CVPR Nashville JUNE 11-15, 2025

TailedCore: Few-Shot Sampling for Unsupervised Long-Tail Noisy Anomaly Detection

Yoon Gyo Jung, Jaewoo Park, Jaeho Yoon, Kuan-Chuan Peng, Wonchul Kim, Andrew Beng Jin Teoh, Octavia Camps

Northeastern University

AiV

YONSEI UNIVERSITY

MITSUBISHI ELECTRIC

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

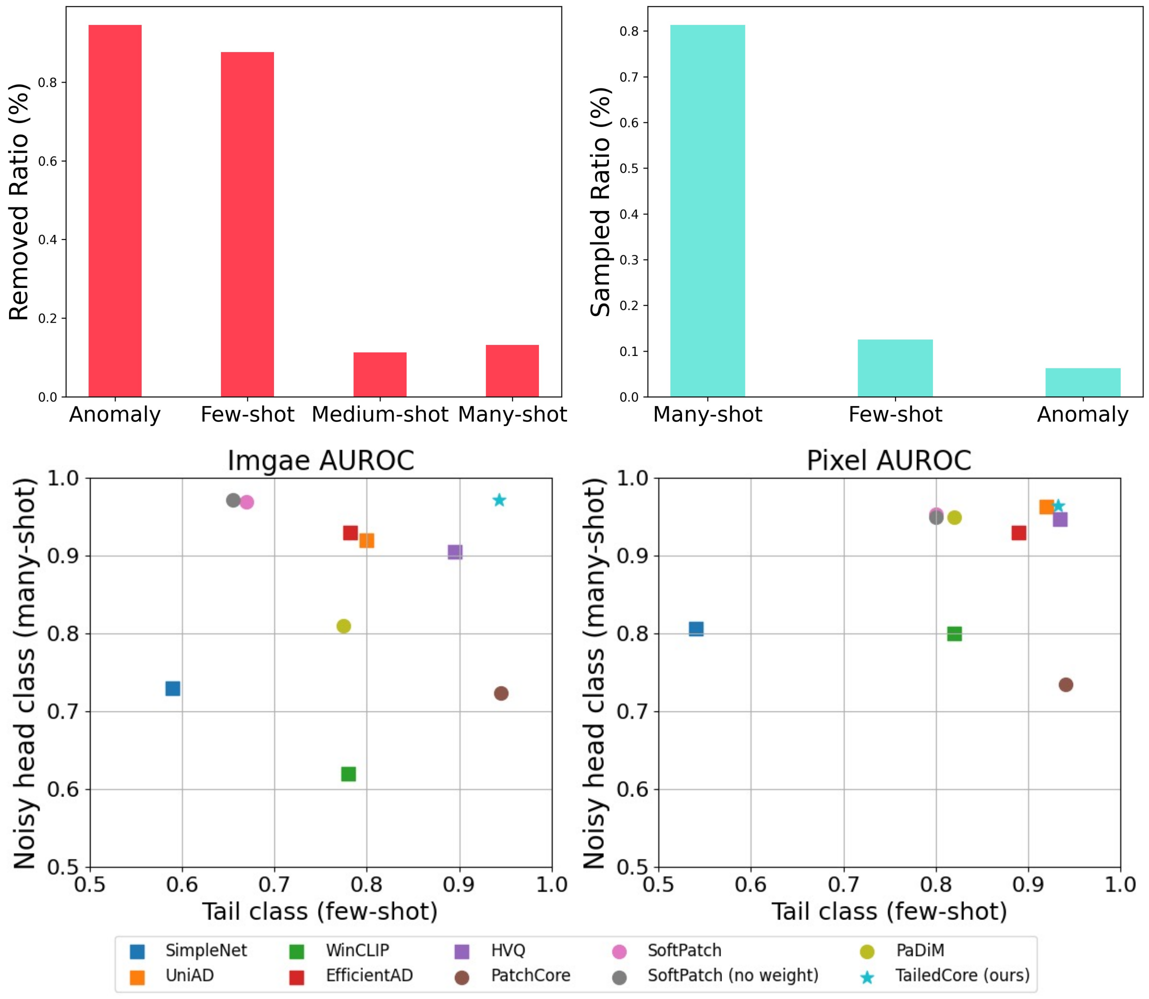
TailedCore: Few-Shot Sampling for Unsupervised Long-Tail Noisy Anomaly Detection

Topic:

- We address unsupervised noisy long-tailed anomaly detection which is much more challenging than solving each individually. Normal data is both contaminated with defective regions and the product class distribution is tailed and unknown.
- Previous methods address either noisy/contaminated unsupervised anomaly detection or long-tailed unsupervised anomaly detection, but none of them have focused these together at once, namely noisy long tailed anomaly detection.
- Setup : only head class is contaminated with noise and tail class (<20 samples) exsits.

Motivation :

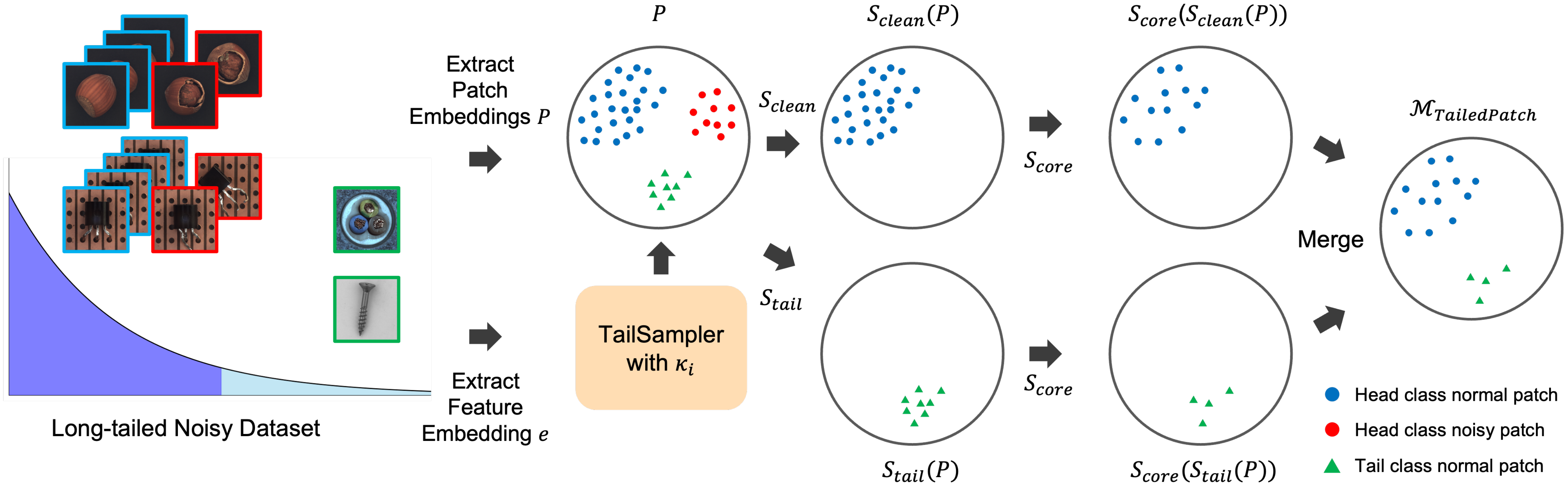
- Tail-versus-noise trade off :
  1) **Noise discriminative models**, such as SoftPatch removes statistically minor patches assuming less frequent data is noise. However, this accidently also **removes tail classes** as shown in the figure above (red bar).
  2) **Greedy sampling** used in patchcore samples tail classes well due to the nature of greedy sampling, however, **also favors noisy patches** as well as shown in the figure above (green bar)



TailedCore: Few-Shot Sampling for Unsupervised Long-Tail Noisy Anomaly Detection

Solution :

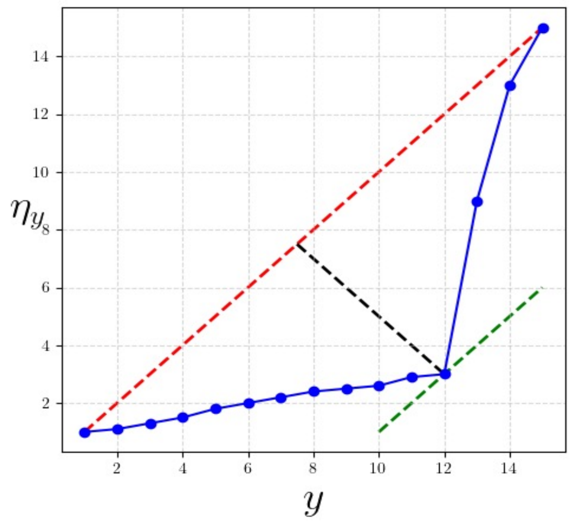
- TailSampler : Selectively sample long-tail class samples while not sampling noise samples by using the global average pooled features. Global features are less affected by anomalies(noise) which are mostly local attributes.
- Denoise with existing noise discriminative methods (e.g. SoftPatch) with $S_{clean}(P)$
- Collect patch features $S_{tail}(P)$ from TailSampler and merge with denoised patches



TailedCore: Few-Shot Sampling for Unsupervised Long-Tail Noisy Anomaly Detection

TailSampler:

- Sort out the long-tail classes by estimating the size of classes from each samples.
- Given percentile p, we estimate the neighbors of embedding e_i, $H_i = \{e \in Z: \measuredangle(e_i, e) \leq m_i/2\}$ for every e_i, where $m_i := \max_{e \in Z} \measuredangle(e_i, e)$, and Z is the set of all embeddings. The adaptive angle is defined to contain p-th percentile of the half-angle region
$$\alpha_i = \measuredangle\left(e_i, e_{p \cdot |H_i|}\right)$$
sorted in increasing order.
- With each of the α_i, we estimate its class size based on neighborhoods of neighborhoods where $N_\alpha(e_i) = \{e \in Z: \measuredangle(e_i, e) < \alpha\}$ denote the neighborhood of e_i, which is the set of all train embedding e within angle α of e_i. Then class size is estimated by the mode of
$$\kappa_i = \underset{e \in N_{\alpha_i}(e_i)}{\mathrm{mode}} \left(\left|N_{\alpha(e)}(e)\right|\right)$$
where $\alpha(e)$ is the adaptive angle with respect to embedding e belonging to the neighborhood $N_{\alpha_i}(e_i)$ of embedding e_i.
- After estimating class sizes of each samples κ_i, estimate size of each classes $\eta_y \approx |C_y|$ inductively by
$$\eta_1 = round\left(\frac{1}{\kappa_{(1)}} \sum_{i=1}^{\kappa_{(1)}} \kappa_{(i)}\right), \dots, \eta_{(y+1)} = round\left(\frac{1}{\kappa_{\eta_{(y+1)}},} \sum_{i=\eta_y+1}^{\min(\kappa_{\eta_{(y+1)}}, |X|)} \kappa_{(i)}\right)$$
- And determine maximum size of tail classes with elbow technique where η_i abruptly changes.



TailedCore: Few-Shot Sampling for Unsupervised Long-Tail Noisy Anomaly Detection

Results

tail type	Pareto			step ($K=4$)			step ($K=1$)		
class type	C_t	C_h	all	C_t	C_h	all	C_t	C_h	all
PaDiM [9] ICPR'21	82.45	80.95	82.06	77.47	81.28	79.19	71.54	81.75	75.63
HVQ [26] NeurIPS'23	83.46	80.23	82.99	82.01	85.50	83.56	74.15	90.15	80.55
WinCLIP [19] CVPR'23	89.35	90.11	90.37	91.60	88.21	90.37	91.80	88.23	90.37
AnomalyCLIP [43] ICLR'24	90.93	90.98	91.48	91.82	90.83	91.48	91.21	91.90	91.48
PatchCore [34] CVPR'22	93.33	87.59	89.18	92.19	71.18	83.83	86.36	70.48	80.01
SoftPatch [20] NeurIPS'22	84.68	86.95	87.71	67.65	**97.54**	79.64	60.66	**97.49**	75.40
TailedCore **(ours)**	**96.55**	**95.24**	**96.12**	**95.82**	95.34	**95.71**	**93.54**	95.77	**94.43**

Table 1. Anomaly classification on MVTecAD with image-level AUROC (%). We report the mean over 5 random seeds for each measurement. Notations: C_h / C_t: head / tail classes.

tail type	Pareto			step ($K=4$)			step ($K=1$)		
class type	C_t	C_h	all	C_t	C_h	all	C_t	C_h	all
PaDiM [9] ICPR'21	90.11	92.66	91.43	82.53	95.29	87.67	78.80	95.54	85.50
HVQ [26] NeurIPS'23	93.63	86.85	90.55	90.73	92.58	91.53	86.36	95.20	89.90
WinCLIP [19] CVPR'23	82.03	84.06	82.29	80.60	84.63	82.29	80.16	85.48	82.29
AnomalyCLIP [43] ICLR'24	91.24	91.69	91.08	89.96	92.66	91.08	89.34	93.68	91.08
PatchCore [34] CVPR'22	93.56	87.98	89.93	93.54	72.09	85.19	92.02	71.35	83.75
SoftPatch [20] NeurIPS'22	92.19	93.83	93.41	80.98	**96.49**	87.24	70.34	**96.89**	80.99
TailedCore **(ours)**	**96.08**	**95.01**	**95.29**	**95.56**	93.20	**94.74**	**94.19**	93.70	**93.99**

Table 3. Anomaly segmentation on MVTecAD with pixel-level AUROC (%). We report the mean over 5 random seeds for each measurement. Notations: C_h / C_t: head / tail classes.

tail type	Pareto			step ($K=4$)			step ($K=1$)		
class type	C_t	C_h	all	C_t	C_h	all	C_t	C_h	all
PaDiM [9] ICPR'21	70.70	83.35	78.64	60.65	88.93	72.43	55.98	86.75	68.80
HVQ [26] NeurIPS'23	73.47	84.03	68.25	68.25	89.30	77.02	61.57	80.40	69.42
WinCLIP [19] CVPR'23	73.25	76.92	75.47	75.98	74.76	75.47	78.80	70.80	75.47
AnomalyCLIP [43] ICLR'24	81.96	82.48	82.05	82.28	81.74	82.05	**83.26**	80.34	82.05
PatchCore [34] CVPR'22	86.11	85.73	85.59	83.53	67.51	76.85	79.33	68.56	74.84
SoftPatch [20] NeurIPS'22	78.04	92.16	86.56	59.70	**95.97**	74.81	52.61	**94.17**	69.92
TailedCore **(ours)**	**87.55**	**93.06**	**90.85**	**85.16**	95.91	**89.64**	82.97	94.11	**87.61**

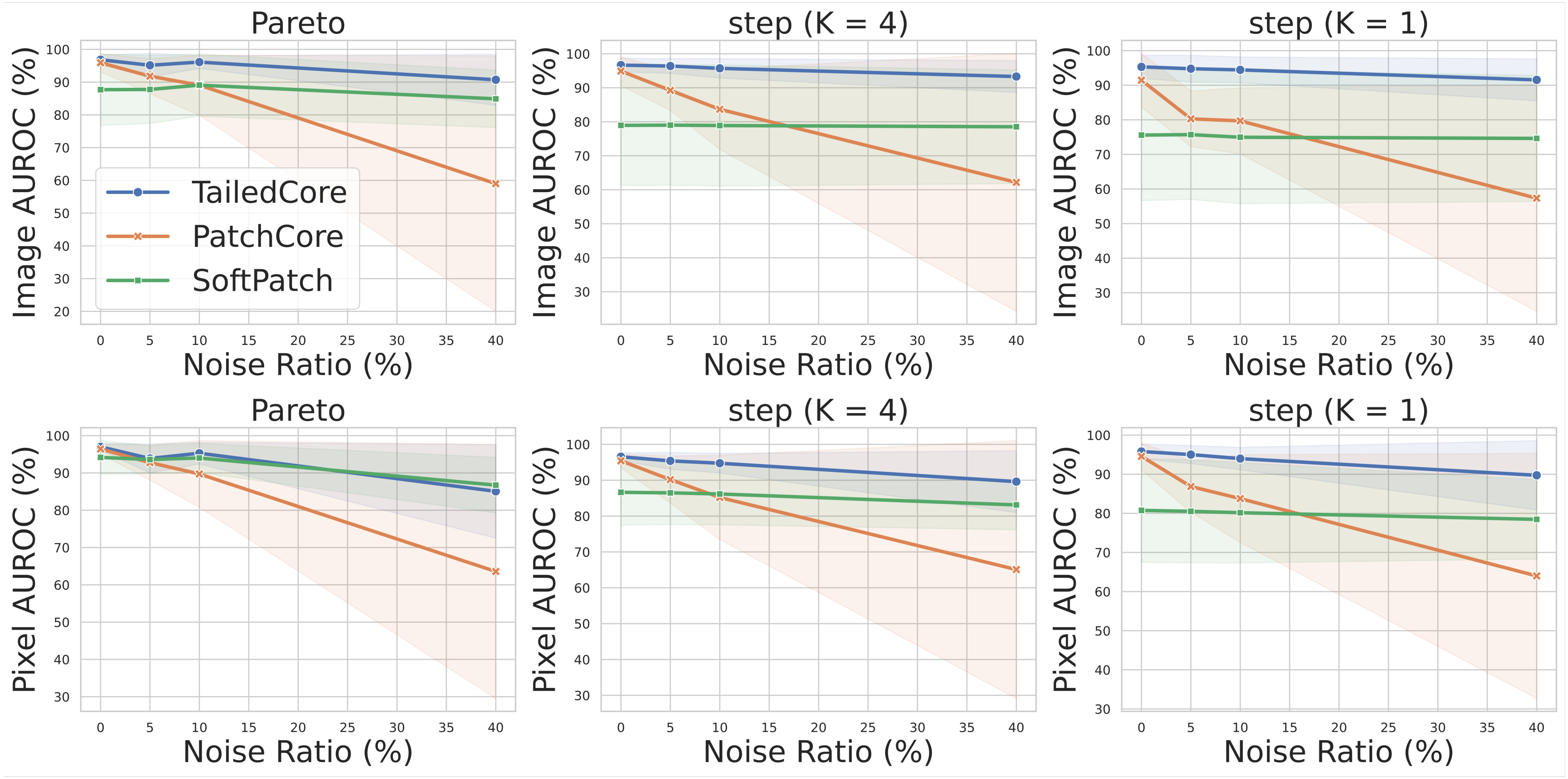
Table 2. Anomaly classification on VisA with image-level AUROC (%). The format and evaluation protocol are the same as Tab. 1.

tail type	Pareto			step ($K=4$)			step ($K=1$)		
class type	C_t	C_h	all	C_t	C_h	all	C_t	C_h	all
PaDiM [9] ICPR'21	89.02	95.10	82.81	83.90	97.36	89.51	82.57	96.57	88.40
HVQ [26] NeurIPS'23	95.27	**97.60**	96.71	93.88	**98.34**	95.74	90.58	95.51	92.63
WinCLIP [19] CVPR'23	71.94	73.97	73.19	74.60	71.21	73.19	73.81	72.32	73.19
AnomalyCLIP [43] ICLR'24	95.60	95.46	95.51	95.54	95.48	95.51	**96.16**	94.60	95.51
PatchCore [34] CVPR'22	96.84	87.99	91.13	95.39	62.96	81.88	94.11	65.30	82.10
SoftPatch [20] NeurIPS'22	93.20	96.74	95.27	83.95	97.10	89.43	80.73	96.82	87.43
TailedCore **(ours)**	**97.98**	97.25	**97.48**	**96.80**	97.02	**96.89**	96.12	**97.39**	**96.65**

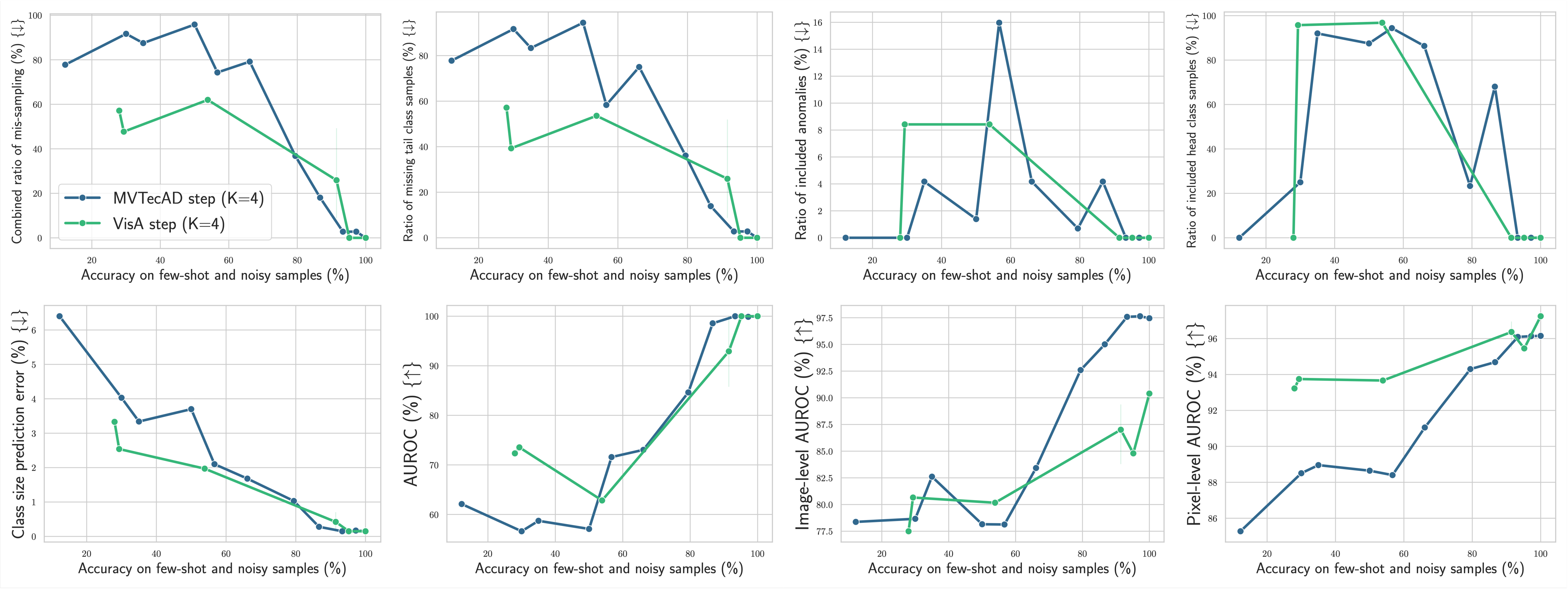
Table 4. Anomaly segmentation on VisA with pixel-level AUROC (%). The format and evaluation protocol are the same as Tab. 3.

TailedCore: Few-Shot Sampling for Unsupervised Long-Tail Noisy Anomaly Detection

Ablation



TailedCore: Few-Shot Sampling for Unsupervised Long-Tail Noisy Anomaly Detection



TailedCore: Few-Shot Sampling for Unsupervised Long-Tail Noisy Anomaly Detection

Segmentation Quality