Online Resource 1: SkinGuardian – Additional Methods, System Design, and Extended Analyses

Aayush Kumar[1*] and Fahad Salim Dalwai[1]

[1]Department of Computer Science, University of Illinois Chicago, Chicago, IL, USA.

*Corresponding author(s). E-mail(s): akuma102@uic.edu;
Contributing authors: fdalw@uic.edu;

S1 Split discipline and leakage prevention

We follow strict evaluation hygiene to prevent train–test contamination. (i) We use stratified train/validation/test splits for ISIC 2019, and we apply data augmentation *only* to training images. (ii) For Fitzpatrick17k, we use the author-provided partitions: train/validation may contribute to training, while the *test* split is held out strictly for subgroup evaluation and is never used for optimization, early stopping, threshold selection, or model selection. (iii) Decision thresholds, calibration (temperature scaling), and operating points are determined on validation data only and then frozen for all test and external evaluations. (iv) External validation on SIIM-ISIC 2020 is performed with no retuning. We note that patient identifiers are not available in these public datasets, so patient-level splitting cannot be enforced; this limitation is explicitly acknowledged.

S2 System Design

SkinGuardian is designed for fully on-device operation, minimizing privacy risks and ensuring compatibility with Snapdragon-powered Windows devices. We describe the main design choices and data flow from image capture to explainable output.

S2.1 Overall Architecture

S2.1.1 Frontend (UI)

A lightweight Flask interface with Tailwind CSS serves as the user interface. This design provides:

- **Ease of use**: packaged as a standalone desktop app with no external dependencies.
- **Responsive layout**: adjusts to various screen sizes for consistent display.
- **Extensibility**: modular templates for future features (e.g., patient info, feedback collection).

S2.1.2 Backend (Inference Engine)

Inference uses ONNX Runtime for efficiency and portability:

- **ONNX conversion**: the PyTorch-trained model is exported to ONNX for cross-platform deployment.
- **Minimal inference dependencies**: inference requires only ONNX Runtime, NumPy, and Pillow; optional explanation generation (e.g., Grad-CAM) uses the PyTorch checkpoint locally.
- **Offline deployment**: the model runs locally without server calls, eliminating data upload risks.

S2.2 Model Implementation and Deployment

S2.2.1 BEiT Fine-tuning

Starting from the pretrained `beit-base-patch16-224` model, we replace the classification head with a binary output layer, enable gradient checkpointing for memory efficiency, and fine-tune on *ISIC 2019* plus the *Fitzpatrick17k train/validation* partitions (the Fitzpatrick17k *test* split is held out strictly for evaluation). For DP-SGD variants, privacy accounting uses the ISIC 2019 training split size ($N \approx 20{,}000$) with $\delta = 1/N$; DP results are reported on ISIC 2019. To enhance trustworthiness, we apply DP-SGD (for DP variants) and adversarial training (FGSM/PGD-10) during fine-tuning.

S2.2.2 ONNX Runtime Deployment

The best model checkpoint is exported to ONNX for efficient edge inference. ONNX Runtime was selected for its cross-platform support, small footprint, and built-in graph optimizations. We leverage features like dynamic quantization (converting weights to INT8) to accelerate inference. The final ONNX model is packaged with the app, allowing it to run on devices with CPU-only capabilities or ARM processors with no code changes.

S2.3 Inference Workflow

1. **Image Capture & Preprocessing:** The user uploads or captures a skin lesion image, which is resized to 224×224, normalized to ImageNet pixel statistics, and converted into a tensor.
2. **ONNX Model Inference:** The preprocessed tensor is passed through the ONNX model, producing a malignancy probability score and a binary prediction (benign vs. malignant).
3. **Optional sensitivity analysis (occlusion):** An occlusion sensitivity analysis optionally runs in the background, highlighting image regions most critical to the model's decision by sliding an occluding patch and observing output changes.
4. **Explainability Generation:** Grad-CAM heatmaps (computed locally via the PyTorch checkpoint) and LIME superpixel explanations are generated to illustrate the model's reasoning (regions influencing the benign/malignant prediction).
5. **UI Rendering:** The prediction, confidence score, and visual explanation overlays are displayed to the user. All computations occur on-device, and no data is transmitted externally.

This design emphasizes *privacy* (no data leaves the device), *portability* (few dependencies and hardware-agnostic model), and *explainability* (multi-faceted model interpretation) while maintaining competitive performance for real-time use.

S3 Training, Privacy, Fairness, and Robustness Details

S3.1 Training Configuration and Fine-Tuning

We fine-tune the BEiT-Base-Patch16-224 model (pretrained on ImageNet-22k) for 50 epochs using the AdamW optimizer (initial learning rate 3×10^{-5}, weight decay 1×10^{-4}). Training is done in mixed precision (FP16) to speed up computation. We employ early stopping on the validation loss to mitigate overfitting.

S3.2 Adversarial and Fairness-aware Training

To improve robustness, we perform adversarial training on 50% of mini-batches. Epochs 1–5 use FGSM with ϵ linearly ramped over $\{0.01, 0.02, 0.03\}$; epochs 6–50 use L_∞ PGD-10 (10 steps, step size α=0.007, ϵ=0.03) with random initialization $\mathcal{U}[-\epsilon, \epsilon]$. When combined with DP-SGD, the adversarial example is generated using current (noisy) gradients but DP noise is not backpropagated through the attack generation (standard practice). For fairness, we compare reweighting and adversarial debiasing; the latter (gradient reversal with a skin-tone discriminator [2]) performed best and is used in the final model.

S3.3 Differentially Private Training

For privacy preservation, we train variants of the model with DP-SGD [1]. We set a per-sample gradient norm clip $C = 1.0$ and add Gaussian noise to gradients with

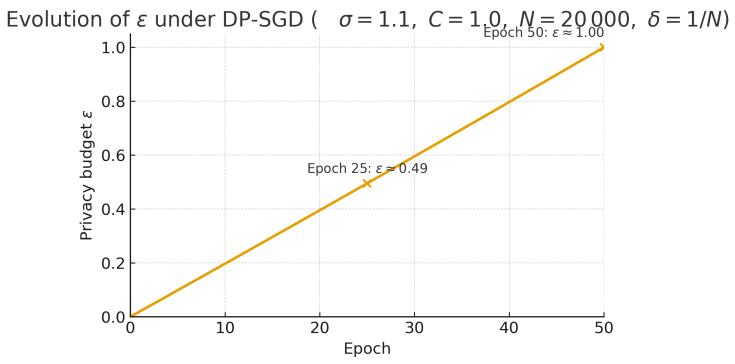

Fig. S1 Evolution of the privacy budget ϵ per epoch for DP-SGD (σ=1.1, C=1.0, N=20,000, δ=1/N). Privacy loss accumulates sublinearly over 50 epochs.

Table S1 Differential privacy hyperparameters for DP-SGD. N: training set size; q: sampling rate; B: batch size; C: gradient clipping norm; σ: noise multiplier; E: epochs; Accountant: privacy accounting method.

N	q	B	C	σ	E	**Accountant**
20,000	0.0016	32	1.0	1.1	50	Rényi DP

multiplier σ such that the target (ϵ, δ) is achieved after all training epochs (using a Rényi differential privacy accountant). In practice, for a training set of size $N \approx$ 20,000 (sampling rate $q \approx 0.0016$, batch size $B = 32$) over $E = 50$ epochs, $\sigma = 1.1$ yields $\epsilon \approx 1.0$ at $\delta = 1/N$. For a stronger privacy level $\epsilon = 0.1$, we increase σ to ≈ 4.0. We track ϵ as training progresses; it grows sublinearly with epochs (e.g., $\epsilon \approx 0.5$ after 25 epochs for $\sigma = 1.1$). Figure S1 illustrates the growth of the privacy loss per epoch for a representative run. We found that modest noise levels (yielding $\epsilon \geq 1$) have minimal impact on accuracy, while very small ϵ (high noise) significantly degrades performance. We implement DP-SGD using Opacus with the Rényi DP (RDP) accountant. We set $\delta = 1/N = 5.0 \times 10^{-5}$ (with N = 20,000). The per-epoch privacy curve $\epsilon(e)$ is tracked by the RDP accountant; an example run is shown in Fig. S1.

S3.3.1 Why differential privacy is included.

Although our benchmarks use public datasets, we include DP-SGD to demonstrate that the same training pipeline can be applied to *institutional or patient-derived clinical images* while providing a formal privacy guarantee against membership inference and related leakage from model parameters. This is complementary to on-device inference: keeping inference local reduces exposure during use, while DP-SGD reduces risk during model development when training data may be sensitive.

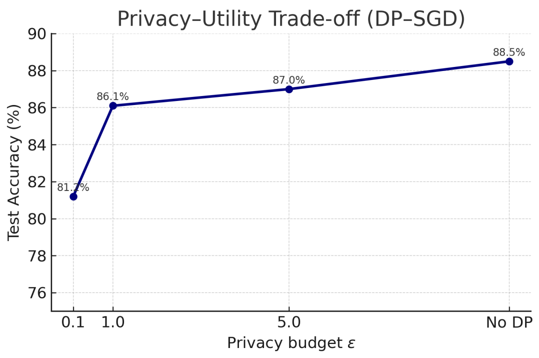

Fig. S2 Privacy–utility trade-off: test accuracy vs. privacy budget ϵ. Accuracy remains $> 85\%$ for $\epsilon \geq 1$ and only declines substantially at the strictest privacy level ($\epsilon = 0.1$), indicating that strong privacy can be attained with moderate cost to utility.

Table S2 Ablation of differential privacy (DP) and adversarial training (AT). Robust accuracy is against PGD-10 ($\epsilon = 8/255$).

Setting	DP ϵ	**Clean Acc**	**Robust Acc**
SkinGuardian-Clean (no DP, no AT)	–	88.5%	66.7%
SkinGuardian-DP (DP only)	1.0	86.1%	64.0%
SkinGuardian-Robust (AT only)	–	87.1%	74.8%
SkinGuardian-DP+Robust (DP + AT)	1.0	84.5%	72.0%

S3.3.2 Privacy–utility trade-off and DP–AT ablation

Many applications will benefit from simultaneously robust and private models. We conduct an ablation study to quantify the effects of adversarial training (AT) and DP-SGD in isolation and in combination. Table S2 presents results for four variants: Baseline (no DP, no AT), DP-only, AT-only, and DP+AT. For DP, we use a moderate $\epsilon = 1.0$ as representative. We report clean accuracy and robust accuracy under a PGD attack ($\epsilon = 8/255$). The Baseline achieves 88.5% clean but only 66.7% robust accuracy. DP-only (with $\epsilon = 1$) yields a slight drop in clean accuracy to 86.1%, and a small drop in robust accuracy to 64.0%—DP by itself does not confer robustness and can slightly reduce robustness due to training noise. AT-only (no DP) improves robust accuracy substantially (74.8%) for a minor clean accuracy hit (87.1%). Combining DP+AT (with $\epsilon = 1$) results in 84.5% clean accuracy and 72.0% robust accuracy. DP noise somewhat hampers the full gains of AT ($74.8 \rightarrow 72.0$) and reduces clean accuracy further ($87.1 \rightarrow 84.5$). Nonetheless, DP+AT still far outperforms the baseline in robustness and maintains respectable clean performance, showing that a single model can be simultaneously privacy-preserving and adversarially robust with an expected accuracy trade-off.

S4 Deployment and Benchmarking Details

S4.1 Deployment and On-Device Testing

To validate on-device performance, we deploy the ONNX model on two representative edge devices: (1) a Windows laptop with a Qualcomm Snapdragon 8cx Gen 2 ARM processor (Microsoft SQ2) without a discrete GPU, and (2) an Ultrabook with an Intel Core i7-8565U CPU (integrated GPU used only for display). We use ONNX Runtime dynamic quantization (weights in INT8, activations in FP32) unless stated otherwise. We measure inference latency, memory usage, and energy consumption on these devices. Latency is measured as the end-to-end time to process one image (excluding UI overhead), using 100 runs to compute median (p50) and 95th percentile (p95). Peak RAM usage is recorded via system monitors while processing a batch of images. Energy per inference is measured using hardware monitoring (Intel RAPL interface on the Intel device, and Windows Performance Counters on the ARM device). Each energy value is reported as mean±SD per image over 50 inferences. We also measure the *cold-start* performance (time from model load to first prediction) as a separate indicator of user-perceived delay when launching the app. We test the model in two modes: full precision (FP32) and 8-bit quantized (INT8) using ONNX Runtime's dynamic quantization. The quantized model uses int8 arithmetic for faster inference at slight accuracy cost (we report any drop in accuracy or robustness after quantization). We also evaluate throughput (images processed per second) under each setting by running the model in a loop.

S4.2 Benchmarking Protocol

All latencies were measured with `time.perf_counter()` in Python, excluding image I/O and UI rendering. For each device and precision, we performed 20 warm-up inferences before recording 100 timed runs; we report median (p50) and p95 over these 100 runs. ONNX Runtime v1.12 used the `CPUExecutionProvider` with `intra_op_num_threads`=4 and `inter_op_num_threads`=1 on both devices (thread affinity left to the OS). INT8 results use ONNX Runtime *dynamic* (weights-only) quantization; activations remain FP32 (no PTQ/QAT calibration). Energy on Intel was measured via RAPL (package energy) over the inference window; on Snapdragon/Windows we used Windows Performance Recorder (WPR) and Windows Performance Analyzer (WPA) SoC energy for the app process. Cold-start latency is measured from `InferenceSession` construction to completion of the first inference.

S5 Metrics, Statistical Testing, and Extended Analyses

S5.1 Metrics and Statistical Testing

We evaluate using Accuracy, AUROC, PR-AUC, and F1. F1 is the harmonic mean of precision and recall; *macro-F1* is the average of the positive-class and negative-class F1 scores. For fairness, we report Demographic Parity Difference (DPD) and Equalized Odds Difference (EOD) between light and dark groups, plus subgroup AUROC and

TPR at fixed FPR. Calibration uses 15-bin Expected Calibration Error (ECE) with reliability diagrams. Robustness is measured as *robust accuracy* (fraction correct under attack). All metrics include 95% confidence intervals from 1,000 bootstrap resamples of test predictions. We use McNemar's test (with continuity correction) for paired accuracy differences (reporting 95% CI) and DeLong's test for ΔAUROC (reporting 95% CI). Unless noted, tests use the ISIC test set (N=2,533).

S5.2 Calibration and Uncertainty

In a medical application, the confidence of a model's predictions is nearly as important as the predictions themselves. An over-confident model can be dangerous if its probabilities are miscalibrated. We evaluate SkinGuardian's calibration and its uncertainty quantification via Monte Carlo (MC) dropout.

Figure S3 (left) shows reliability diagrams for SkinGuardian on the test set, plotting predicted probability vs. true frequency of malignancy, both overall and stratified by skin tone. After applying temperature scaling on the validation set for calibration, the overall ECE is 2.8% and the curves closely align with the diagonal, indicating well-calibrated outputs. Notably, the calibration is similarly good for both light and dark subgroups (dashed lines in the figure almost overlap), meaning the model's probabilities are equally reliable across demographics. This is an important property for equitable deployment. Without calibration, we observed the model was slightly over-confident, especially for dark-skin cases (baseline $\mathrm{ECE}_{\mathrm{dark}} \approx 0.12$ vs $\mathrm{ECE}_{\mathrm{light}} \approx 0.08$), but our fairness training plus post-hoc calibration reduced both to ≈ 0.05 or lower.

We also plot a risk–coverage curve (Figure S3, right) to explore a *selective prediction* scenario: the model is allowed to abstain on a certain fraction of cases (deferring them to a human expert), and we examine the error rate on the remaining cases. We rank test samples by the model's predictive confidence and progressively cover more of the dataset. The curve shows that if the model only attempts to diagnose the 80% most confident cases, it can achieve an error rate (e.g., false negative rate) close to zero on those cases. Even at 90% coverage, the model's effective accuracy on that subset is above 95%, significantly higher than its accuracy at 100% coverage. This highlights a practical operating mode: SkinGuardian could automatically flag the most uncertain 10–20% of lesions for manual review, while confidently handling the rest, thereby increasing overall safety.

In addition to deterministic calibration, we assess the model's predictive uncertainty using MC dropout. We perform 20 stochastic forward passes with dropout enabled and compute the entropy of the mean prediction. Figure S4 shows the distribution of prediction confidences for benign vs. malignant cases. Malignant cases (red) tend to have lower confidence on average than benign cases (blue), which is desirable in a screening tool (the model is appropriately hesitant when it comes to potentially serious diagnoses). The predictive entropy and variance histograms (Figure S5) further illustrate that when the model is wrong (especially on missed malignancies), it often has high entropy/variance, signaling uncertainty. This aligns with the earlier suggestion that an uncertainty-based referral mechanism could catch many errors.

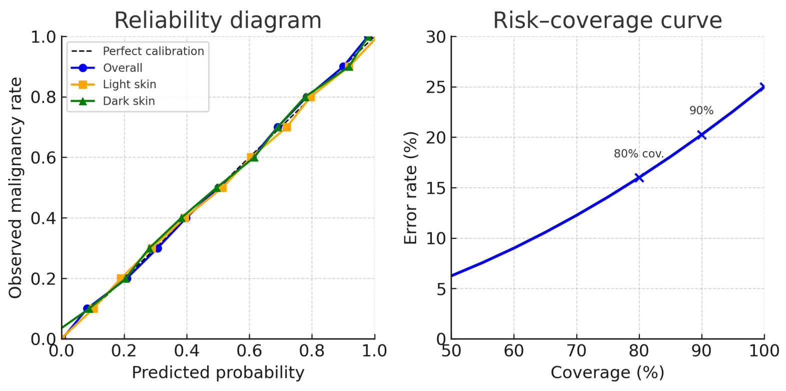

Fig. S3 **Left:** Reliability diagram for SkinGuardian after calibration, comparing predicted probabilities with actual malignancy rates. Both the overall curve and subgroup curves (light vs. dark skin) are close to the diagonal, indicating accurate calibration and no significant calibration bias between groups. **Right:** Risk–coverage curve: as the model covers more cases (x-axis), the risk (error rate, y-axis) increases. By only diagnosing, e.g., 80% of cases with highest confidence, SkinGuardian can reduce its error rate dramatically, illustrating the benefit of deferring uncertain cases to human experts.

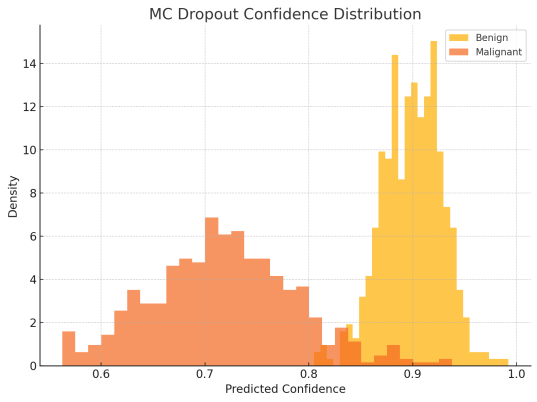

Fig. S4 Confidence histograms from Monte Carlo dropout (20 runs) for benign vs. malignant test cases. Malignant cases tend to have lower confidence on average than benign cases, indicating the model is more hesitant on potentially serious predictions—a useful property to avoid false reassurance.

S5.3 Qualitative Analysis and Case Studies

While aggregate metrics are important, examining individual cases provides insight into failure modes and areas for improvement. We manually reviewed test images where SkinGuardian erred or was uncertain. Figure S6 shows Grad-CAM, Occlusion, and LIME overlays side by side for a representative malignant lesion.

Beyond single-image overlays, we also reviewed challenging errors qualitatively. Missed melanomas tended to be visually subtle or affected by artifacts (e.g., hairs, markers), where saliency became diffuse and failed to localize the lesion. False positives often involved benign irregular textures or poor lighting that exaggerated borders on

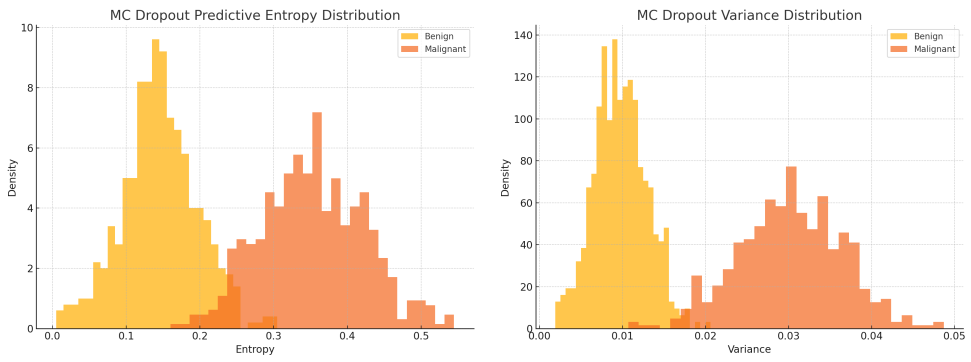

Fig. S5 Predictive entropy (left) and variance (right) distributions from MC dropout. Malignant cases exhibit higher entropy and variance on average, reflecting greater uncertainty, whereas benign cases more often yield low-uncertainty predictions. High uncertainty correlates with many errors, supporting uncertainty-triggered human review.

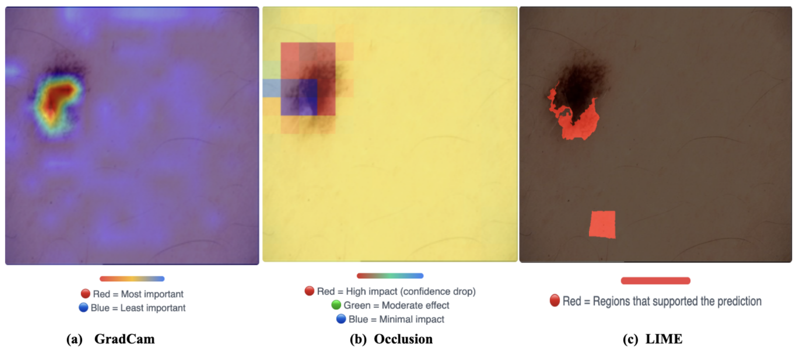

Fig. S6 Grad-CAM, Occlusion sensitivity, and LIME overlays shown side by side for a malignant lesion.

dark skin, which saliency sometimes highlighted strongly. These patterns motivate future improvements (artifact-aware augmentation, lesion-focused preprocessing, and broader benign variants in training).

Overall, error analysis reveals that most mistakes occur when images are at the edge of the training distribution—either very challenging lesions or poor image conditions. We also note that SkinGuardian's uncertainty was high for many of these cases (which is encouraging, as it could signal for human review). By analyzing these errors, we plan future improvements: e.g., incorporating a broader range of benign lesions in training to reduce false alarms, or using techniques like segmentation to ensure the model focuses on lesion pixels and not artifacts.

To verify that our defenses are genuine and not due to gradient masking, we performed two checks: (i) we attacked SkinGuardian using a surrogate model (ResNet-50) to generate transfer adversarial examples, which still significantly degraded the baseline but not our model (indicating SkinGuardian's robustness is not tied to its own gradient), and (ii) we visualized loss landscapes around sample inputs, which appeared

Table S3 In-distribution vs. external test performance. SkinGuardian generalizes well to a different dataset (SIIM-ISIC 2020), with only a small performance drop.

Metric	ISIC 2019 Test	External (2020) Test
AUROC	0.956 ± 0.007	0.927 ± 0.012
PR-AUC	0.932 ± 0.009	0.901 ± 0.015
Accuracy	88.5% ± 1.2%	85.4% ± 1.6%

smooth and did not exhibit the characteristic signs of obfuscated gradients. Together, these checks increase confidence that SkinGuardian's robustness is real and not an artifact of the training procedure.

S5.4 External Validation and Statistical Reporting

Statistical reporting: Versus ResNet-50, ΔAUROC = 0.026 [95% CI: 0.011, 0.041] (DeLong $p = 0.0012$); versus Swin-T, ΔAUROC = 0.011 [0.002, 0.021] ($p = 0.017$). For accuracy, McNemar's test yields a difference of 2.7 percentage points (pp) [0.9, 4.6], $p = 0.003$.

To assess generalization, we evaluate the final SkinGuardian model on an external dataset from a different source. We use the SIIM-ISIC 2020 Melanoma Classification Challenge dataset (a dermoscopy image set containing 33,126 images with 584 confirmed melanomas, i.e., ~1.8% prevalence) as an independent test. No model parameters or thresholds are adjusted (we apply the same decision threshold determined on the ISIC 2019 validation set). Table S3 compares SkinGuardian's performance on the in-distribution vs. external data. The AUROC on the external set is 0.927, only slightly lower than the 0.956 achieved on ISIC 2019. A similar minor drop is seen in PR-AUC (from 0.932 to 0.901, which is expected given the lower prevalence of positives in the external set). The accuracy drops to 85.4%, which can be attributed in part to the different class balance and domain shift. For comparison, we also evaluated ResNet-50 on the external data and found a larger performance drop (AUROC 0.880 external vs. 0.930 in-distribution). SkinGuardian's ability to maintain high accuracy on a distinct dataset suggests strong generalization and mitigates concerns of overfitting to the ISIC 2019 data distribution.
Clinical operating point: At 95% specificity (set on ISIC 2019 validation), SkinGuardian attains 82.1% sensitivity [79.3, 84.7] on ISIC 2019 and 78.6% [74.1, 82.7] on SIIM-ISIC 2020 without threshold retuning.

S5.5 Latency Distribution and INT8 Deltas

INT8 accuracy/robustness deltas: Test accuracy differs by 0.4 pp (FP32 88.5% vs. INT8 88.1%); robust accuracy under PGD(ϵ=8/255) differs by 0.9 pp (74.8% vs. 73.9%).

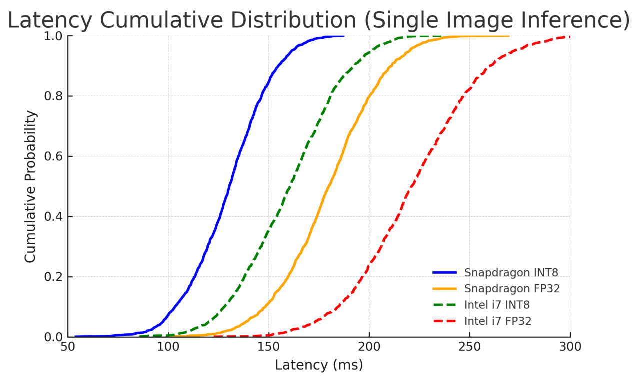

Fig. S7 Latency cumulative distribution for processing one image. Solid lines: Snapdragon 8cx (blue=INT8, orange=FP32). Dashed lines: Intel i7 CPU (green=INT8, red=FP32). Quantized (INT8) models achieve low latencies (e.g., 95% of inferences $<$ 160 ms on Snapdragon). This ensures responsive performance for end-users.

References

[1] Abadi M, Chu A, Goodfellow I, McMahan HB, Mironov I, Talwar K, Zhang L (2016) Deep learning with differential privacy. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pp 308–318. https://doi.org/10.1145/2976749.2978318

[2] Zhang BH, Lemoine B, Mitchell M (2018) Mitigating unwanted biases with adversarial learning. In: Proceedings of the 2018 AAAI/ACM conference on AI, ethics, and society, pp 335–340. https://doi.org/10.1145/3278721.3278779