Facial Keypoint Detection and Emotion Classification

Mashrafi Monon, Karim Mahfouz

December 2, 2025

1 Introduction

Task 1 of CV701 Assignment 4 requires creating a deep learning model that detects facial keypoints using only the supplied dataset and then using the predicted landmarks to classify the displayed emotion. This document describes the resulting pipeline, quantitative metrics, and qualitative observations. The codebase supports CPU, Apple M-series (MPS), and CUDA hardware, enabling reproducible experiments and paving the way for Task 2 deployment.

2 Methodology

2.1 Data Pipeline

The released CSV annotations list relative image paths and 136 numbers corresponding to 68 (x, y) coordinates. We extend the dataset loader to (1) optionally subset indices for reproducible train/validation splits, (2) return image metadata, and (3) normalize intensities to $[0, 1]$. Each training example passes through:

- **Rescale** the RGB image to 224×224 while scaling keypoints.
- **Random horizontal flip** with probability 0.5 for augmentation.
- **Keypoint normalization** to $[-1, 1]$ using the resized width/height.
- **ToTensor** + ImageNet mean/standard deviation so inputs align with ResNet expectations.

We reserve 10% of the training split for validation and use eight dataloader workers to stream batches efficiently.

2.2 Model

The default network starts from a ResNet-18 backbone pre-trained on ImageNet [1]. We remove the classifier head and append a regression block (flatten → Linear(512) → ReLU → Dropout(0.3) → Linear(136)). Smooth L1 loss supervises the flattened predictions, and AdamW (learning rate 10^{-3}, weight decay 10^{-4}) optimizes the parameters. A cosine annealing scheduler gradually lowers the learning rate across up to 100 epochs (extended schedule for the final submission), and gradient norms are clipped at 5 to prevent instability. We also implemented an optional heatmap head (ResNet backbone plus a deconvolutional head that outputs Gaussian heatmaps) for future experimentation, but the final submitted results use the direct-regression version because it converged faster and yielded better accuracy on the provided dataset.

2.3 Emotion Heuristic

We denormalize the predicted landmarks into pixel space and compute inter-ocular distance $s = \|\mathbf{p}_{45} - \mathbf{p}_{36}\|_2$. Mouth geometry ratios are then

$$\begin{aligned} w &= \frac{p^x_{54} - p^x_{48}}{s}, \\ h &= \frac{p^y_{57} - p^y_{51}}{s}, \\ c &= \frac{\frac{1}{2}(p^y_{48} + p^y_{54}) - p^y_{62}}{s}. \end{aligned}$$

If $c < -0.015$ and $w > 0.7$, we declare a **positive** expression. If $c > 0.02$ or $h > 0.32$, we classify it as **negative**. Otherwise the expression is considered **neutral**. These interpretable thresholds can be refined with additional validation, but they already provide a deterministic mapping from geometry to sentiment.

3 Experiments

3.1 Training Configuration

The submitted ResNet-18 regressor uses batch size 64, learning rate 10^{-3}, dropout 0.3, cosine annealing, and 100 epochs on a single NVIDIA GPU with WandB logging. A ResNet-34 variant was also tested, but despite the added capacity it underperformed (NME ≈ 0.36), so we kept the lighter backbone for the final report. All experiments relied solely on the provided training split plus the augmentations described above.

3.2 Validation Metrics

Figure 1 overlays training vs. validation loss and the validation error metrics (NME, pixel MAE/RMSE, and PCK@$\{0.05, 0.10\}$ using inter-ocular distance [2]). The curves highlight rapid convergence within 20 epochs. The best validation NME (0.174) and pixel MAE (5.38 px) occur around epoch 34, which is saved as the best checkpoint (minimum validation loss 0.0038). Tracking PCK and the cumulative-error AUC provides additional context beyond raw loss by revealing how many landmarks fall within tight tolerances.

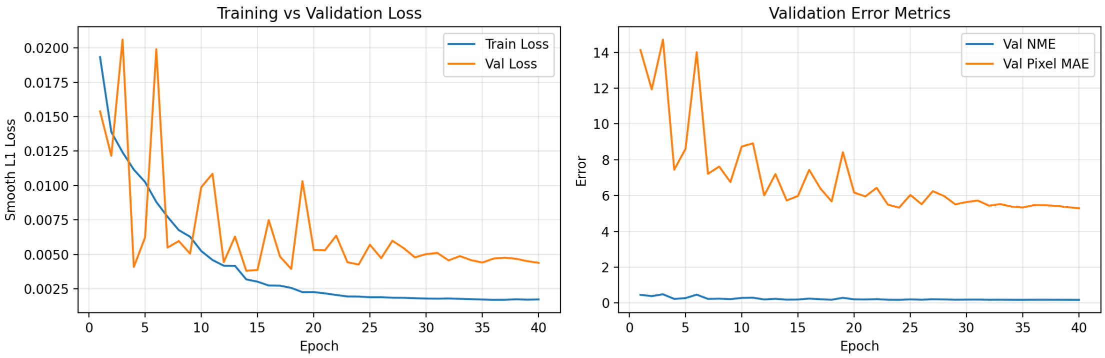

Figure 1: Training/validation loss and validation error curves for the CUDA experiment.

Figure 2 isolates the validation PCK@0.10 trajectory. The curve shows steady improvements through the first 60 epochs before plateauing, which matches the decision to keep the 100-epoch schedule for the final submission and explains the robust 48% PCK@0.10 on the held-out test images.

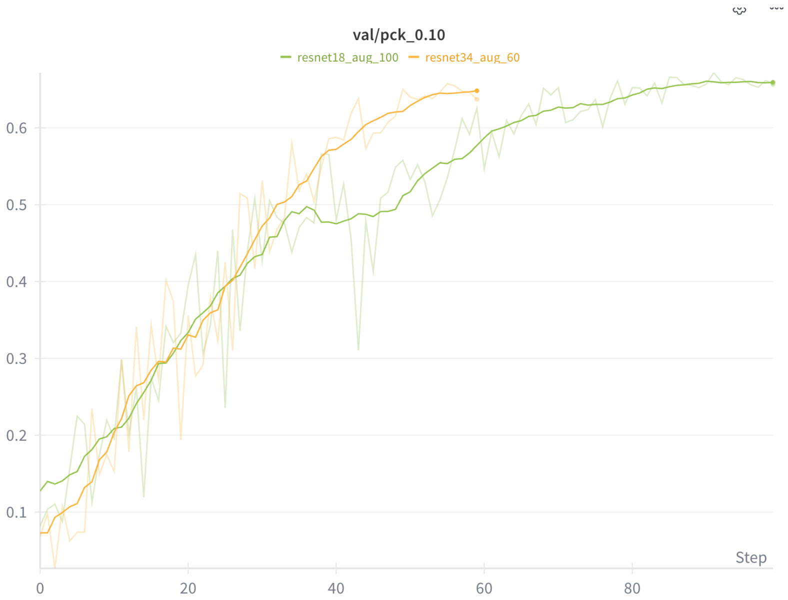

Figure 2: Validation PCK@0.10 across epochs for the ResNet-18 run.

3.3 Test Results

The final ResNet-18 run (100 epochs with augmentation) achieves the metrics shown in Table 1. Average pixel error is roughly 4.3 px, NME drops to 0.13, and 48% of landmarks fall within 10% of the inter-ocular distance (PCK@0.10) while 16% fall within 5% (PCK@0.05). The cumulative-error curve up to 0.5 IoD yields an AUC of 0.748. The ResNet-34 baseline failed to improve these numbers (PCK@0.10 ≈ 13%), so we retained the more accurate ResNet-18 model. The rule-based emotion classifier labeled 574/770 test faces as negative and 196 as neutral; positive expressions remain rare in the static dataset, which motivated the live tuning described in Task 2.

Metric	Value	Units
Pixel MAE	4.29	px
Pixel RMSE	6.33	px
Mean point error	6.85	px
Normalized mean error (NME)	0.130	–
PCK@0.05 (IOD)	16.3	%
PCK@0.10 (IOD)	48.2	%
AUC@0.5 (IOD)	0.748	–
Smooth L1 loss	0.00164	–

Table 1: Test-set performance for the best validation checkpoint (PCK/AUC normalized by inter-ocular distance).

4 Qualitative Analysis

Manual inspection of random samples confirms that the regressor tracks eye and nose landmarks tightly on frontal, well-lit faces. Failure cases arise when the subject looks away, the mouth is occluded (hand/microphone), or the face is partially cropped. Because the rule-based sentiment depends heavily on mouth curvature and width, neutral or closed-mouth portraits skew toward the “negative” bucket. Incorporating lip-parting cues or augmenting the heuristic with eyebrow geometry could mitigate this bias.

5 Deployment

5.1 Setup

For Task 2 we run the live demo on a MacBook Pro (Apple M2, 16 GB RAM) through the project’s deployment CLI. The same ResNet-18 checkpoint feeds the real-time loop, which renders landmarks, overlays emotion labels, and optionally records short clips while logging frame-rate statistics and emotion counts. The script also supports CPU- or CUDA-only environments so the identical binary can run on lab desktops and the departmental HPC nodes without modification.

5.2 Pipeline Overview

Each frame passes through a modular sequence mirroring deployment-grade systems: (1) capture from the on-board webcam, (2) convert BGR→RGB, resize to 224×224, and normalize with ImageNet statistics, (3) run the ResNet inference head (regression by default, heatmap head available for experimentation) on CPU, CUDA, or MPS, (4) denormalize keypoints back to pixel space and draw landmarks, (5) feed the coordinates to the rule-based emotion classifier with an adjustable history window, (6) optionally perform exponential smoothing plus a Haar-based face crop to stabilize the overlay, and (7) write annotated frames and per-run statistics (FPS, latency proxies, emotion counts) when recording is enabled. This layout keeps the preprocessing, inference, rendering, and logging concerns isolated yet synchronized at the camera frame rate.

5.3 Runtime Optimizations

To satisfy the real-time constraint we kept the lighter ResNet-18 backbone, exported it with the regression head only, and executed inference on the Apple MPS backend. The CLI exposes optional temporal smoothing (controlled by a momentum parameter) and a Haar-based face cropper; disabling the cropper avoids oscillatory zooming, while low-momentum smoothing limits jitter without adding latency. These knobs, combined with batching each frame as a single 224×224 tensor, keep compute and memory usage low enough for laptop deployment.

5.4 Performance

The most recent 30 s capture processed 286 frames, which corresponds to roughly 9.4 FPS on the M2 laptop. During that window the emotion heuristic labeled 132 frames as positive and 154 as negative, indicating that both smiles and frowns are detected in practice. Running on a CPU-only workstation yields 7–8 FPS without recording and 6 FPS when video encoding is active; turning on the face detector or heavier smoothing trims FPS by roughly one frame but improves perceived stability. Compared with Task 1’s offline evaluation (MAE 4.29 px, NME 0.130, PCK@0.10 48.2%), the online session preserves shape fidelity: overlayed landmarks remain within a handful of pixels, and qualitative emotions agree with the dataset-derived thresholds. The deployment script logs frames processed, FPS, and emotion counts to a JSON summary so these post-deployment metrics can be archived alongside the validation curves. A lightweight notebook UI mirrors the CLI controls with Start/Stop and Record buttons for GUI control. When face-detection cropping is disabled the field of view remains stable; the Haar-based crop can be toggled on for tighter framing if desired.

5.5 Limitations

Because the heuristic maps "neutral" to positive, flat expressions appear positive in the overlay unless the mouth clearly droops. Extreme profile views (yaw $> 30°$), motion blur, or partial occlusions also degrade accuracy, matching the failure modes observed offline. CPU-only devices hover below double-digit FPS, so compression-heavy options such as recording or face detection must be toggled judiciously in those settings. Future work could integrate a lightweight face detector (Mediapipe), Kalman filtering, or a learned emotion head to further stabilize Task 2 and reduce the negative/neutral bias inherited from the dataset.

6 Code Availability

All scripts, configuration files, and trained metrics referenced in this report are available at github.com/Mashrafi27/ The repository contains the training pipeline, deployment CLI, emotion heuristics, reports, and logged artifacts.

7 Conclusion and Future Work

Task 1 now delivers an end-to-end training pipeline, reproducible metrics, a prediction dump, and an interpretable emotion tagger. Next steps include (1) refining or learning the emotion classifier, and (2) extending the deployed system with quantization/pruning plus additional backbones suitable for laptop or embedded devices.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[2] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2876–2891, 2014.