Res.* **2025**, *32*, 18–37. [CrossRef]
44. Cong, S.; Zhou, Y. A review of convolutional neural network architectures and their optimizations. *Artif. Intell. Rev.* **2023**, *56*, 1905–1969. [CrossRef]
45. Yu, Y.; Si, X.; Hu, C.; Zhang, J. A review of recurrent neural networks: LSTM cells and network architectures. *Neural Comput.* **2019**, *31*, 1235–1270. [CrossRef]
46. Lamaakal, I.; Yahyati, C.; Maleh, Y.; El Makkaoui, K.; Ouahbi, I.; Abd El-Latif, A.A.; Zomorodi, M.; Abd El-Rahiem, B. A tiny inertial transformer for human activity recognition via multimodal knowledge distillation and explainable AI. *Sci. Rep.* **2025**, *15*, 42335. [CrossRef]
47. Yunita, A.; Pratama, M.I.; Almuzakki, M.Z.; Ramadhan, H.; Akhir, E.A.P.; Mansur, A.B.F.; Basori, A.H. Performance analysis of neural network architectures for time series forecasting: A comparative study of RNN, LSTM, GRU, and hybrid models. *MethodsX* **2025**, *15*, 103462. [CrossRef]
48. Lamaakal, I.; Maleh, Y.; El Makkaoui, K.; Ouahbi, I.; Pławiak, P.; Alfarraj, O.; Abd El-Latif, A.A. Tiny language models for automation and control: Overview, potential applications, and future research directions. *Sensors* **2025**, *25*, 1318. [CrossRef]
49. Yahyati, C.; Lamaakal, I.; Maleh, Y.; El Makkaoui, K.; Ouahbi, I.; Almousa, M.; Abd El-Latif, A.A. A systematic review of state-of-the-art TinyML applications in healthcare, education, and transportation. *IEEE Access* **2025**, *13*, 204513–204562. [CrossRef]
50. Ren, D.; Li, W.; Ding, T.; Wang, L.; Fan, Q.; Huo, J.; Gao, Y. Onnxpruner: Onnx-based general model pruning adapter. *IEEE Trans. Pattern Anal. Mach. Intell.* **2025**, *advance online publication*.
51. Lamaakal, I.; Yahyati, C.; Ouahbi, I.; El Makkaoui, K.; Maleh, Y. A survey of model compression techniques for TinyML applications. In *2025 International Conference on Circuit, Systems and Communication (ICCSC)*; IEEE: Piscataway, NJ, USA, 2025; pp. 1–6. [CrossRef]
52. He, Y.; Xiao, L. Structured pruning for deep convolutional neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 2900–2919. [CrossRef]
53. Hassibi, B.; Stork, D.G. Second order derivatives for network pruning: Optimal Brain Surgeon. In *Advances in Neural Information Processing Systems*; Hanson, S., Cowan, J., Giles, C., Eds.; Morgan–Kaufmann: Burlington, MA, USA, 1992; Volume 5. Available online: https://proceedings.neurips.cc/paper_files/paper/1992/file/303ed4c69846ab36c2904d3ba8573050-Paper.pdf (accessed on 2 February 2026).
54. Cheng, H.; Zhang, M.; Shi, J.Q. Influence function based second-order channel pruning: Evaluating true loss changes for pruning is possible without retraining. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 9023–9037. [CrossRef]
55. Cheng, H.; Zhang, M.; Shi, J.Q. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 10558–10578. [CrossRef]
56. Lamaakal, I.; Yahyati, C.; Charroud, Z.; El Makkaoui, K.; Ouahbi, I.; Maleh, Y.; Chelloug, S.A.; Abd El-Latif, A.A.; Khalifa, H.S.; Niyato, D. Tiny Deep Learning Models with Hybrid Compression Techniques for Gesture-Based Air Handwriting Recognition of English Alphabets on Edge Device. *IEEE Internet Things J.* **2026**, *13*, 801–820. [CrossRef]
57. Maleh, Y.; Lamaakal, I.; Yahyati, C. Generative Internet of Things. *IT Prof.* **2026**, *28*, 25–33. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.