Dual-Head CNN for Simultaneous Edge and Corner Detection from a Single Image

1 Problem Definition

The project aims to detect both edge and corner features using deep convolutional neural networks. Classical edge and corner detectors are not learned from data but are done using handcrafted filters like Harris or Canny. This project formulates the task as a dual-output pixel-wise binary prediction problem, where a shared encoder extracts scene features from 512 x 512 input image patches cropped from a single 5616 × 3744 RGB image. The mathematical formulation of the problem is described in Figure [1]. The ground truth maps are obtained using Canny and Harris detectors over the image patches. The output is a pixel-wise prediction of both edges and corners. This task is challenging, yet interesting due to class imbalance as most pixels belong to the background (pixel value 0), lack of generalization from one image, and sparse supervision as the models get very little feedback during training about where these edge and corner features truly exist as edges and corners occupy very few pixel locations in the image. A successful solution would involve network learning to distinguish structure-rich regions from uninformative backgrounds, and we measure performance using F1-score and Intersection-over-Union (IoU) for both edges and corners on held-out validation set.

2 Related Work

Potje et al. [1] utilized a backbone Hourglass CNN network similar to U-Net for efficient computation in original image resolution of dense heat map of specialized keypoints which inspired usage of similar U-Net modelling strategy with skip connections for edge and corner detection. Barroso-Laguna et al. [2] proposed a Key.Net model which showed possibility of keypoint detection through combination of handcrafted and learned filters. The authors generated synthetic training set from ImageNet ILSVRC 2012 dataset using random geometric transformations and setting 25% as held-out validation set. Inspiration taken from this to generate synthetic training set by cropping 512 x 512 patches from single image and leaving out 20% of total crops for validation. Bhattacharjee et al. [3] proposed Swin Transformer model that encodes input image into shared representation and has task-specific transformer-based decoder heads and uses shared attention for modelling dependencies across tasks like depth estimation, 2D keypoint detection, edge detection, semantic segmentation etc. This inspired the dual-head decoder concept in our approach. DeTone et al. [4] proposed training detectors on synthetic datasets of simple geometric shapes, then applying to real image multiple times through random homographies sampling for pseudo-label generation. Zhao et al. [5] proposed learning keypoints using pseudo label extracted using available detectors and based on it performing supervised learning. These works inspired the ground truth generation via Harris and Canny detectors. Soria et al. [6] proposed a light CNN named Tiny and Efficient Edge Detector (TEED) with only 58K parameters which utilized skip connections and modified Weighted Cross-Entropy (WCE) as loss function for end-to-end supervised learning. The convolution layers had the Mish activation function. This inspired the concept of skip connections and ablation study to utilize activation functions other than ReLU.

3 Method

Our approach involved designing dual-head CNN architecture where a shared encoder extracts rich hierarchical features, and two dedicated decoder heads simultaneously predict edges and corners. Our design explored two architectural backbones: First UNetDualHead – A lightweight U-Net inspired encoder-decoder with skip connections. Sample architecture seen in Figure [2]. Next ResNetLiteDualHeadSkip – A ResNet-style encoder with lightweight residual blocks and symmetric upsampling path. Sample architecture seen in Figure [3].

From a high-resolution 5616 × 3744 image, 512 × 512 overlapping patches were extracted with a stride of 256, yielding the dataset. However, some parts of the high-resolution image contain sky which doesn't have corners and edges, so only 104 structure-rich crops were retained by

filtering 156 crops with fewer than 5% non-zero pixels in both the Canny edge and Harris corner maps to ensure that the model focuses on learning from informative regions only. Edge ground truth was generated using OpenCV's Canny detector and Corner ground truth was generated using Harris detector with experimented dilation and thresholding.

Ablation studies were conducted with the following loss strategies - Binary Cross Entropy (BCE) (baseline) details in Figure [4], Weighted BCE (Assigning higher loss weight to corner loss) details in Figure [5], Focal Loss (Focusing on hard-to-classify pixels with modulated terms based on confidence) details in Figure [6]. Training was done using the following optimizers for ablative comparison: Adam, AdamW (weight decay regularization), SGD with momentum, details given in Figures [8], [9] and [7] respectively. Exploration using various learning rates [10^{-3}, 10^{-4}, 10^{-2}] and activation functions like Leaky ReLU and Mish for stable gradient flow, were experimented for improving corner map predictions.

Training was done with input image size 512 × 512, Batch Size: 4, Epochs: 35 for UNetDualHead and 100 for ResNetDualHeadSkip Model. Evaluation was done using metrics: Intersection-over-Union (IoU) and F1-score computed for each task (edge, corner). A flowchart of the pipeline (image preprocessing → model training → inference) is shown in Figure [10].

4 Results

Experiments conducted using the preprocessed dataset consisting of 104 high-structure 512×512 image patches extracted from a high-resolution image were done using ablative variants of UNetDualHead and ResNetDualHeadSkip backbones to jointly predict Canny edges and Harris corners. The IoU (Intersection over Union) and F1-Score were the evaluation metrics and results are shown in Table [1]. The UNetDualHead model with AdamW optimizer performed the best on IoU Edge and F1 Edge metrics while UNetDualHead with Mish activation function performed best on the IoU Corner and F1 Corner metrics. For ResNetDualHeadSkip model category, the Mish activation variant performed the best across all metrics.

The training and validation loss curves of the best performing models with their individual metrics are plotted in Figures [11] to [13] using symbols and line styles making it quite accessible for color-blind readers. All other models' plotting is stored in the project repository for reference. Single Image Edge and Corner Prediction results by these best performing models in the above-mentioned specific metrics have been displayed in Figures [14] - [15]. It can thus state that UNetDualHead outperforms ResNetDualHeadSkip when predicting edges while the corner predictions for ResNetDualHeadSkip are relatively better.

5 Reflection

Edge and corner detection models, have applications in robotics and smart infrastructure, but raise ethical concerns if misused. When integrated into surveillance systems, high-precision structural detection could inadvertently compromise personal privacy by enabling detailed environmental mapping without consent. Reliance on sparse, biased single image training data may result in poor generalization, leading to errors in critical scenarios. Deep Learning Models should be utilized with model explainability techniques for fairness purposes. Pedestrians, building occupants, or assistive technology users could be affected by flawed or intrusive deployments of detection models. These issues emphasize need for transparent, fair, and responsible deployment of vision models in accordance with Toronto Declaration [10].

6 Conclusion

This project successfully demonstrated dual-head CNN architecture for joint edge and corner detection from a single image. Among various configurations, UNetDualHead with AdamW optimizer and ResNetDualHeadSkip with Mish Activation achieved the best performance in their model ablation groups. Future work could explore better generalization with larger computes on bigger datasets formed using smaller strides with more overlapping patches or using standard datasets (Eg. BSDS500) or leveraging attention-based modules as in [3].

References

[1] G. Potje, F. Cadar, A. Araujo, R. Martins and E. R. Nascimento, "Enhancing Deformable Local Features by Jointly Learning to Detect and Describe Keypoints," 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 2023, pp. 1306-1315, doi: 10.1109/CVPR52729.2023.00132.

[2] Barroso-Laguna, A., Riba, E., Ponsa, D., & Mikolajczyk, K. (2019). Key.Net: Keypoint Detection by Handcrafted and Learned CNN Filters. arXiv.org. https://arxiv.org/abs/1904.00889

[3] D. Bhattacharjee, T. Zhang, S. Süsstrunk and M. Salzmann, "MuIT: An End-to-End Multitask Learning Transformer," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 2022, pp. 12021-12031, doi: 10.1109/CVPR52688.2022.01172.

[4] DeTone, D., Malisiewicz, T., & Rabinovich, A. (2017). SuperPoint: Self-Supervised Interest Point Detection and Description. arXiv.org. https://arxiv.org/abs/1712.07629

[5] Zhao, S., Gong, M., Zhao, H., Zhang, J., & Tao, D. (2023). Deep corner. International Journal of Computer Vision, 131(11), 2908–2932. https://doi.org/10.1007/s11263-023-01837-3

[6] Soria, X., Li, Y., Rouhani, M., & Sappa, A. D. (2023). Tiny and efficient model for the edge detection generalization. arXiv.org. https://arxiv.org/abs/2308.06468

[7] Lin, T., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. arXiv.org. https://arxiv.org/abs/1708.02002

[8] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv.org. https://arxiv.org/abs/1412.6980

[9] Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. arXiv.org. https://arxiv.org/abs/1711.05101

[10] The Toronto Declaration. (2018). https://www.torontodeclaration.org/declaration-text/english/

Let the input patch be denoted as $X \in R^{3\times H\times W}$.
The model predicts two binary masks:

- **Edge map**: $Y_{edge} \in [0,1]^{1\times H\times W}$
- **Corner map**: $Y_{corner} \in [0,1]^{1\times H\times W}$
- **Combined output**: $Y = f_\theta(X) \in [0,1]^{2\times H\times W}$

where f_θ is the CNN model parameterized by weights θ and the two output channels correspond to edge and corner predictions.

The objective is to minimize the **binary classification loss** between predicted and ground-truth maps:

$$L_{total} = \lambda L_{edge} + (1-\lambda) L_{corner}$$

where each L_{edge} and L_{corner} can be BCE, focal loss, or a weighted variant.

Figure [1]: *Mathematical formulation of the joint edge and corner detection task as a dual-output pixel-wise binary classification problem. Training is supervised using a weighted sum of binary classification.*

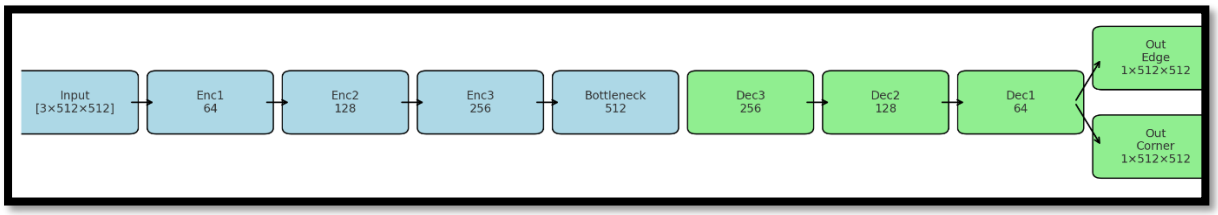

Figure [2]: ***Block-style Architecture of UNetDualHead Network*** *designed for simultaneous edge and corner detection from a single RGB image of resolution 3×512×512. The model follows the classical U-Net encoder-decoder structure with skip connections. The encoder path consists of three convolutional blocks (Enc1 → Enc2 → Enc3) that progressively downsample the input while increasing feature depth (64 → 128 → 256 channels). A bottleneck block with 512 channels encodes the compressed latent representation. The decoder path (Dec3 → Dec2 → Dec1) upsamples and refines spatial features using transposed convolutions and concatenates them with corresponding encoder features through skip connections. Finally, two separate 1×1 convolution heads predict the* ***Edge Map*** *and* ***Corner Map****, each with shape 1×512×512. This dual-head setup enables the network to learn both edge- and corner-relevant features from shared representations.*

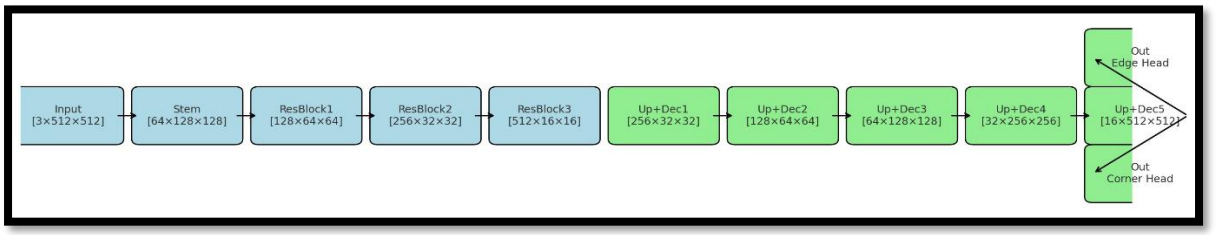

Figure [3]: ***Block-style Architecture of ResNetLiteDualHeadSkip Network*** *takes a 3×512×512 image as input and uses a stem layer followed by three residual blocks to encode deep features while reducing spatial dimensions. The decoder consists of five upsampling and convolution blocks that restore spatial resolution. Two separate output heads predict the edge and corner maps of size 1×512×512. Architecture leverages residual learning and skip connections for accurate joint edge-corner detection.*

Let $y \in \{0,1\}$ be the ground truth, and $y' \in [0,1]$ be the predicted probability.

$$L_{BCE} = -[y \log(y') + (1-y) \log(1-y')]$$

Figure [4]: *Mathematical formulation of* ***Binary Cross-Entropy (BCE) Loss****, which quantifies the error through distance between predicted probabilities and binary ground truth labels for pixel-wise classification.*

Let y ∈ {0,1} be the ground truth, and y' ∈ [0,1] be the predicted probability

$$L_{Weighted\ BCE} = -\left[w_1\, y \log(y') + w_0\,(1-y)\log(1-y')\right]$$

Where:

- $w_1 = \frac{1}{p}$, $w_0 = \frac{1}{1-p}$ where p is the proportion of positive pixels

Figure [5]: *Mathematical formulation of **Weighted Binary Cross-Entropy (BCE) Loss**, which tackles class imbalance by assigning higher weights to underrepresented foreground pixels using inverse class frequency.*

Let y ∈ {0,1} be the ground truth, and y' ∈ [0,1] be the predicted probability

$$L_{Focal} = -\,\alpha\,(1-y')^{\gamma}\, y \log(y') - (1-\alpha)\, y'^{\gamma}\,(1-y)\log(1-y')$$

Where:

- $\alpha \in [0,1]$: balancing parameter (Eg., 0.25)
- $\gamma \geq 0$: focusing parameter (Eg., 2.0)
- When $\gamma = 0$, focal loss reduces to BCE

Figure [6]: *Mathematical expression of **Focal Loss** (Lin et al., [7]), which dynamically down-weights easy examples and focuses training on hard negatives using a modulating factor $(1-y')^{\gamma}$, where γ is the focusing parameter and "α" balances class importance.*

$$v_t = \mu\, v_{t-1} - \eta\ \nabla_\theta J(\theta_t)$$
$$\theta_{t+1} = \theta_t + v_t$$

Where:

- $\boldsymbol{\theta_t}$: parameters at step t
- $\boldsymbol{\eta}$: learning rate
- $\boldsymbol{\mu}$: momentum coefficient (e.g., 0.9)
- $\mathbf{J(\theta)}$: loss function
- $\boldsymbol{\nabla_\theta J(\theta_t)}$: gradient of the loss w.r.t. parameters

Figure [7]: *Mathematical formulation of **Stochastic Gradient Descent (SGD) with momentum**, where velocity v_t accumulates past gradients to accelerate convergence and dampen oscillations. Momentum coefficient μ = 0.9 helps smooth updates during training.*

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta J(\theta_t) \quad \text{(1st moment)}$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\nabla_\theta J(\theta_t)^2 \quad \text{(2nd moment)}$$
$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \quad \text{(bias correction)}$$
$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$
$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

Where:

- $\boldsymbol{\beta_1} = 0.9$, $\boldsymbol{\beta_2} = 0.999$ are decay rates
- $\boldsymbol{\epsilon}$: small constant to prevent division by zero
- $\boldsymbol{\eta}$: learning rate

Figure [8]: *Mathematical formulation of the **Adam optimizer** adapted from (Kingma & Ba [8]), combining momentum (1st moment) and adaptive learning rates (2nd moment) with bias correction for efficient and stable training.*

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta J(\theta_t) \quad \text{(1st moment)}$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\nabla_\theta J(\theta_t)^2 \quad \text{(2nd moment)}$$
$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \quad \text{(bias correction)}$$
$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$
$$\theta_{t+1} = \theta_t - \eta\left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon} + \lambda\theta_t\right)$$

Where:

- λ: weight decay coefficient (regularization strength)
- The term $\lambda\theta_t$ applies **L2 regularization** separately from adaptive gradient update

Figure [9]: *Mathematical formulation of AdamW optimizer (Loshchilov & Hutter [9]), extending Adam by decoupling weight decay (L2 regularization) from the adaptive gradient update for improved generalization.*

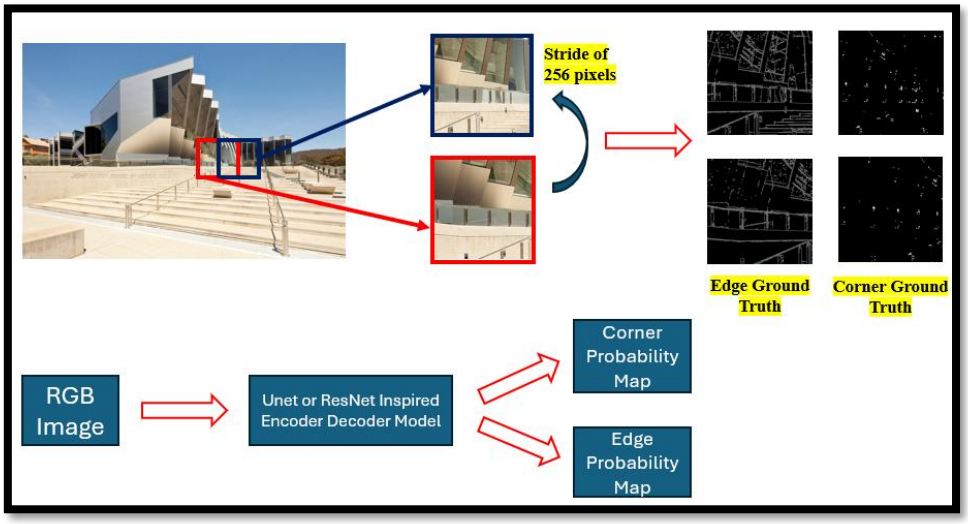

Figure [10]: *Overview of the **proposed pipeline**. The high-resolution input image is cropped into overlapping patches (stride = 256 pixels). Each patch is passed through a dual-head encoder-decoder network (U-Net or ResNet-inspired), which outputs pixel-wise probability maps for edge and corner detection. Ground truth labels are derived using classical detectors (Canny for edges, Harris for corners) to supervise training.*

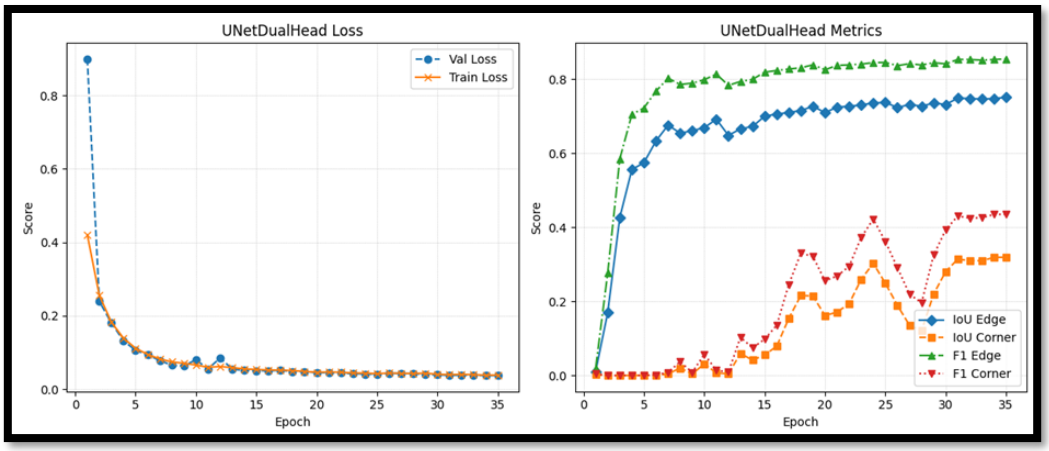

Figure [11]: *Training and validation loss curves (left) and metric trends (right) for the **UNetDualHead model trained using AdamW optimizer and Binary Cross Entropy (BCE) loss**. The model shows rapid convergence in edge prediction (high IoU and F1), while corner detection improves more gradually over epochs.*

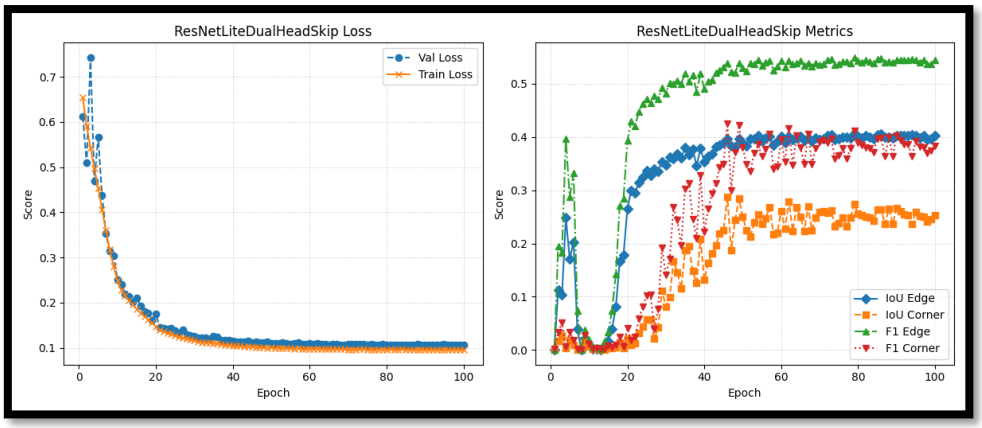

Figure [12]: *Training/validation loss (left) and evaluation metrics (right) for the **ResNetLiteDualHeadSkip model trained with Mish activation and Binary Cross Entropy (BCE) loss**. Edge detection converges steadily with moderate performance, while corner detection shows gradual improvement, reflecting the model's robustness in handling class imbalance.*

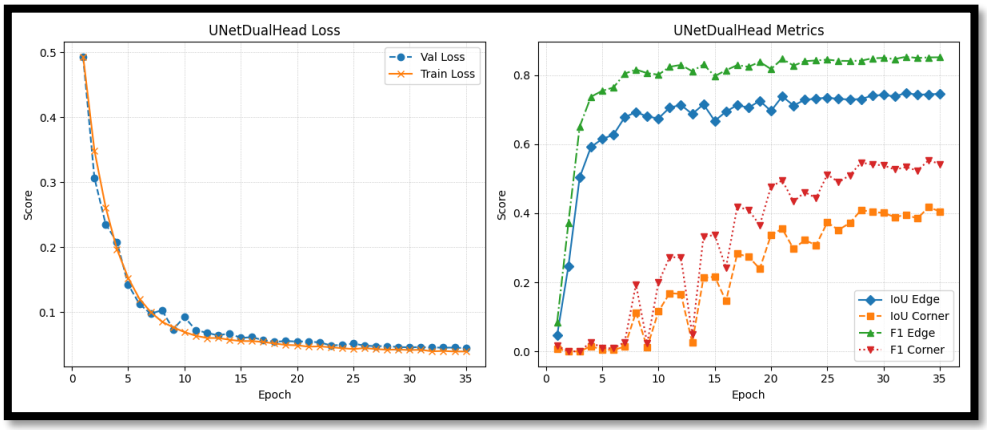

Figure [13]: *Training/validation loss (left) and evaluation metrics (right) for the **UNetDualHead model trained with Mish activation and Binary Cross Entropy (BCE) loss**. Both edge and corner predictions show consistent improvement across epochs, with notable gains in F1 and IoU metrics, especially for corner detection.*

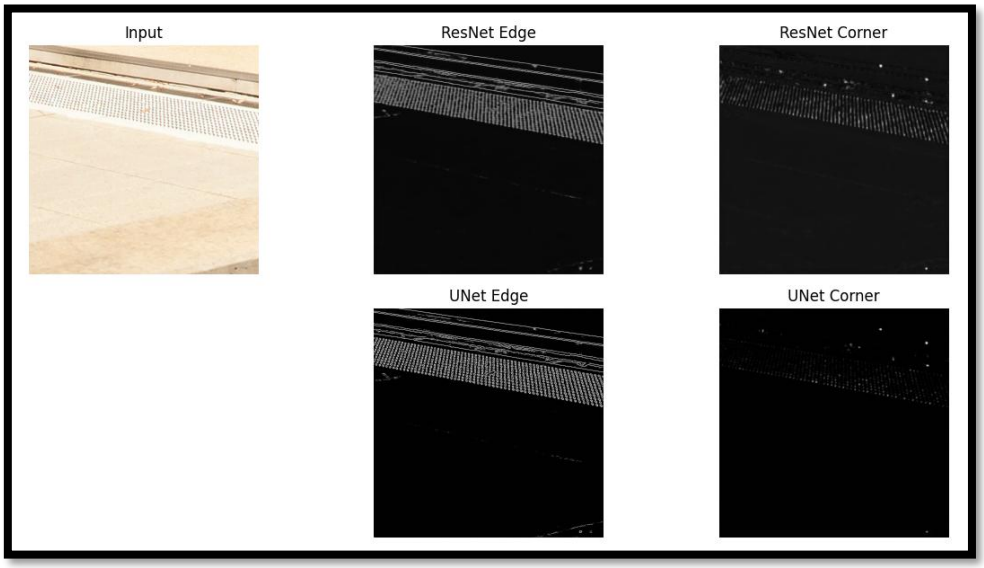

***Figure [14]:** Qualitative comparison of edge and corner predictions from **ResNetLiteDualHeadSkip and UNetDualHead models using AdamW optimize**r. The UNet model produces sharper and more defined edge maps and while the ResNet variant shows clearer corner activations, particularly around textured regions.*

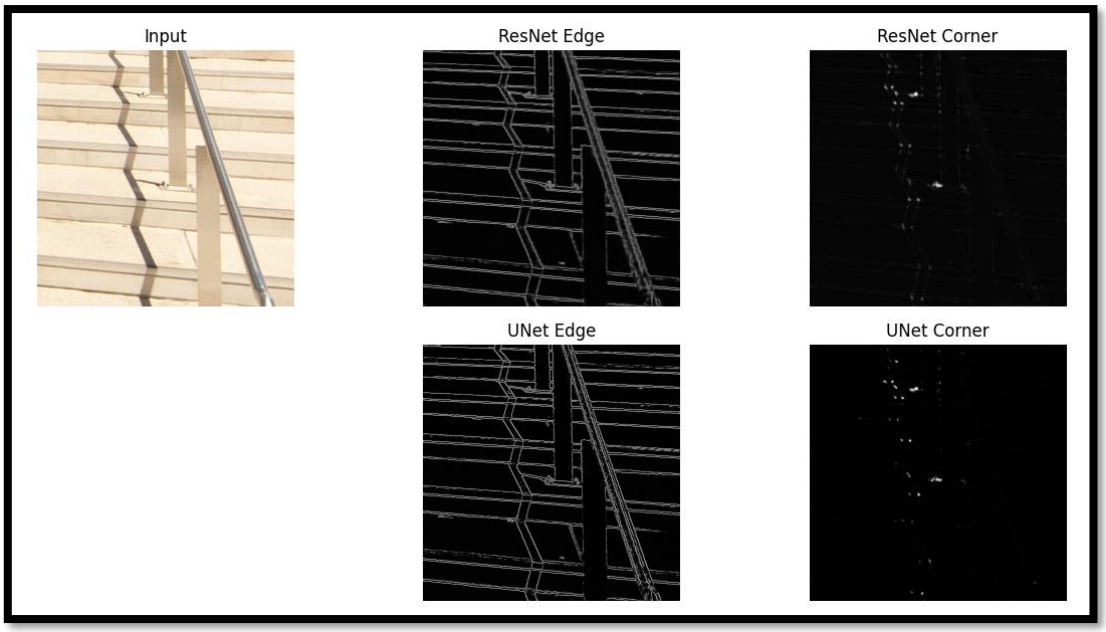

Figure [15]: *Prediction outputs from **ResNetLiteDualHeadSkip and UNetDualHead models using Mish activation**. Both models provide continuous and precise edge delineation along stair and railing structures, while their corner detections are relatively weak.*

Table [1]: *Comparison of performance metrics (IoU and F1-score for edge and corner detection) across various ablations using different model types (UNetDualHead and ResNetLiteDualHeadSkip) and training configurations. Metrics are computed on the held-out validation set.*

Model	IoU Edge ↑	IoU Corner ↑	F1 Edge ↑	F1 Corner ↑
UNetDualHead + BCE	0.728	0.357	0.839	0.505
UNetDualHead + WBCE	0.686	0.398	0.809	0.536
UNetDualHead + Focal Loss	0.705	0.327	0.821	0.461
UNetDualHead (Adam + Slower Learning Rate) + BCE	0.610	0.01	0.754	0.02
UNetDualHead (Adam + Faster Learning Rate) + BCE	0.732	0.121	0.842	0.205
UNetDualHead (AdamW) + BCE	**0.751**	0.319	**0.854**	0.435
UNetDualHead (SGD with Momentum) + BCE	0.692	0	0.815	0
UNetDualHead (LeakyReLU) + BCE	0.71	0.38	0.828	0.523
UNetDualHead (Mish) + BCE	0.748	**0.417**	0.851	**0.55**
ResNetDualHeadSkip + BCE	0	0.014	0	0.02
ResNetDualHeadSkip + WBCE	0	0.088	0	0.155
ResNetDualHeadSkip + Focal Loss	0.0228	0.152	0.039	0.239
ResNetDualHeadSkip (Adam + Slower Learning Rate) + BCE	0.145	0.006	0.254	0.012
ResNetDualHeadSkip (Adam + Faster Learning Rate) + BCE	0.264	0	0.377	0
ResNetDualHeadSkip (AdamW) + BCE	0.346	0	0.486	0
ResNetDualHeadSkip (SGD with Momentum) + BCE	0.0002	0	0.0004	0
ResNetDualHeadSkip (LeakyReLU) + BCE	0.303	0.181	0.439	0.303
ResNetDualHeadSkip (Mish) + BCE	**0.404**	**0.28**	**0.549**	**0.402**