A. Whitening Details

As $\hat{\Sigma}$ is a real symmetric matrix, it can be decomposed as $\hat{\Sigma} = Q\Lambda Q^T$, where Q is an orthogonal matrix with column q_i being the i-th eigenvector of $\hat{\Sigma}$, and Λ is the diagonal matrix with the element $\Lambda_{ii} = \lambda_i$ being the i-th eigenvalue of $\hat{\Sigma}$. Since the regularization of $\hat{\Sigma}$ ensures that its eigenvalues are real and positive, the whitening matrix $\Lambda^{-\frac{1}{2}}Q^T$, where the inverse square root is taken elementwise (because Λ is diagonal), can be used to build white feature vectors

$$w = \left(\Lambda^{-\frac{1}{2}}Q^T\right)(x - \hat{\mu}) \quad , \tag{6}$$

such that the linear projection on the left is $(\hat{\Sigma}^{-1})^{\frac{1}{2}}$, the square root matrix (*not* elementwise) of the empirical precision matrix (inverse of the empirical covariance matrix):

$$\left(\Lambda^{-\frac{1}{2}}Q^T\right)^2 = \left(\Lambda^{-\frac{1}{2}}Q^T\right)\left(\Lambda^{-\frac{1}{2}}Q^T\right) \tag{7a}$$

$$= Q\left(\Lambda^{-\frac{1}{2}}\Lambda^{-\frac{1}{2}}\right)Q^T \tag{7b}$$

$$= Q\Lambda^{-1}Q^T = \hat{\Sigma}^{-1} \tag{7c}$$

$$\therefore \left(\Lambda^{-\frac{1}{2}}Q^T\right) = (\hat{\Sigma}^{-1})^{\frac{1}{2}} \quad . \tag{8}$$

Therefore the Mahalanobis distance (Equation 1) can be expressed as $\| w \|_2$:

$$D_M(x)^2 = (x - \hat{\mu})^T \left[(\hat{\Sigma}^{-1})^{\frac{1}{2}}(\hat{\Sigma}^{-1})^{\frac{1}{2}}\right](x - \hat{\mu}) \tag{9a}$$

$$= \left[(\hat{\Sigma}^{-1})^{\frac{1}{2}}(x - \hat{\mu})\right]^T \left[(\hat{\Sigma}^{-1})^{\frac{1}{2}}(x - \hat{\mu})\right] \tag{9b}$$

$$= w^T w = \| w \|_2^2 \tag{9c}$$

$$\therefore D_M(x) = \| w \|_2 \quad , \tag{10}$$

where $\| \cdot \|_2$ is the Euclidean norm. In short, Equation 6 provides an alternative to compute the anomaly score as

$$D_M(f_N(I)) = \| \Lambda^{-\frac{1}{2}}Q^T (f_N(I) - \hat{\mu}) \|_2 \quad . \tag{11}$$

Notice that the entry w_i (the i-th row of the white vector w) corresponds to the projection of the centered feature vector $(x - \hat{\mu})$ onto the eigenvector q_i from $\hat{\Sigma}$ scaled by $\lambda_i^{-\frac{1}{2}}$. In practice, this step makes it simpler to obtain a reduced model because it can be reproduced by choosing the entries from w.

Assuming eigenvalues λ_i (and respective eigenvectors q_i) to be sorted in increasing order, the entries w_i are equally sorted because the rows of $\left(\Lambda^{-\frac{1}{2}}Q^T\right)$ follow the same order. Therefore a reduced model $\mathbf{Q}' \subseteq \mathbf{Q}$ consists of selecting the entries w_i with the same indexes of the selected eigencomponents in $\mathbf{Q}'$. Example: if $\mathbf{Q}' = \{q_{75}\,,\, q_{11}\,,\, q_{15}\}$, then $\| w' \|_2 = \| (w_{75}\,,\, w_{11}\,,\, w_{15})^T \|_2$.

B. Experiment 1: test set overfit

Figure 6 represents the AUROC performance for component-wise analysis (k vs. AUROC curves), as described in the Section 5. It provides a visual depiction of how individual components within nodes perform across different categories. Each row in the graph represents a category, while each column represents a node.

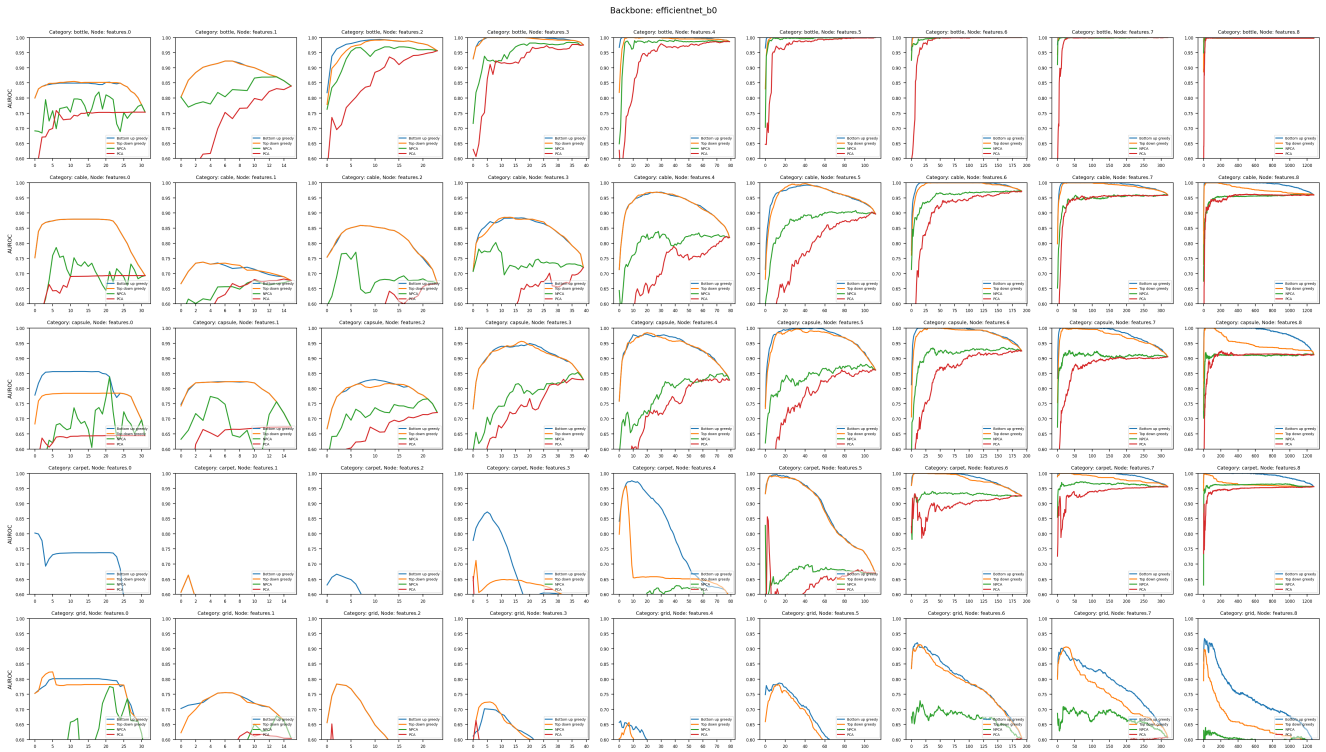
The x-axis of the graph corresponds to the number of components k (i.e. "how much is d reduced to?"). In this experiment, we rank all the eigencomponents of the $\hat{\Sigma}$. The Y-axis indicates the AUROC value, which measures the predictive performance of the components.

The graph includes separate curves that represent different approaches used in the analysis. It shows the AUROC performance of the greedy search for both traversal modes (Bottom-Up and Top-Down) compared to the results obtained from other dimension reduction strategies, including PCA,NPCA, and the supspace "$[\Phi_2, \Phi_3]$" introduced in [10].

Figure 7 is part of the same experiment but the X-axis is restricted to the $k \leq 40$ components (out of d), the Y-axis remains the same with the respective AUROCs. These graphs also include the PCA, NPCA results with a horizontal dashed line marking the highest AUROC for the respective method. We also mark the AUROC performance with no dimension reduction (d components) with a horizontal dashed line to highlight the importance of the component selection.

Exceptional case 1: toothbrush is easier Our analysis revealed an exceptional case in the category of "toothbrush", where all the nodes achieved nearly-maximum performance. This outcome suggests that detecting anomalies in this category is comparatively easier than in other categories. Although, it must be noted that this category has the smallest test set with only 30 anomalous images while others have nearly twice or three times as many, and it only has one anomaly type while others such as "pill", "screw", and "zipper" have between 5 and 7 anomaly types. See Table 1 in Appendix J.

Exceptional case 2: very low-dimensional categories The greedy eigencomponent selection achieves particularly impressive results in categories "wood", "leather", "tile", and "bottle". A perfect score (100% AUROC) is consistently achieved with less than 5 eigencomponents in several nodes.



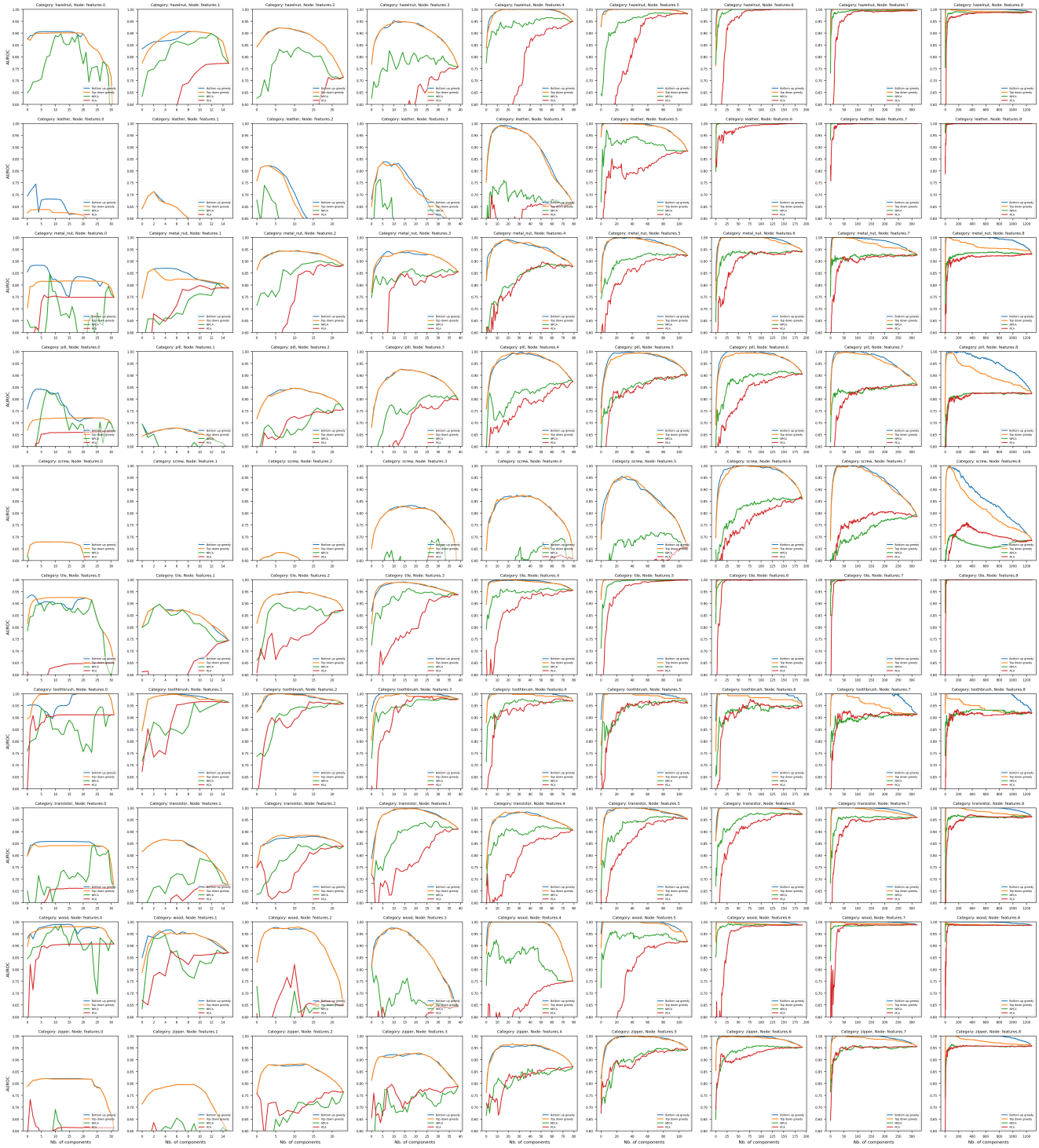
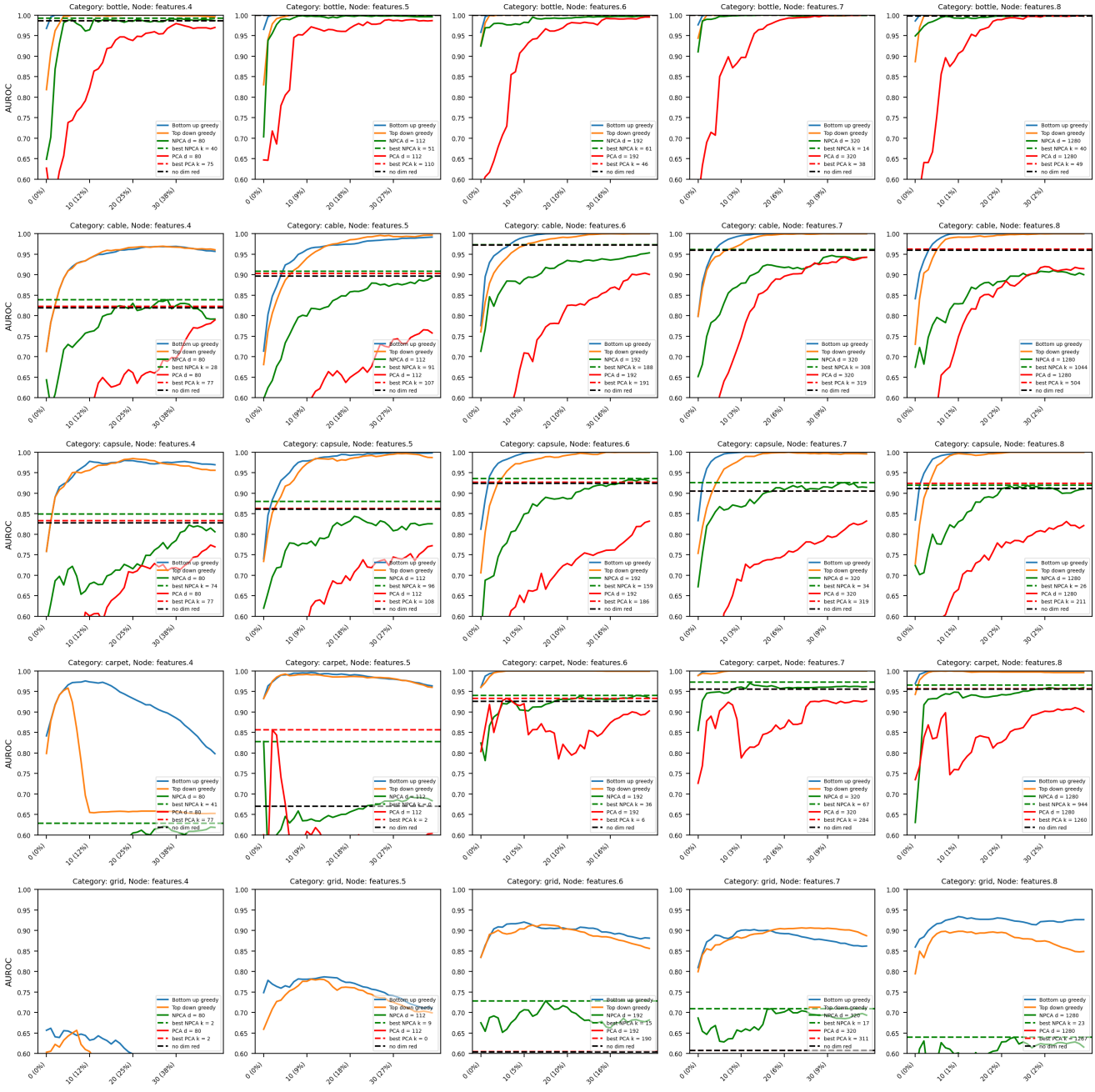
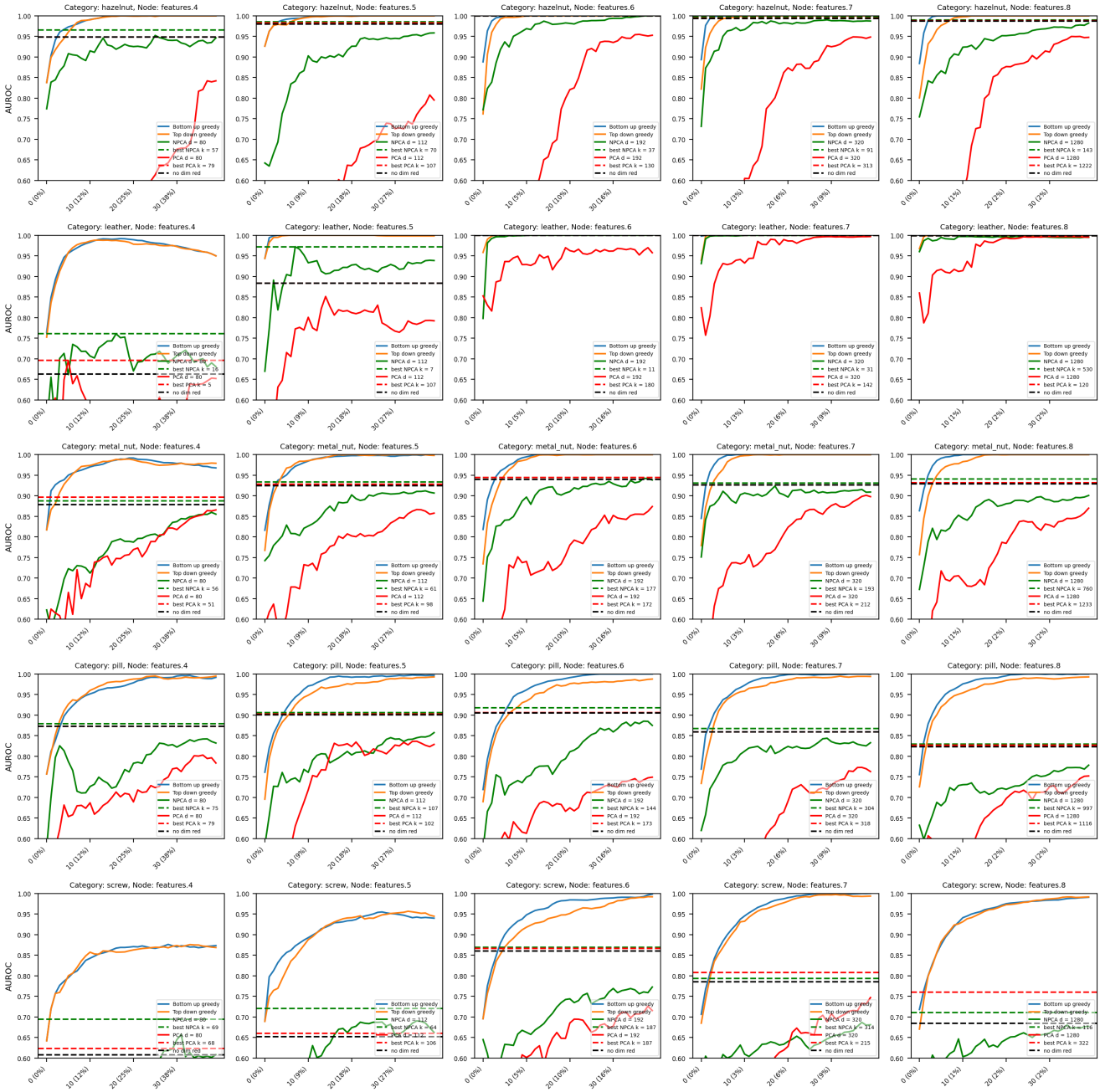


Figure 6: Experiment 1: all plots.

Backbone: efficientnet_b0




Category: hazelnut, Node: features.4
Category: hazelnut, Node: features.5
Category: hazelnut, Node: features.6
Category: hazelnut, Node: features.7
Category: hazelnut, Node: features.8
Category: leather, Node: features.4
Category: leather, Node: features.5
Category: leather, Node: features.6
Category: leather, Node: features.7
Category: leather, Node: features.8
Category: metal_nut, Node: features.4
Category: metal_nut, Node: features.5
Category: metal_nut, Node: features.6
Category: metal_nut, Node: features.7
Category: metal_nut, Node: features.8
Category: pill, Node: features.4
Category: pill, Node: features.5
Category: pill, Node: features.6
Category: pill, Node: features.7
Category: pill, Node: features.8
Category: screw, Node: features.4
Category: screw, Node: features.5
Category: screw, Node: features.6
Category: screw, Node: features.7
Category: screw, Node: features.8
AUROC
Bottom up greedy
Top down greedy
0 (0%)
10 (12%)
20 (25%)
30 (38%)
10 (9%)
20 (18%)
30 (27%)
10 (5%)
20 (10%)
30 (16%)
10 (3%)
20 (6%)
30 (9%)
10 (1%)
20 (2%)
30 (2%)

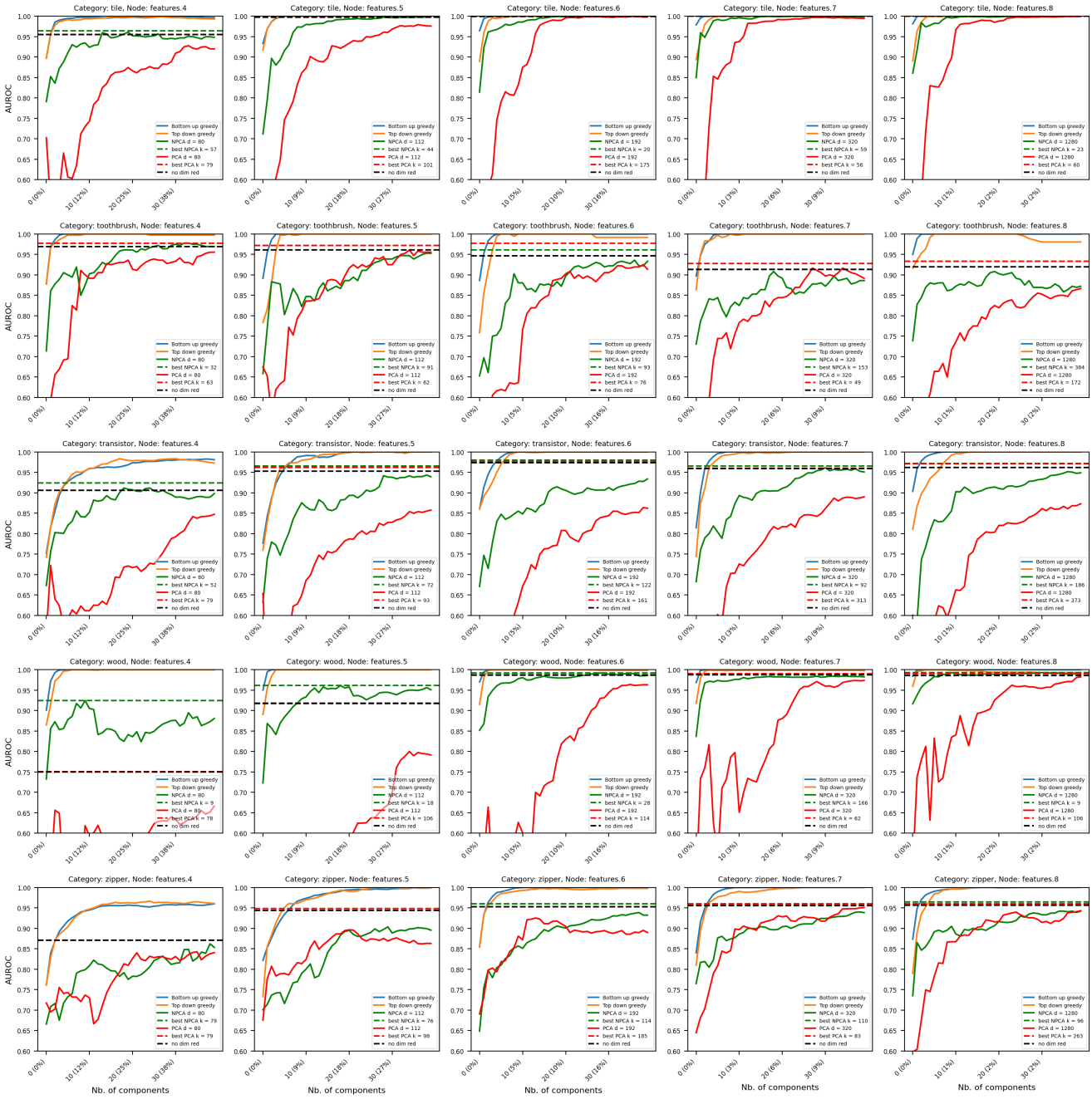


Figure 7: Experiment 1: all plots. Zoom on the X-axis $k \leq 40$ components.

C. Experiment 1: A Comparative Analysis with Other Methods

Figure 8 shows the best AUROC (Y-axis) out of all values of k per node N (X-axis shows the node index) for our proposed GreedyES - Bottom-Up approach (Experiment 1, Sec. 5) and the scenario without any dimension reduction. It compares the AUROC of other dimensionality reduction techniques, demonstrating their effectiveness when combined with various backbone architectures.

PatchCore with Wide ResNet-50-2 Backbone: Represented by a dashed blue line, PatchCore [14] employs a Wide ResNet-50-2 backbone. A notable feature of this method is its use of a memory bank subsampling strategy, our plot shows the one which samples at a rate of 25%.

Gaussian-AD* with EfficientNetB4 Backbone: Our subfigures also include two dashed lines (orange and green) corresponding to Gaussian-AD in [12, 13]. Both of these methods utilize the EfficientNetB4, that offers high AD results at medium complexity.

- **NPCA 1% (Orange Line):** This line illustrates the performance when utilizing the NPCA 1% of retained variance. NPCA method focuses on retaining those principal components that account for the least variance in the normal data, offering a unique way to reduce dimensionality
- **No-Dimensional Reduction (Green Line):** The green dashed line shows the results when there is no dimensionality reduction applied. This serves as a baseline to understand the raw capabilities of the EfficientNetB4 backbone without any reduction techniques interfering.

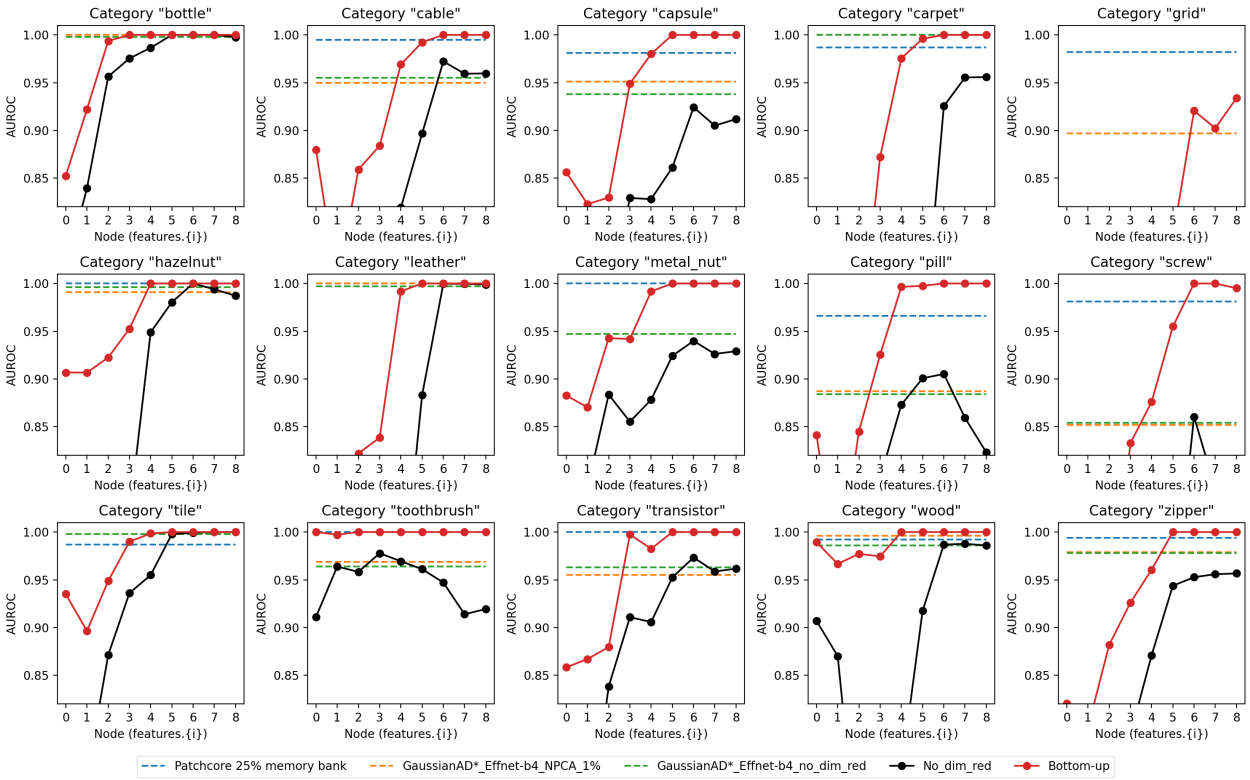


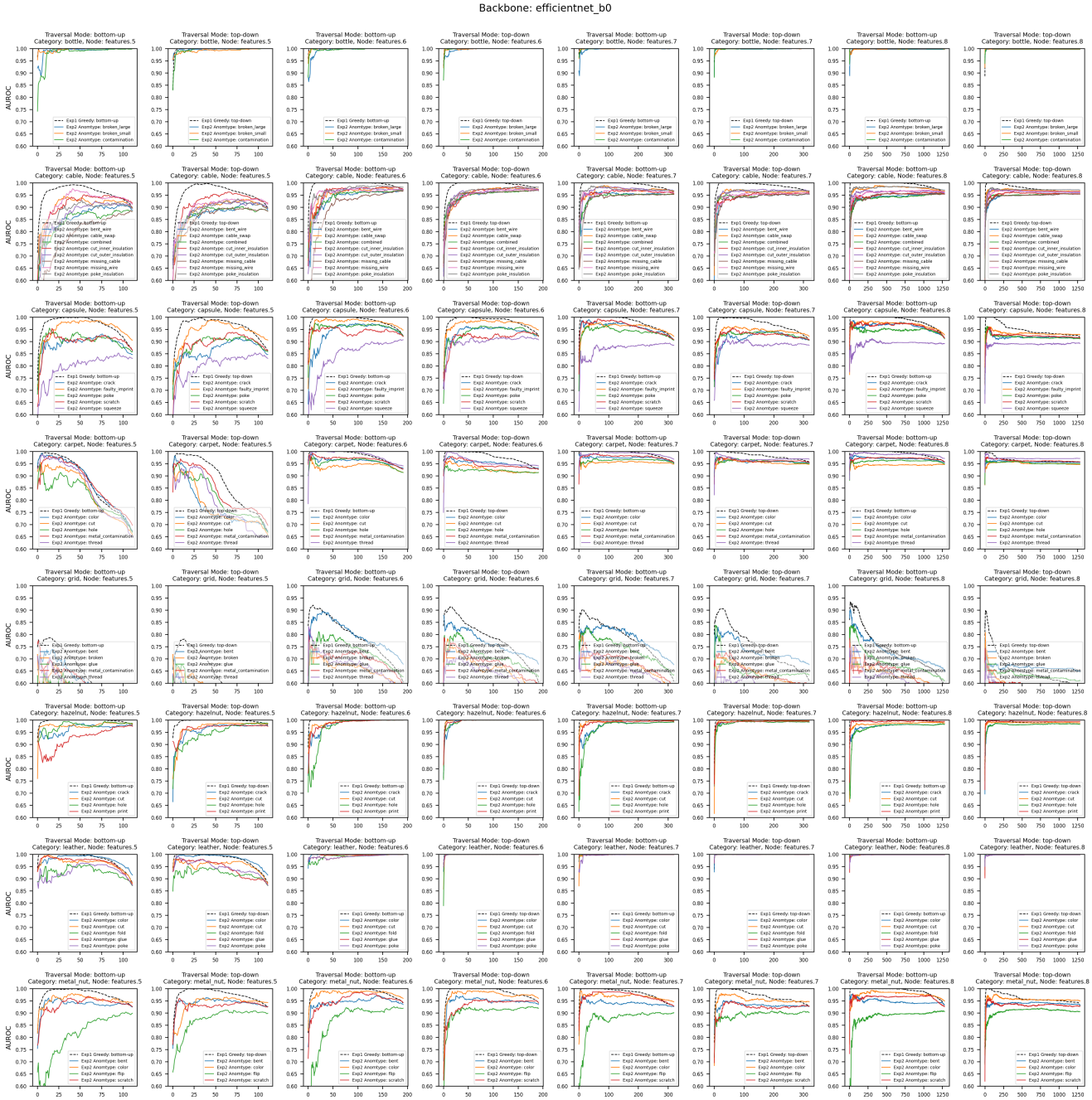
Figure 8: Experiment 1: best AUROC out of all values of k per node for both Bottom-Up and no-dimensional reduction scenarios. The dashed blue line corresponds to the results of PatchCore algorithm from [14]. The dashed orange and green lines refer to the results from [12, 13], using Gaussian-AD* EfficientNetB4 as a backbone, the orange line focuses on NPCA method of 1%, and the green one shows the result with no-dimensional reduction.

D. Experiment 2: per anomaly type

Figure 9 shows all the scenarios, as discussed in Section 6, of the results obtained from Experiment 2. These plots grant critical insights into the performance of the various anomaly types, along with the influence of the component selection strategy on the generalization process. Figure 9 also includes a juxtaposition of the results from Experiment 2 along with the results from Experiment 1 for reference.

The line plots illustrate the number of eigencomponents on the X-axis and the corresponding AUROC values on the Y-axis. Notice that each curve corresponds to a different $\mathbf{W}_{\text{greedy}}$ / $\mathbf{W}_{\text{eval}}$ splits, so they do not match the same performance at the point $k = d$.

Notice that category "toothbrush" is not in the results of this experiment as it only has one anomaly type.



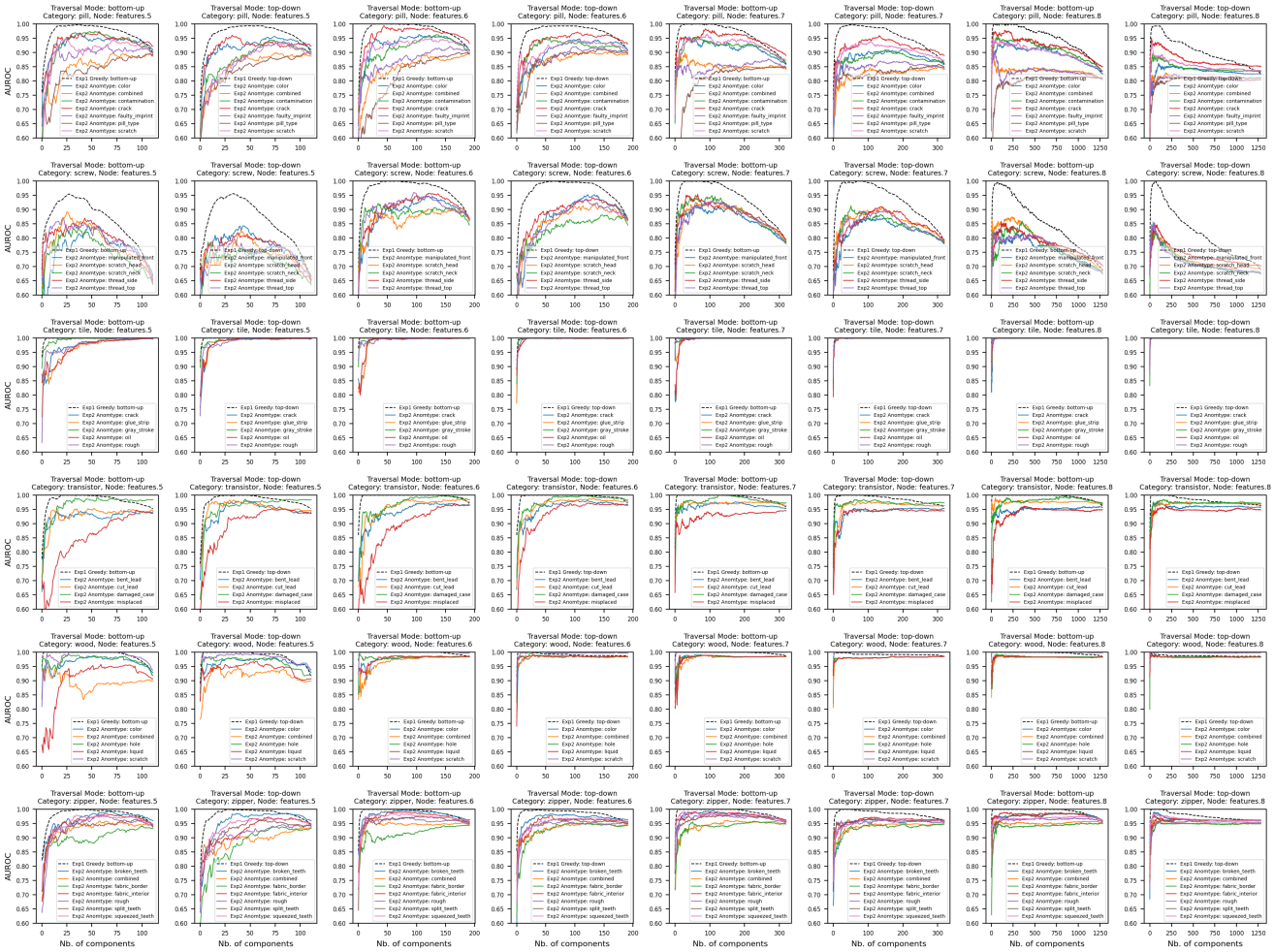
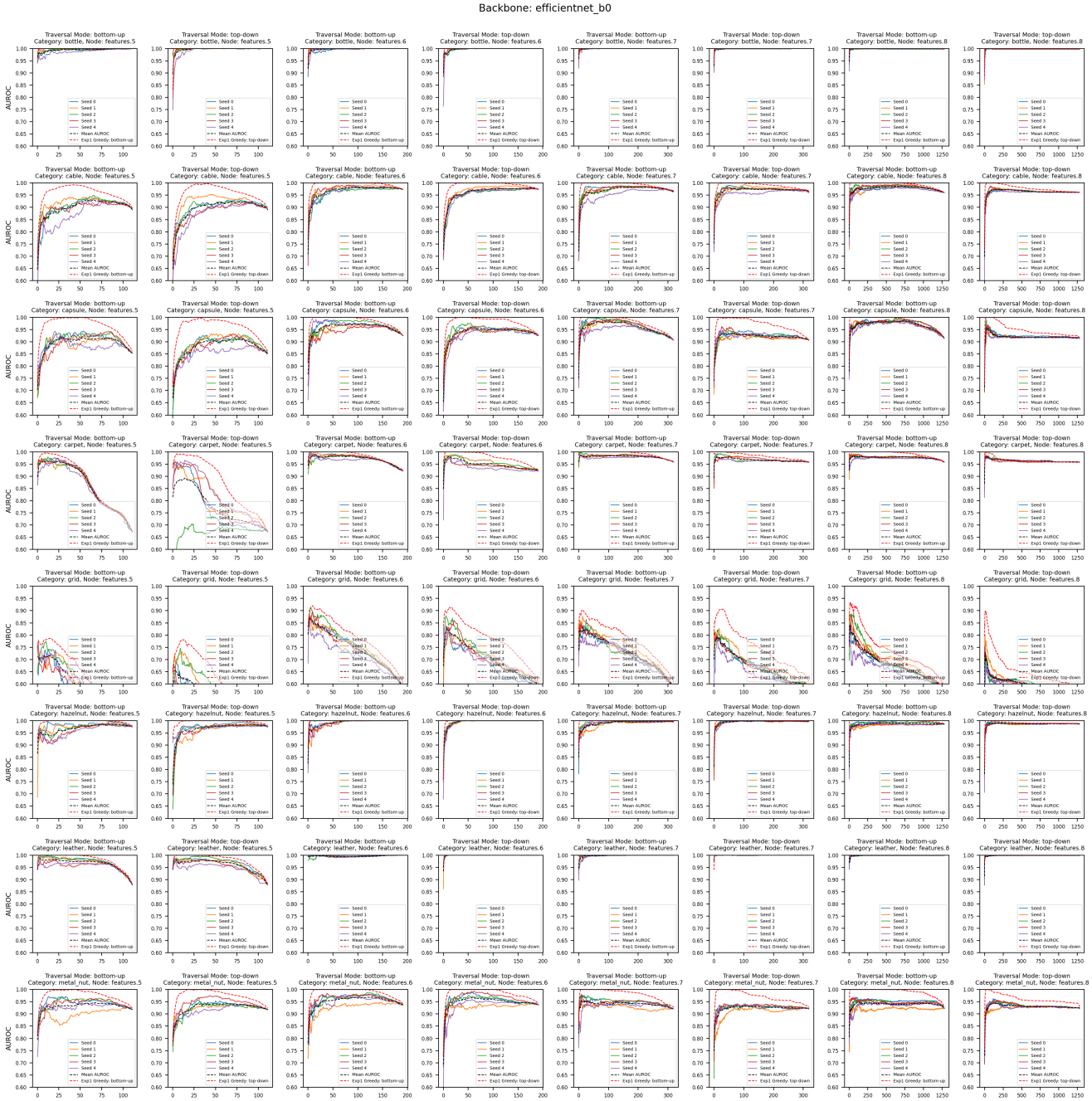


Figure 9: Experiment 2: all plots.

E. Experiment 3: fixed number of images

Figure 10 shows all the scenarios, as discussed in Section 7, of the results obtained from Experiment 3. The figure also includes a comparison of the results from Experiment 3 with both Bottom-Up and Top-Down traversal modes employed in Experiment 1. The line plots in the figure depict the number of eigencomponents on the X-axis and the corresponding AUROC values on the Y-axis.

These plots show the performance of the various seeds along with their (cross-seed) mean performance and the curve from Experiment 1 for reference, providing valuable insights into the generalization capacity of the greedy eigencomponent selection.



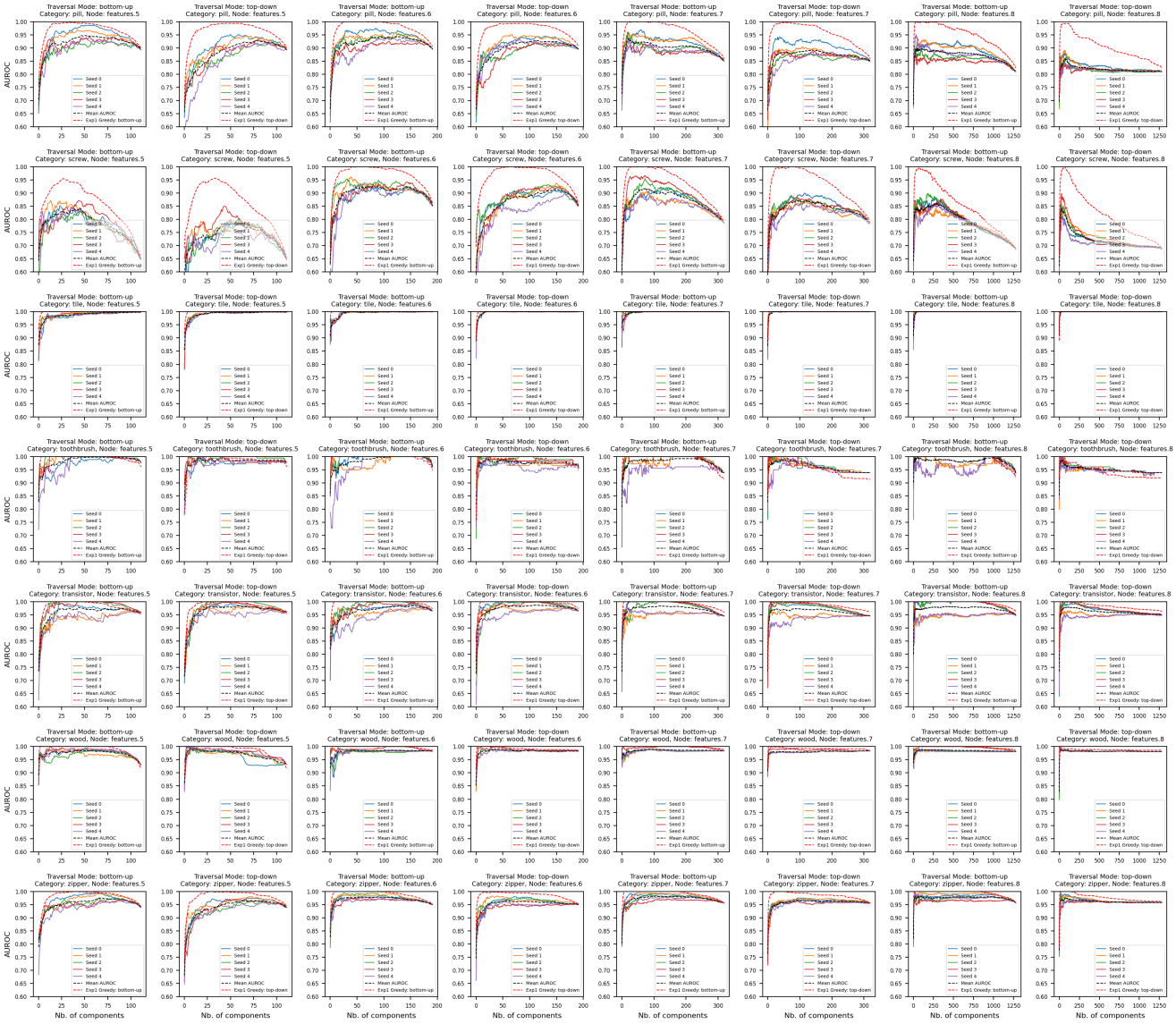


Figure 10: Experiment 3: all plots.

F. Regimes

Figure 11 is derived from the results of Experiment 1, where we distinguish three main regimes from the greedy eigencomponent selection of the Bottom-Up traversal mode. We designate the "Rise" in blue, the "Plateau" in red, and the "Drop" in green. The use of distinct colors provides a better visualisation and differentiation of these three regimes.

We selected representative cases and based this choice on a certain criteria: we included cases were AUROC reaches the score of 1, and deliberately focused on showing only $\geq$ `f5` because starting from this node we can clearly observe different regimes and because they achive the best performance. We also excluded categories such as "screw" and "grid" as their results didn't align with our regime analysis, meaning the results made it hard to distinguish these three regimes.

The Rise regime (blue): This phase represents the initial selection of eigencomponents where each new addition significantly boosts the performance.

The Plateau regime (red): In this phase the addition of more eigencomponents doesn't significantly improve or degrade the performance. In fact, for most cases with only several exceptions we observe almost all the time a 100% AUROC.

The Drop regime (green): Finally, the 'Drop' regime represents the phase where adding more eigencomponents starts to degrade the performance. This can happen due to the incorporation of noisy, irrelevant, or redundant components which don't contribute positively and are simply bad components.

The most interesting cases would be "cable", "capsule", "pill", "transistor", and "zipper" with the nodes `f6` and `f7` because we can see clearly all the three regimes and utilize its eigencomponents for further analysis mentioned in Sections G and H.

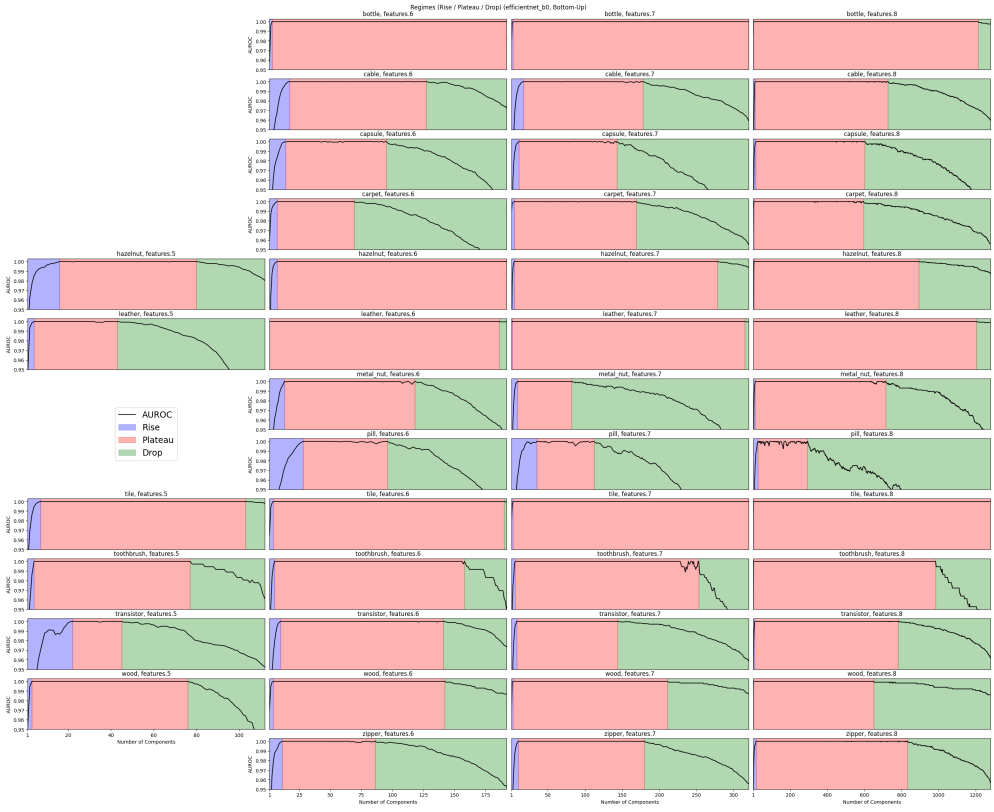


Figure 11: Experiment 1: eigencomponents regimes.

G. Minimal Number of Dimensions at Maximum AUROC

Figure 12 shows an analysis of the optimal dimension reduction size for all the scenarios in Experiment 1 (Section 5). We select minimal number of dimensions k such that its corresponding AUROC is maximal – it corresponds to the left most point of the plateau regime (see Figure 11 in the Appendix F).

The X-axis corresponds to the node depth in EfficientNetB0 and the Y-axis shows its corresponding optimal number of components k (dots, scaled on the left) and the dashed line (scaled on the right) shows the original feature vector size d. The marker color represent a point's corresponding AUROC scaled from 0.90 to 1.00 (light blue to pink).

The optimal number of components (left y-axis) of high-performance models (pink markers) has a negative trend relative to the node depth (deeper nodes implicate less components), while the original embedding size (right y-axis) is bigger. In other words, higher dimensional embeddings tend to be capable of encoding the normality of the images in lower dimensional subspaces.

Figure 12: Experiment 1: minimal number of (reduced) dimensions k at maximum AUROC.

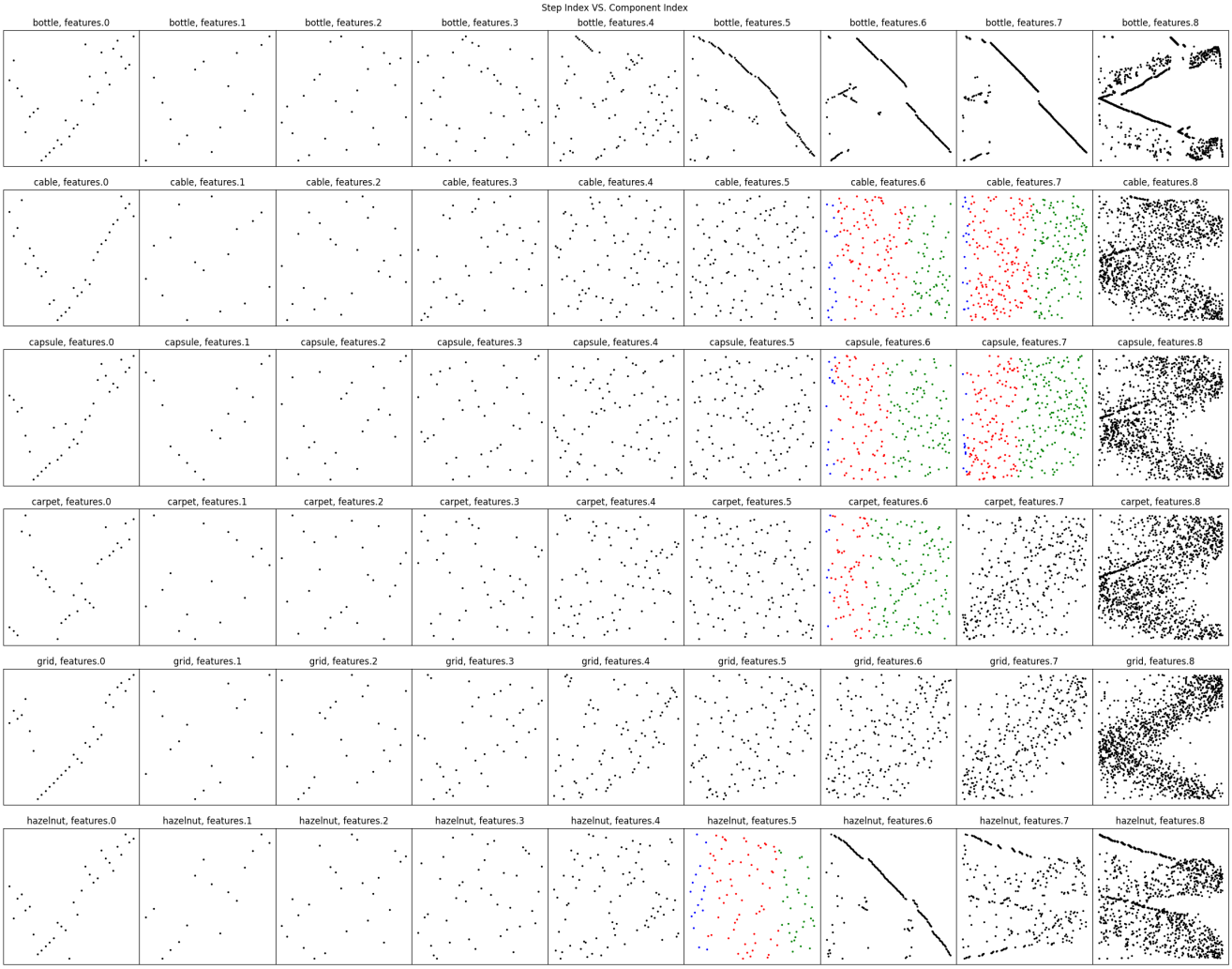
H. Are the best components from the smallest or largest eigenvalues? Both.

We analyse if the order that components show up in the Bottom-Up component selection relates to PCA or NPCA. Figure 13 shows the step index vs. the component index of all the runs with EfficientNetB0 using the Bottom-Up strategy.

The x-axis is the step index, which corresponds to the depth of the search tree in greedy, so it represents, from left to right, the order that the eigencomponents were added. The y-axis is the component index, which corresponds to the eigenvalues order, so it represents, from bottom to top, the smallest to highest eigenvalues. PCA's graph would be a line with slope -1 (first component is the largest, last component is the smallest), and NPCA's would be a line with slope 1 (first component is the smallest, last component is the largest).

The most relevant nodes are from `f5` to `f8` because they achieve the best performances (see Figures 6 and 3). The most relevant components are generally the first 10 to 30 ones (left most part of each plot; see Figure 12) because that generally corresponds to the "rise" regime (see Section F). The cases with most visible regimes are plotted with the same regime colors used in Figure 11.

Despite some exceptionally structured cases like (some nodes from) categories bottle and tile, which are similar to the PCA component sorting, most plots are disordered and seemingly random. This shows there is no relation between eigenvalue magnitude (i.e. the amount of variance encoded in an eigencomponent) and utility for AD, which was implicitely assumed in previous works using PCA, NPCA, and in [10]. Notice this lack of entanglement between the two is particularly visible at the first (therefore most useful) components.



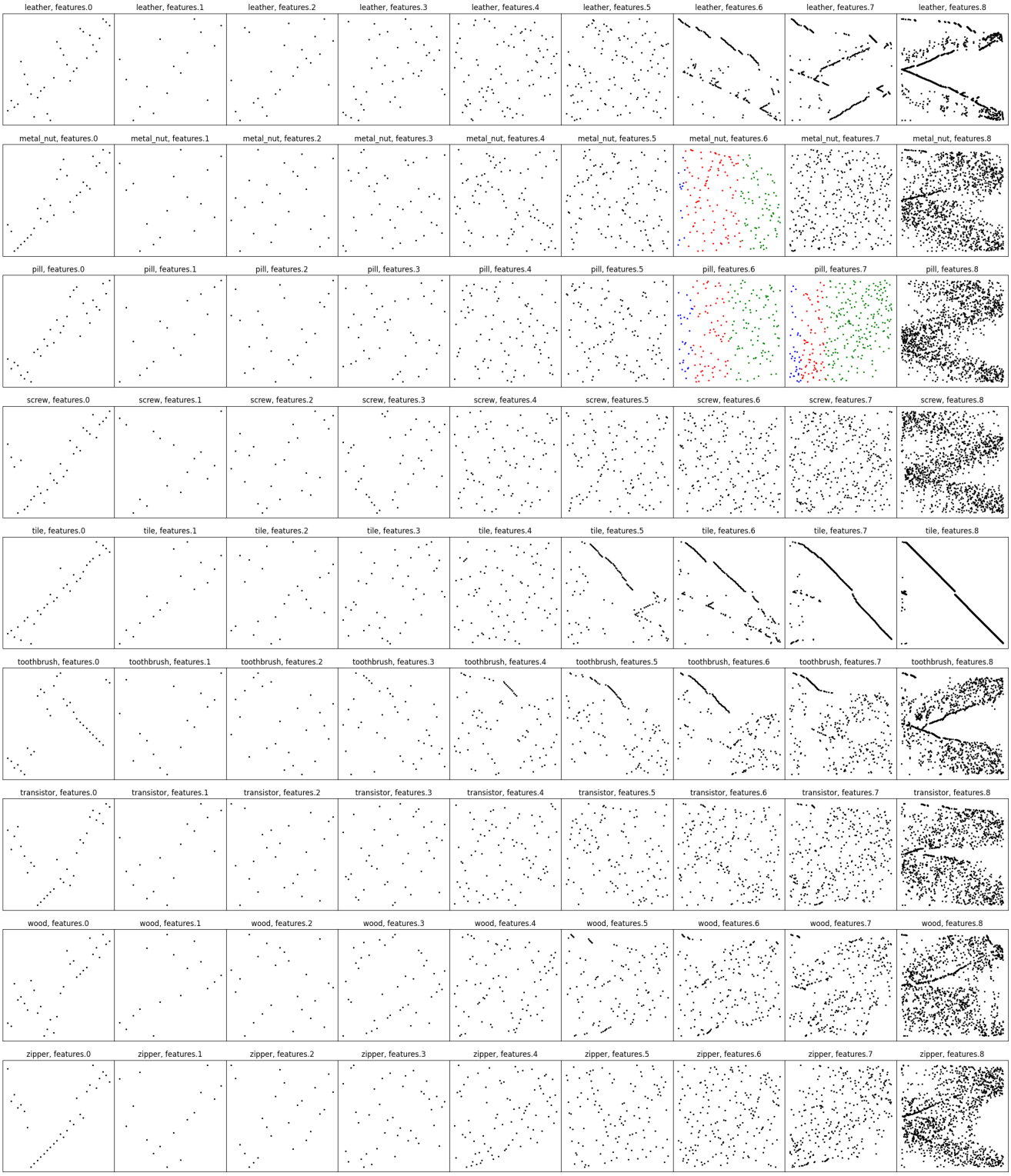


Figure 13: Step index vs. component index.

I. Are the eigencomponents from the plateau and drop regions redundant? Or noisy?

Can one say that the components selected after the "rise" regime (cf. definition of regimes in Section F) are contain redundant, noisy, or even spurious information? Figures 15 and 14 present the outcomes of two simulations aimed at addressing these questions.

The d eigencomponents of a Gaussian model are sorted as they were yielded by the results of the Bottom-Up greedy search using the full test set (i.e. overfit), so a dimension reduction with k components corresponds to the k-th step (or depth) of the search tree traversal. This order of eigencomponents is interpreted as "from the most useful to the least useful (or most harmful)" relative to the AD performance.

Starting at a position $k' \in \{1, \ldots, d\}$ of this list, the eigencomponents before it are retained, the rest is discarded, then synthetic axes are progressively added to replace the discarded ones. The respective performance of each simulated dimension reduction (original plus synthetic axes) is recorded and compared to the the original dimension reduction with the same size k. Example: for a feature vector with size $d = 10$, a starting position $k' = 4$, and simulated $k = 5$, the first 3 components are from the actual eigendecomposition (according to the greedy search order) and the two last components are synthetic.

Two starting positions k' are considered: the first and the last position of the "plateau" regime, which corresponds to retaining, respectively, the "rise" components and the "rise + plateau" components. Two types of synthetic signal are considered: noise (Figure 14) and redundant signals (Figure 15). The noise is drawn from a standard normal distribution (all axes are independent). The redundant signal is a Gaussian random projection[6] of the $k' - 1$ retained axes (original eigencomponents) – in other words, multiple random linear combinations of the existing signal. Four scenarios, considering the combinations of the aforementioned parameters, are considered tested with the feature vector size $k \in \{k', \ldots, d\}$.

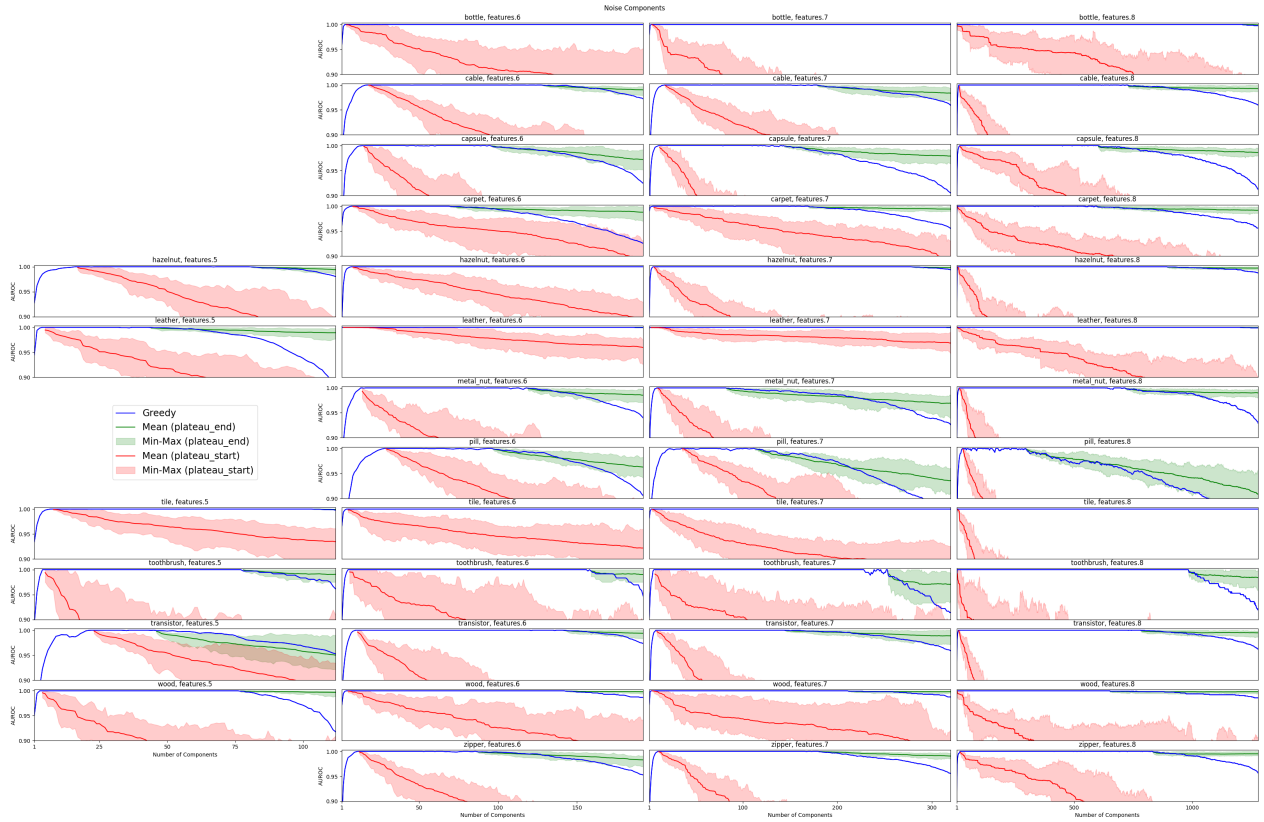


Figure 14: Simulated performance with **noise** signal. Red/green: replacement starts at the start/end of the "plateau" regime.

To make a fair comparison between the simulated performance and the original performance, the scale of a synthetic axis at position i is chosen such that its empirical standard deviation on the test set $\hat{\sigma}^{(i)}_{\text{test}}$ equals that of the original component at

[6] We use the implementation from scikit-learn (`sklearn.random_projection.GaussianRandomProjection`).

the position i in the sorted list. Each scenario is repeated with 30 different random seeds; Figure 14 and Figure 15 show the minimum, average, and maximum performance seen at each dimension reduction size k.

Most cases (category and node combination) show a consistent behavior: compared to the components in the "plateau" regime, noise deteriorates the performance, and faster (less synthetic axes) than the redundant signal, which often remains close to the plateau's performance.

These results suggest that the eigencomponents in the plateau regime behave like redundant synthetic data, actually with more stable behavior than the latter. The eigencomponents in the drop regime, however, have spurious features that do not discriminate the normal from the anomalous class – in fact provoking a faster performance drop than pure noise.

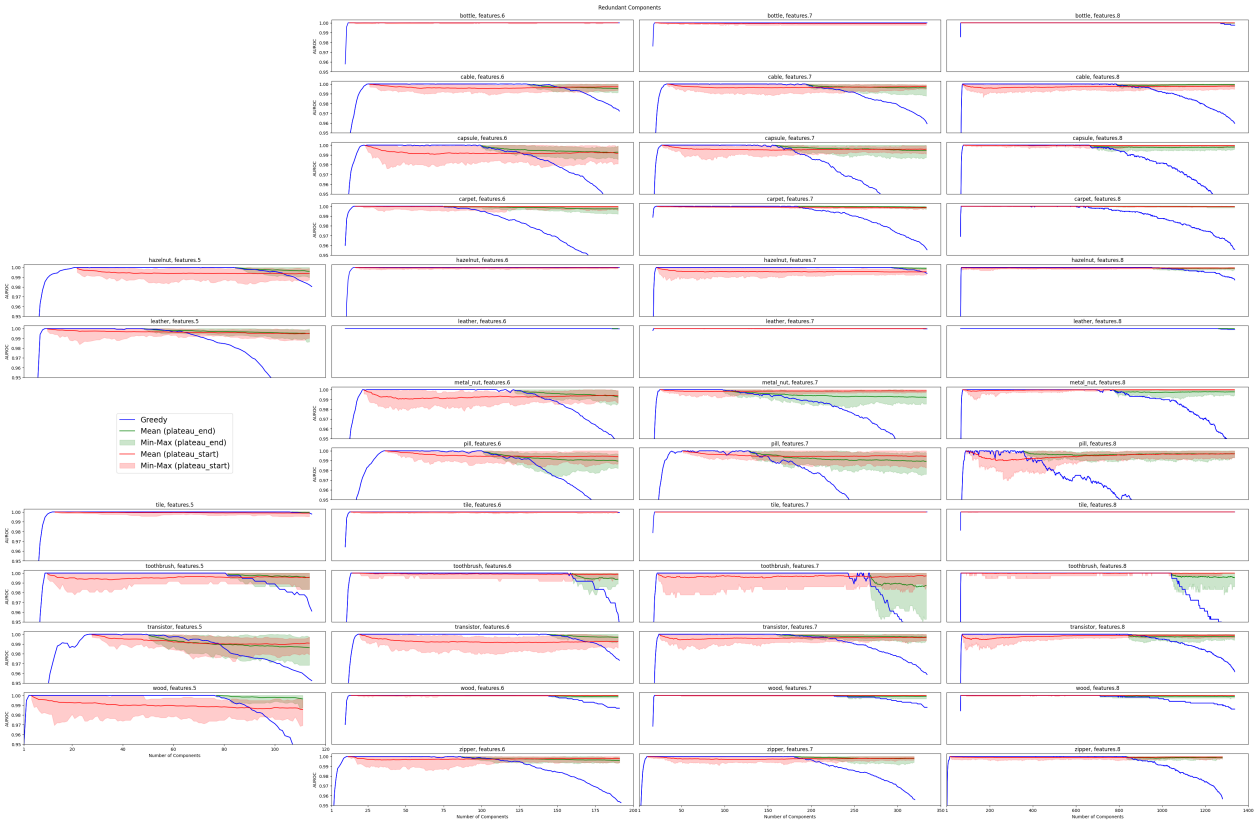


Figure 15: Simulated performance with **redundant** signal. Red/green: replacement starts at the start/end of the "plateau" regime.

J. MVTec-AD Dataset Overview

The table provides a comprehensive overview of the MVTec-AD dataset and the data split used in Experiment 3, described in Section 7. It displays the number of images for each anomaly type within every category. Additionally, it includes specific counts of anomalous images used for training purposes.

As in Experiment 3, we establish a minimum number of anomalous images 15 in $\mathbf{W}_{\text{greedy}}$ and randomly select images from all anomaly types, this table shows the number of images allocated for the greedy eigencomponent selection. The remaining anomalous images not included in the greedy set $\mathbf{W}_{\text{greedy}}$ constitute the evaluation set $\mathbf{W}_{\text{eval}}$, and all the normal images from MVTec-AD's test set are shared by both.

Category	Test set split	Images per anomaly type	Greedy search split	Evaluation split	Train split (only normal images)
bottle	broken small	22	5	17	
	contamination	21	5	16	
	broken large	20	5	15	
	Total	**63**	**15**	**48**	**209**
	good	20			
cable	missing_wire	10	2	8	
	cable_swap	12	2	10	
	bent_wire	13	2	11	
	cut_inner_insulation	14	2	12	
	poke_insulation	10	2	8	
	missing_cable	12	2	10	
	cut_outer_insulation	10	2	8	
	combined	11	2	9	
	Total	**92**	**16**	**76**	**224**
	good	58			
capsule	poke	21	3	18	
	faulty_imprint	22	3	19	
	squeeze	20	3	17	
	crack	23	3	20	
	scratch	23	3	20	
	Total	**109**	**15**	**94**	**219**
	good	23			
carpet	cut	17	3	14	
	thread	19	3	16	
	hole	17	3	14	
	metal_contamination	17	3	14	
	color	19	3	16	
	Total	**89**	**15**	**74**	**280**
	good	28			
grid	broken	12	3	9	
	thread	11	3	8	
	bent	12	3	9	
	glue	11	3	8	
	metal_contamination	11	3	8	
	Total	**57**	**15**	**42**	**264**
	good	21			
hazelnut	print	17	4	13	
	hole	18	4	14	
	cut	17	4	13	
	crack	18	4	14	
	Total	**70**	**16**	**54**	**391**
	good	40			
leather	glue	19	3	16	
	cut	19	3	16	
	fold	17	3	14	
	poke	18	3	15	
	color	19	3	16	
	Total	**92**	**15**	**77**	**245**
	good	32			
metal_nut	color	22	4	18	
	bent	25	4	21	
	scratch	23	4	19	
	flip	23	4	19	
	Total	**93**	**16**	**77**	**220**
	good	22			

Category	Test set split	Images per anomaly type	Greedy search split	Evaluation split	Train split (only normal images)
pill	color	25	3	22	
	scratch	24	3	21	
	contamination	21	3	18	
	combined	17	3	14	
	faulty_imprint	19	3	16	
	pill_type	9	3	6	
	crack	26	3	23	
	Total	**141**	**21**	**120**	**267**
	good	26			
screw	scratch_head	24	3	21	
	thread_top	23	3	20	
	scratch_neck	25	3	22	
	thread_side	23	3	20	
	manipulated_front	24	3	21	
	Total	**119**	**15**	**104**	**320**
	good	41			
tile	test,glue_strip	18	3	15	
	test,gray_stroke	16	3	13	
	test,oil	18	3	15	
	test,crack	17	3	14	
	test,rough	15	3	12	
	Total	**84**	**15**	**69**	**230**
	good	33			
toothbrush	defective	30	15	15	
	Total	**30**	**15**	**15**	**60**
	good	12			
transistor	cut_lead	10	4	6	
	misplaced	10	4	6	
	damaged_case	10	4	6	
	bent_lead	10	4	6	
	Total	**40**	**16**	**24**	**213**
	good	60			
wood	color	8	3	5	
	liquid	10	3	7	
	hole	10	3	7	
	combined	11	3	8	
	scratch	21	3	18	
	Total	**60**	**15**	**45**	**247**
	good	19			
zipper	combined	16	3	13	
	broken_teeth	19	3	16	
	split_teeth	18	3	15	
	squeezed_teeth	16	3	13	
	rough	17	3	14	
	fabric_interior	16	3	13	
	fabric_border	17	3	14	
	Total	**119**	**21**	**98**	**240**
	good	32			

Table 1: MVTec-AD Image Count Details (for Experiment 3)