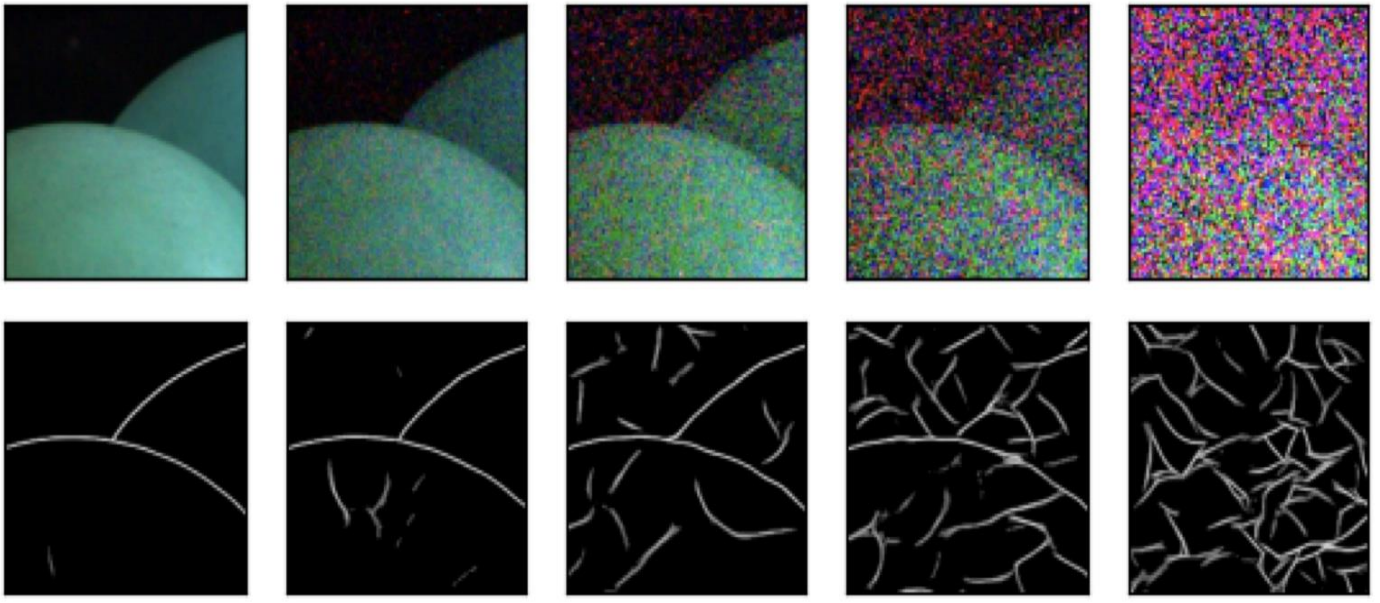


Improving Noisy Boundary Extractions via Cross-patch Attention

Thomas Snow, Yuchong Zhang
University of Toronto

Motivation

- Existing boundary extraction algorithms perform worse on noisy input images: the outputs often contain incorrect extra boundaries or miss parts of the correct boundaries.
- 
- Figure 1: Boundary extraction result as noise increases using the Boundary Attention Model from [1]
- The extraction results can differ a lot even across noisy samples of the same image with the same noise level.
 - Thus, we hope to train a model that takes in the boundary result of multiple noisy samples of the same image and extracts the correct boundaries.

Related Work

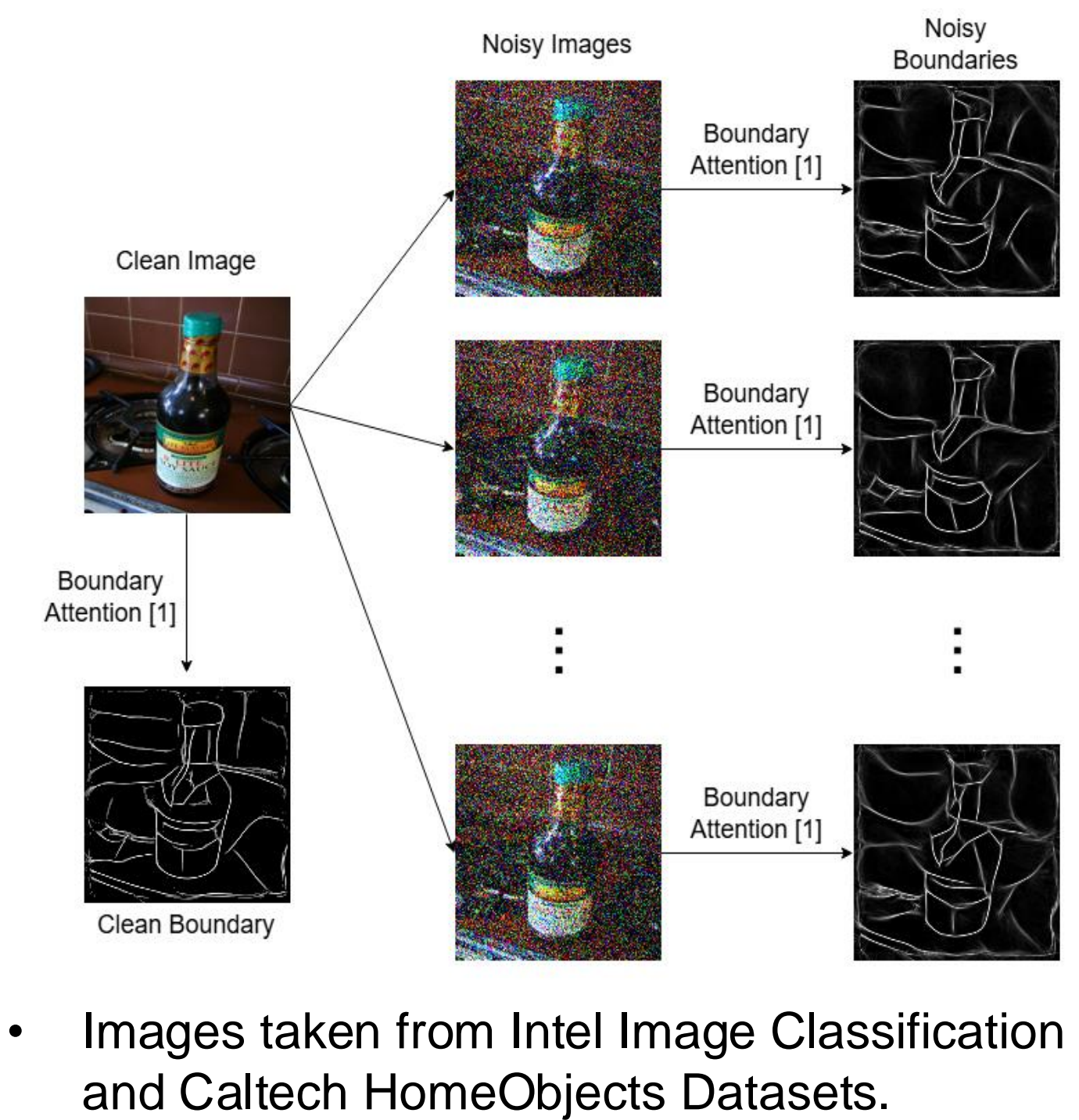
Boundary Extraction at Low SNR

- [1] and [2] represent boundary elements as “junctions” where boundaries meet and learn junction representations.
- Trained on synthetic images, but it claims to generalize well to real-world images.

Vision Transformer

- The vision transformer takes in patches of the input images as “tokens” to learn their spatial relationships.
- Motivation to apply attention to corresponding patches of the noisy boundaries.

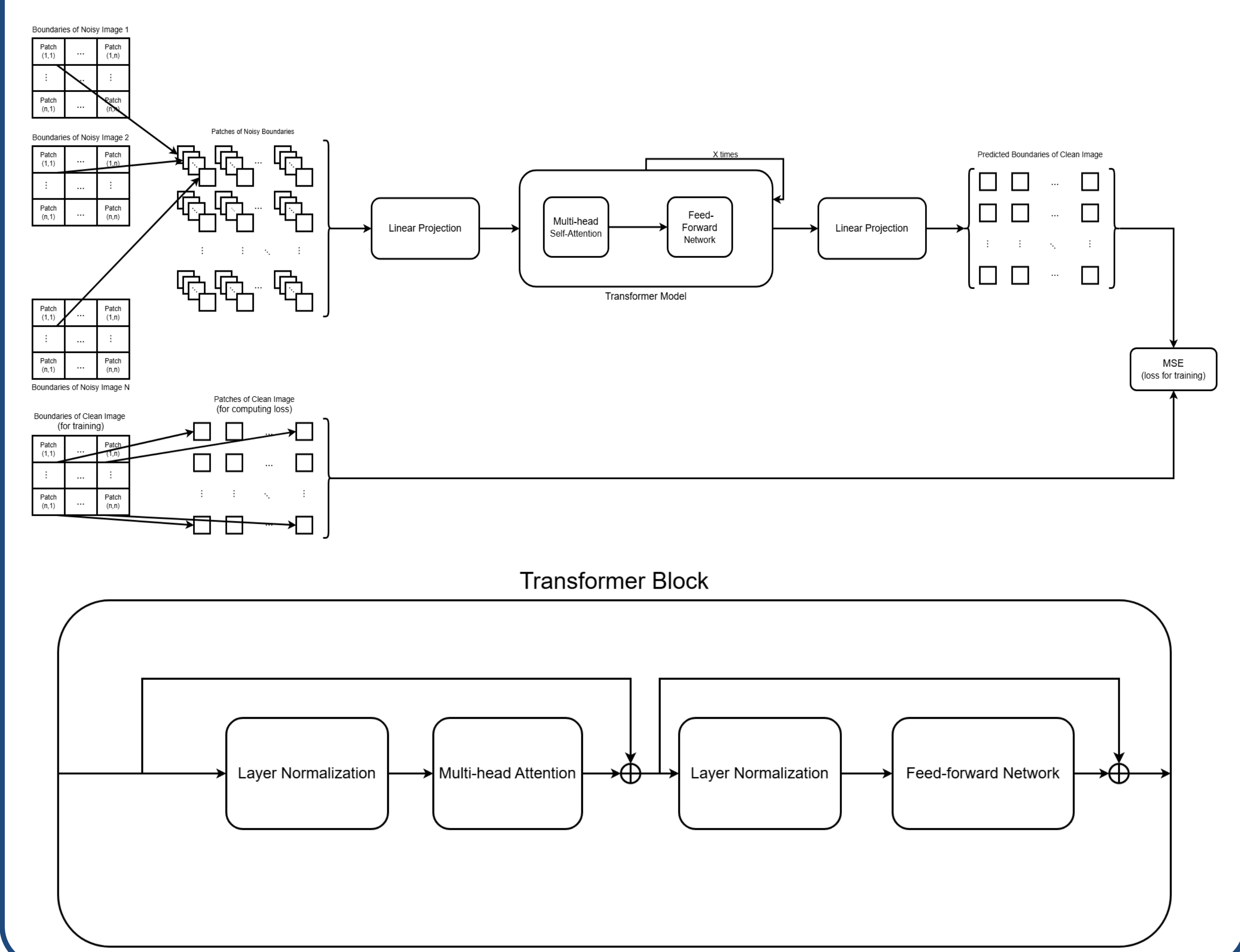
Data



References

- [1] Polansky, Herrmann, Hur, Sun, Verbin, Zickler. Boundary attention: Learning curves, corners, junctions and grouping, 2024.
[2] Verbin, Zickler. Field of junctions: Extracting boundary structure at low SNR, Oct 2021.
[3] Moreels, P., & Perona, P. (2022). Caltech Home Objects 2006 CaltechDATA.
[4] Intel, Image Classification Challenge.

Method and Model



Experimental Results

Original Image	Noisy Boundaries 1	Noisy Boundaries 2	Noisy Boundaries 3	Noisy Boundaries 4	Noisy Boundaries 5	Clean Boundaries	Model Output
PSNR	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	
Avg Input	15.98	15.38	14.43	14.36	14.81	14.46	
Max Input	16.17	15.76	14.71	14.64	15.09	14.73	
Output	17.95	17.26	16.48	16.43	17.15	16.35	

Gaussian Noise level	Avg Input PSNR	Avg Output PSNR	Percent Increase
0.3	15.51	17.68	13.98%
0.4	15.15	17.37	14.65%
0.5	14.81	16.32	10.20%