Gaussian Image Anomaly Detection with Greedy Eigencomponent Selection

Tetiana Gula*
Mines Paris, PSL University
Centre for mathematical morphology
77300 Fontainebleau, France
tetiana.gula@etu.minesparis.psl.eu

João P. C. Bertoldo*
Mines Paris, PSL University
Centre for mathematical morphology
77300 Fontainebleau, France
jpcbertoldo@minesparis.psl.eu

Abstract

This paper addresses the challenge of Anomaly detection (AD) in images by proposing a novel dimensionality reduction technique using pre-trained convolutional neural network (CNN) with EfficientNet model. We introduce two tree search methods with a greedy strategy for improved eigencomponent selection. We conducted three experiments to evaluate our approach: examining components choice on test set performance when intentionally overfitting, training on one anomaly type and testing on others, and examining training with a minimal image set based on anomaly types. Unlike traditional methods that emphasize variance, our focus is on maximizing performance and understanding component behavior in diverse settings. Results show our technique outperforms both Principal Component Analysis (PCA) and Negated PCA (NPCA), suggesting a promising advancement in AD efficiency and effectiveness.

1. Introduction

Anomaly detection (AD) is a challenging task in machine learning with a wide range of applications, from fraud detection in financial transactions to fault diagnosis in industrial systems. In recent years, deep learning approaches have shown promising results in detecting anomalies from images, particularly using pre-trained convolutional neural networks (CNNs). However, one of the key challenges is that CNNs can produce a large number of features, which can lead to computational challenges, and the presence of redundant information may not contribute the detection task.

This work focuses on dimensionality reduction using a multivariate Gaussian (MVG) model trained on image features extracted from a pre-trained CNN, as proposed in [12], on the well-established MVTec Anomaly Detection (MVTec-AD) dataset [2, 1]. Specifically, our investigation focuses on the potential of using eigendecomposition combined with Sequential-Feature-Selection (SFS) [6], employing a greedy tree search approach to choose eigencomponents from the covariance matrix of a Gaussian model. We introduce two types of eigencomponent selection, namely Bottom-Up and Top-Down. The Bottom-Up approach gradually adds eigencomponents that yield the best performance, while the Top-Down approach gradually removes eigencomponents that impact it negatively.

We test our algorithm on a hypothetical setting where the eigencomponent selection has access to the test set of anomalies, revealing that it is *possible* to significantly boost a model's performance with an embedding space surprisingly small, thus unveiling substantial redundancy in the pre-trained feature spaces. In contrast, generalization is not easily achievable even in scenarios where the performance improvement is seemingly easy.

Our contribution highlights the importance of dimensionality reduction, and our proposed approach shows a promising alternative to traditional dimensionality reduction techniques in the field of AD.

2. Methods

We build on top of Gaussian-AD [13]: a multivariate Gaussian (MVG)-based model is built and a Greedy Eigencomponent Selection (GreedyES) is applied to directly optimize the AD performance. We show that this problem is equivalent to applying Sequential-Feature-Selection (SFS) [6] on the space of feature vectors transformed by a whitening operation, which we introduce as an intermediate step in [13] to simplify the eigencomponent selection.

2.1. Multivariate Gaussian (MVG)

A data point $I \in \mathbb{R}^n$ is passed as an input to a backbone neural network f; then, global average pooling is applied to the activations from its node N – the term "node", instead of "layer", is used to match the concept of node as "a block

* Equal Contribution.

of layers" as in `pytorch`. This referred to as "feature extractor" $f_N : \mathbb{R}^n \to \mathbb{R}^d$, where d is the number of channels outputted from N and $x = f_N(I)$ is a feature vector.

Assuming that the feature vectors extracted from a set of *normal* data points $\mathbf{I}_{\text{train}}$ follow an MVG distribution $\mathcal{N}(\mu, \Sigma)$, with mean μ and covariance matrix Σ, anomalous data points are likely to lie far away from the mean of this distribution, where the notion of distance is measured with the Mahalanobis distance:

$$D_M(x) = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \tag{1}$$

Using the feature vectors $\mathbf{X}_{\text{train}} = \{ f_N(I) \,|\, I \in \mathbf{I}_{\text{train}} \}$, the empirical mean vector $\hat{\mu} \in \mathbb{R}^d$ is fitted with the Maximum Likelihood Estimator (MLE) $\hat{\mu} = \frac{1}{|\mathbf{X}_{\text{train}}|} \sum_{x \in \mathbf{X}_{\text{train}}} x$, and the empirical covariance matrix $\hat{\Sigma} \in \mathbb{R}^{d \times d}$ is fitted using LeDoit-Wolf's method [9]. This estimator ensures a positive definite inverse covariance matrix $\hat{\Sigma}^{-1}$ by adding a regularization term to the MLE while automatically selecting the optimal regularization parameter based on the number of observations and features in the dataset, achieving a balance between bias and variance.

At inference time, the parameters $\hat{\mu}$ and $\hat{\Sigma}$ are plugged into Equation 1, the Mahalanobis distance $D_M(\cdot)$ is used as anomaly score (higher means "more anomalous"), and a binary decision ("is the image normal or anomalous?") is made based on a threshold. As further explained in Sec. 4.3, a threshold selection-free performance metric is used in this work, therefore we do not focus on the threshold-setting aspect.

2.2. Whitening

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 $\Lambda^{-\frac{1}{2}} Q^T$ – inverse square root taken elementwise since Λ is diagonal – can be used to build white feature vectors

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

and Equation 1 can be computed as its Euclidean norm

$$D_M(x) = \| \, w(f_N(I)) \, \|_2 \tag{3}$$

(details in Appendix A).

2.3. Greedy Eigencomponent Selection (GreedyES)

Notice that the entry w_i (the i-th entry of the white vector w) corresponds to the projection of the centered feature vector $(x - \hat{\mu})$ onto the eigenvector q_i (scaled by $\lambda_i^{-\frac{1}{2}}$), so applying feature selection on w is equivalent to choosing

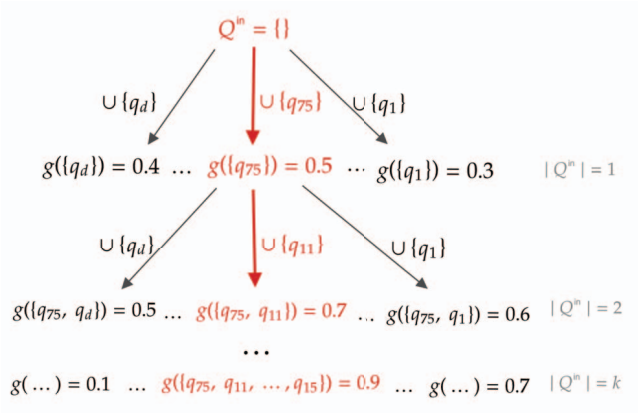


Figure 1: Greedy Bottom Up algorithm as tree search. Starting with an empty set, the algorithm iteratively adds the component with the best performance (function g) when combined withe the current selection until it reaches the size k. The red path represents the chosen path at each step.

eigencomponents from the set $\mathbf{Q} = \{q_1, \dots, q_d\}$ (eigenvalues omitted for simplicity). Consider $\mathbf{Q}_k \subseteq \mathbf{Q}$, such that $|\mathbf{Q}_k| = k$, and g, a performance metric (higher is better).

Our framework consists of finding the optimal subset

$$\mathbf{Q}_k^* = \underset{\mathbf{Q}_k \subseteq \mathbf{Q}}{\operatorname{argmax}} \; g(\mathbf{Q}_k) \tag{4}$$

that maximizes the performance metric g – which is chosen to be the Area Under the Receiver Operating Characteristic curve (AUROC) (details in Sec. 4.3).

Equation 4 is a combinatorial problem with a search space of size d-choose-k. To make this problem amenable, we propose to approximate it with a greedy algorithm analogous to Sequential-Feature-Selection (SFS) [6] iteratively building $\mathbf{Q}_k^*$ one eigencomponent at a time while locally optimizing g. In the space of white vectors w, our approach is indeed an SFS, but we coin it "Greedy Eigencomponent Selection (GreedyES)" to remind it is equivalent to truncating the eigendecomposition of $\hat{\Sigma}^{-1}$ to k components.

GreedyES can be carried out in two ways: starting with an empty set then adding them one by one, which we call the "Bottom-Up" variant (*i.e.* Forward-SFS, Algorithm 1, illustrated in Fig. 1), or starting with the set of all eigencomponents then removing them one by one, which we call the "Top-Down" variant (*i.e.* Backward-SFS, Algorithm 2). Note that Equation 2 makes it simple to simulate $\mathbf{Q}' \subseteq \mathbf{Q}$ by choosing the entries from w (details in Appendix A).

3. Related Work

Gaussian-AD [13] originally used Principal Component Analysis (PCA) to truncate $\hat{\Sigma}$ (*i.e.* dimensionality reduction in the feature space of w) and introduced *Negated* PCA

Algorithm 1 Greedy Bottom Up

Require: $d = |\mathbf{Q}| > 0$ and $1 \leq k \leq d$

procedure GREEDYBOTTOMUP($\mathbf{Q}$, k, g)
  $\mathbf{Q}^{\text{in}} \leftarrow \emptyset$ # set of eigenvectors *IN* the model
  $\mathbf{Q}^{\text{out}} \leftarrow \mathbf{Q}$ # set of eigenvectors *OUT* of the model
  while $|\mathbf{Q}^{\text{in}}| < k$ **do**
    $q^* \leftarrow \text{argmax}_{q \in \mathbf{Q}^{\text{out}}} \; g\left(\mathbf{Q}^{\text{in}} \cup \{q\}\right)$
    $\mathbf{Q}^{\text{in}} \leftarrow \mathbf{Q}^{\text{in}} \cup \{q^*\}$
    $\mathbf{Q}^{\text{out}} \leftarrow \mathbf{Q}^{\text{out}} \setminus \{q^*\}$

Algorithm 2 Greedy Top Down

Require: $d = |\mathbf{Q}| > 0$ and $1 \leq k \leq d$

procedure GREEDYTOPDOWN($\mathbf{Q}$, k, g)
  $\mathbf{Q}^{\text{in}} \leftarrow \mathbf{Q}$ # set of eigenvectors *IN* the model
  $\mathbf{Q}^{\text{out}} \leftarrow \emptyset$ # set of eigenvectors *OUT* of the model
  while $|\mathbf{Q}^{\text{in}}| > k$ **do**
    $q^* \leftarrow \text{argmax}_{q \in \mathbf{Q}^{\text{in}}} \; g\left(\mathbf{Q}^{\text{in}} \setminus \{q\}\right)$
    $\mathbf{Q}^{\text{in}} \leftarrow \mathbf{Q}^{\text{in}} \setminus \{q^*\}$
    $\mathbf{Q}^{\text{out}} \leftarrow \mathbf{Q}^{\text{out}} \cup \{q^*\}$

(NPCA). While the former retains the k eigencomponents q_i from $\hat{\Sigma}$ with the *largest* variance λ_i^2 (a.k.a. principal components), the latter retains those with the *smallest* λ_i^2. Using our formulation, NPCA with k dimensions corresponds to truncating w to $w_{1:k} = (w_1, \ldots, w_k)^T$, and PCA corresponds to $w_{d-k+1:d}$. Deep Feature Selection [10] generalized the notion of decomposing w based on λ_i^2 by defining the subvectors[1] $w_{1:m_2}$, $w_{m_2:m_1}$, and $w_{m_1:d}$, where m_1 and m_2 are breakpoints heuristically computed based on λ_i^2.

In the Gaussian-AD framework, authors of [12, 13] employed various EfficientNet models, whereas in our work we used EfficientNetB0. For comparative purposes, we illustrate the results of a different backbone EfficientNetB4 (see Sec. C) presented in the mentioned papers and refer to the framework as Gaussian AD* to mark the use of a different backbone.

Other pixel-wise MVG-based works have used other dimension reductions. Patch Distribution Modeling Framework (PaDiM) [4] uses a random feature selection directly on the feature space of $x = f(I)$ (notation is imprecise but the models are, indeed, conceptually analogous). Semi-orthogonal [8] generalized this idea using random orthogonal projection matrices – the former can be seen as a special case of the latter by using only binary entries. These two strategies provide faster training because they do not require decomposing $\hat{\Sigma}$, and they correspond, in fact, to using a random matrix as the left operand in Equation 2 – and so can (N)PCA, by zeroing the first or last rows of that matrix.

We propose a more general framework where the choice of w_i is unrestricted and, therefore, untangled from the notion of variance (λ_i^2) inherited from PCA. Besides, the hyperparameter k, commonly expressed by the ratio of retained variance $\left(\sum_{\lambda \in \mathbf{\Lambda}'} \lambda^2\right) / \left(\sum_{\lambda \in \mathbf{\Lambda}} \lambda^2\right)$, where $\mathbf{\Lambda}' \subseteq \mathbf{\Lambda} = \{\lambda_1, \ldots, \lambda_d\}$, has been given little attention to in these previous works – [7] proposed alternatives to select k with un/semi-supervised heuristics. In contrast, our experiments thoroughly consider all the possible values of $k \in \{1, \ldots, d\}$ for (N)PCA and GreedyES.

Our results show that the constraints imposed by (N)PCA (first or last eigencomponents) largely inhibit the full potential of dimensionality reduction; while Semi-orthogonal [8] theorizes that the smallest eigenvalues yield optimal AD performance, our results contradict this assumption, suggesting the variance-performance entanglement is spurious – thus we recommend that dimensionality reduction should *not* be indexed by the retained variance.

Finally, it is worth noting that PatchCore [14] – formerly State of the art (SOTA) on MVTec-AD (see Sec. 4.1 and Sec. C) – also uses random linear projections as an intermediate step to reduce the execution time of the coreset algorithm. However, as the reduced memory bank comes out of it, the dimension reduction is not further used, which could be a research direction to be explored in future work.

GreedyES should *not* be confused with Incremental PCA (IPCA). While IPCA is useful for reducing computational complexity in processing large datasets, our approach is designed to excel at highlighting anomalous images by selecting eigencomponents of $\hat{\Sigma}$ (all of them are computed).

4. Experimental setup

We establish a general setup to study GreedyES then use three different data splits (Sec. 4.4) for the execution and evaluation, which change the meaning of the results, so discussions are presented separately.

GreedyES is analyzed across the major nodes of an EfficientNet backbone and across all categories in MVTec-AD. In a given scenario (fixed category and node), all the possible values of $k \in \{1, \ldots, d\}$ are evaluated using both greedy modes (Bottom-Up and Top-Down), and results are plotted as k-vs-AUROC curves, with the reduced number of dimensions k on the X-axis and its respective AUROC on the evaluation data set on the Y-axis. Note that the baseline (no dimensionality reduction) corresponds to the rightmost point on these curves ($k = d$). The graphs from all scenarios are documented in the Appendix, while we show representative cases in the main text.

Our setup does *not* focus on model selection, but on observing the behavior of GreedyES *across scenarios*, so the reader is encouraged to see the Appendices. For instance, we do not seek to answer "Which is the best node?" – it has been observed [13, 10, 4, 14] that mid-depth (nor shallow-

[1] [10] terms "sub*spaces*", but we extrapolate the idea with our notation.

est nor deepest) nodes tend to yield best performance – our results, however, contradict this observation. Considering multiple backbone nodes, which are independent Gaussian-AD models, provides insights about the backbone itself. Similarly, we do *not* show cross-category average performance, but rather *per*-category performances because they are all independent AD tasks. While the former provides a literature-comparable summary metric, our focus is to identify phenomena that generalize across different data.

4.1. Dataset

We use the MVTec Anomaly Detection (MVTec-AD) dataset [2, 1]. It comprises 15 categories with 3629 normal images for training and 1725 images for testing. Each category is used independently as *15 independent datasets*, with its training set containing only defectless (normal) images , while its test set contains both normal and anomalous images, which are from a variety of defects, such as surface scratches, dents, distorted or missing object parts, etc (details in Table 1 in the Appendix J). The defects were manually generated to produce realistic anomalies as they would occur in real-world industrial inspection scenarios. Image samples are deliberately not shown for the sake of space[2].

4.2. Feature extractor

Like DFS [10], EfficientNet [15] is used as backbone for the feature extraction. Specifically, we use EfficientNetB0 pre-trained on classification on ImageNet[3]. We analyze the nine main nodes from EfficientNetB0, which are sequentially named from "`f0`" to "`f8`"[4](short for "`features.N`", which is torchvision's notation for EfficientNet's major nodes). The vectors' size d usually ranges from 10s to 100s, reaching up to $\sim$1000 in the deepest node (`f8`). Our backbone choice is mostly based on previous works (for comparability) and due to EfficientNets having many (nine) major nodes, so the transition from shallow to deep nodes is smoother than, for instance, ResNet [11].

4.3. Performance Metric

We use the Area Under the Receiver Operating Characteristic curve (AUROC) [5] both as the g function and as evaluation metric. The AUROC score is a widely used metric for AD and conveniently threshold selection-free, unlike binary classification metrics like the accuracy. It measures the probability that the anomaly score assigned to an anomalous instance is higher than that assigned to a normal instance. An AUROC of 0.5 indicates random guessing, and an AUROC of 1 indicates perfect discrimination.

In practice, using the True Positive Rate (TPR) and the False Positive Rate (FPR) as functions of the threshold T,

$$\text{AUROC} = \int_0^1 tpr(fpr^{-1}(x))\,dx \quad , \tag{5}$$

where $fpr : T \mapsto \text{FPR}$ and $tpr : T \mapsto \text{TPR}$. Other performance metrics have been omitted for the sake of clarity since, as shown in the results, models with 100% AUROC score (*i.e.* perfect discrimination) can be achieved.

4.4. Data split

Let $\mathbf{W}_{\text{test}} = \{\, w(f_N(I)) \,|\, I \in \mathbf{I}_{\text{test}} \,\}$ be the set of white vectors (Equation 2) from the test set $\mathbf{I}_{\text{test}}$ from MVTec-AD – *not* the set used for evaluation! While the train set is fully (and only) used to compute $\hat{\mu}$ and $\hat{\Sigma}$, the set $\mathbf{I}_{\text{test}}$ is further split in two: $\mathbf{W}_{\text{greedy}}$ and $\mathbf{W}_{\text{eval}}$. The set $\mathbf{W}_{\text{greedy}}$ is used to execute GreedyES, in the function g (Algorithms 1 and 2), and the set $\mathbf{W}_{\text{eval}}$ is used to evaluate the models (reported values in the results). Note that AUROC requires $\mathbf{W}_{\text{greedy}}$ and $\mathbf{W}_{\text{eval}}$ to contain both normal *and* anomalous instances. A fully unsupervised scenario assumes no access to anomalous samples, but our focus is to gain insights into the MVG model.

5. Experiment 1: test set overfit

5.1. Experiment 1 Setup

We set $\mathbf{W}_{\text{greedy}} = \mathbf{W}_{\text{eval}} = \mathbf{W}_{\text{test}}$, which is an intentional overfit of the evaluation set. The goal here is *not* to learn, but rather analyze the potential missed by (N)PCA due to its constraints related to the eigencomponents' variances λ_i^2. Despite this setup being unrealistic, it is useful for diagnostic purpose because one can measure GreedyES's *potential* compared to PCA, NPCA [12], and DFS [10]. Furthermore, additional analyses also reveal interesting insights into the feature extractor itself.

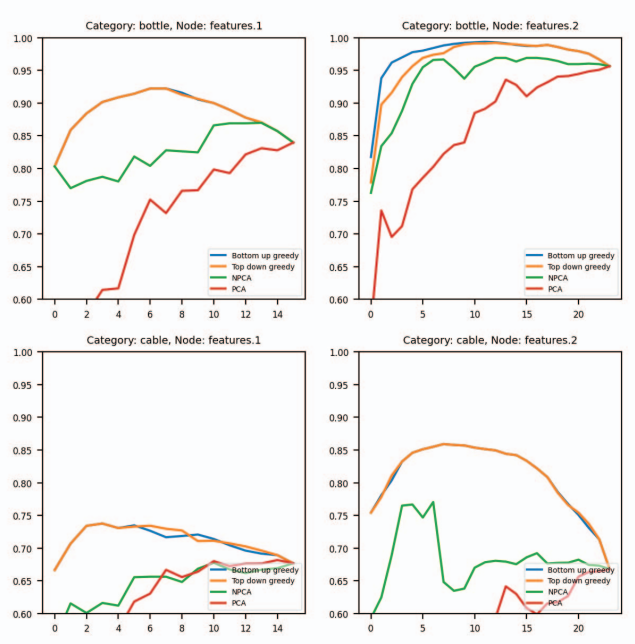
In Experiment 2 (Sec. 6) and Experiment 3 (Sec. 7), $\mathbf{W}_{\text{greedy}}$ and $\mathbf{W}_{\text{eval}}$ do *not* have anomalous images in common so the generalization power of our proposed framework can be compared to the *achievable* performances revealed by Experiment 1. In particular, we focus Experiment 2 and Experiment 3 on the nodes from `f5` up to `f8` because they can reach 100% AUROC and show more stable behavior.

5.2. Experiment 1 Results

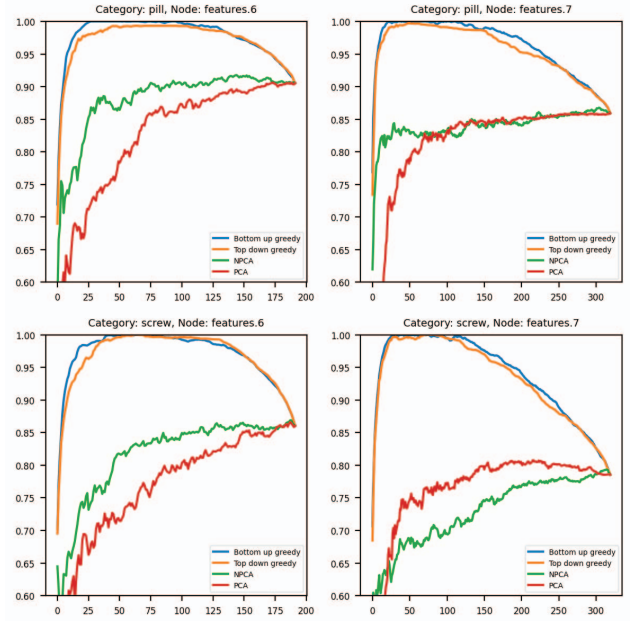
Fig. 2 shows a selection of k-vs-AUROC curves from representative cases in Experiment 1 – all scenarios are documented in Fig. 6 (Appendix B). GreedyES (Bottom-Up and Top-Down) is compared with PCA and NPCA. Notice that at the point where $k = d$ (*i.e.* no dimension reduction) all the lines merge because they simplify to the same model.

Fig. 3 shows a summary plot with the node index on the X-axis and the best AUROC achieved by that node on the Y-axis – each graph corresponds to picking a row (category)

[2]Examples can be found at www.mvtec.com.
[3]EfficientNet_B0_Weights.IMAGENET1K_V1 from torchvision.
[4]Other authors have named them from 1 to 9.



(a) { bottle, cable } × { f1, f2 }



(b) { pill, screw } × { f6, f7 }

Figure 2: Experiment 1: selection of representative k-vs-AUROC curves. All scenarios in Fig. 6 (Appendix B).

from Fig. 6 (Appendix B) and extracting the maximum AUROC for each curve. Only Bottom-Up is shown because it shares similar results with the Top-Down. Additionally, we compare it with the results achieved in [10][5]. Since [10] proposes several subspace decompositions (*i.e.* several dimension reductions), we selected the alternative "$[\Phi_2, \Phi_3]$" – which is equivalent to NPCA with a heuristic choice of k – because it achieves the best results in most scenarios. We also show the results when no dimension reduction is applied. This helps us grasp how this "ideal" GreedyES compares to others.

[5]We thank the authors for having shared their results data with us.

5.3. Experiment 1 Discussion

Our analysis suggests that it is *possible* to (sometimes greatly) enhance the performance of the MVG model by cherry-picking eigencomponents from $\hat{\Sigma}$ – it is Furthermore, we find that deeper layers require fewer components to achieve high-performance (details in Appendix G).

Most scenarios using Bottom-Up exhibit a monotonic increase in performance until reaching a saturation level, after which they exhibit diminishing performance. This behavior is further analyzed in Appendix F, where we split the k-vs-AUROC curves in three regimes (*i.e.* "phases"): rise, plateau, and drop. Some scenarios, however, show an edge case behavior with a rise followed directly by a drop of performance (*e.g.* categories "metal nut", "pill", and "screw" with f8, and category grid with the last four nodes).

It's clear that (N)PCA – even at optimal k – are generally much bellow the achievable performance demonstrated by GreedyES (see Fig. 2). High performance can be achieved using only 30-40 components, and even nearly-perfect class discrimination (100% AUROC) with less than ten components (see categories "bottle", "carpet", "hazelnut", "leather", "toothbrush", "tile", "wood" in the Appendix F). In other words, it is possible to encode the normality of a semantic class in very small embeddings (compared to d), underscoring the importance of dimension reduction for deep layers of the network, where the number of dimensions can be substantial (100s or even 1000s).

Fig. 3 shows that – except for category "grid" – deeper layers tend to be more informative for AD than shallower ones, being capable of achieving perfect score (or very close to it). This finding contradicts previous conclusions because, when using (N)PCA or no dimension reduction, the deepest layers tend to show a drop in performance (*e.g.* categories "capsule", "pill", "screw", "toothbrush"), which was believed to be due to a bias towards the pre-training task.

Bottom-Up vs. Top-Down Bottom-Up and Top-Down, behave similarly in most scenarios. However, Bottom-Up tends to achieve better results with longer plateaus, therefore successfully avoiding spurious eigencomponents. Finally, some exceptional cases were observed and pointed out in Appendix B.

5.4. Additional analyses

Experiment 1 shows that the features learned by the CNN f can be used to build surprisingly small (yet effec-

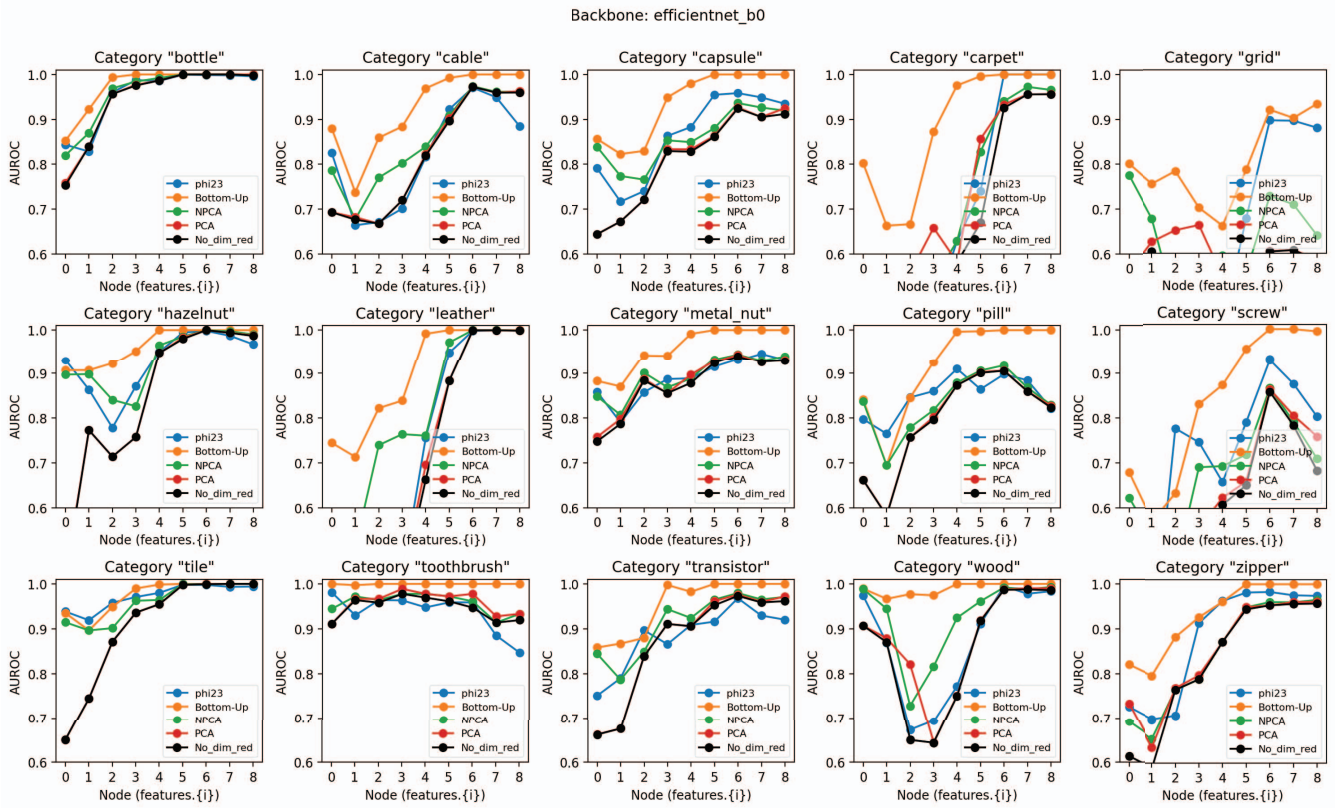


Figure 3: Experiment 1: best AUROC out of all values of k per node. The curve "phi23" refers to the results from [10] with the alternative "$[\Phi_2, \Phi_3]$".

tive) embeddings to detect anomalies in images. In Fig. 2b, for example, GreedyES reaches 100% AUROC with < 30 eigencomponents out of $\sim$ 300 while the best (N)PCA reaches $< 85\%$ AUROC. Therefore, more detailed analyses of this experiment are presented in Appendix H and in Appendix I.

The ratio of retained variance $\left(\sum_{\lambda \in \mathbf{\Lambda}'} \lambda^2\right) / \left(\sum_{\lambda \in \mathbf{\Lambda}} \lambda^2\right)$, where $\mathbf{\Lambda}' \subseteq \mathbf{\Lambda} = \{\lambda_1, \ldots, \lambda_d\}$, in an (N)PCA-decomposed subspace is commonly used as a reference signal for the dimension reduction parametrization, but Appendix H reveals that there is no connection between eigencomponent-wise variance λ_i^2 and its AD suitability. Otherwise said, the eigencomponents with the largest or smallest variance do not discriminate normal from anomalous data better than one another, contradicting the core premise of (N)PCA.

Appendix I shows two simulations investigating the nature of the three regimes observed with the Bottom-Up algorithm: rise, plateau, drop (details in Appendix F). Results suggest that the eigencomponents in the plateau are redundant with the most useful ones, and eigencomponents in the drop regime have spurious features provoking a performance to drop faster than noise, so they likely assign higher activations to normal images (the opposite of what is expected), which has also been observed in [3].

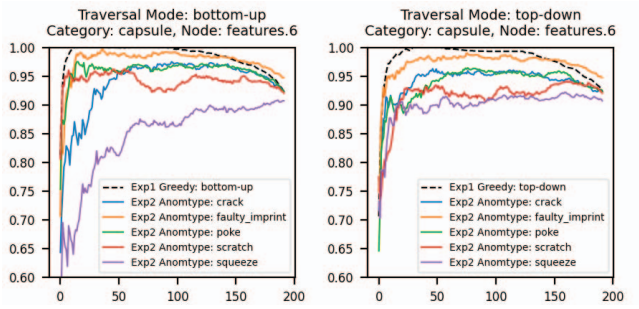
These analyses explain why (N)PCA require a larger k to achieve their best score and it is yet systematically lower than GreedyES's: the constraint imposed to select eigencomponents based on their variance results in selecting spurious ones from the drop regime (Appendix F). On the other hand, NPCA consistently outperforms PCA, suggesting indeed some – as we show, not-enough – correlation between eigencomponent variance and AD performance.

6. Experiment 2: per anomaly type

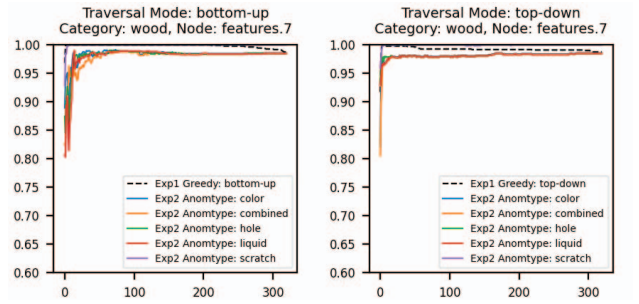
6.1. Experiment 2 Setup

Experiment 2 consists of segregating the anomaly types in $\mathbf{W}_{\text{greedy}}$ and $\mathbf{W}_{\text{eval}}$. Unlike the Experiments 1 and 3, here the candidates are ranked (function g) based on a single anomaly type. The evaluation, however, encompasses all the other anomaly types, while both $\mathbf{W}_{\text{greedy}}$ and $\mathbf{W}_{\text{eval}}$ use all the normal images from the test set. This experiment enables us to assess how well GreedyES generalizes from a single anomaly type, providing insights into its ability to extrapolate the learned patterns.

6.2. Experiment 2 Results



(a) capsule, `f6`



(b) wood, `f7`

Figure 4: Experiment 2: selection of representative k-vs-AUROC curves. All scenarios in Fig. 9 (Appendix D).

Fig. 4 shows a selection of k-vs-AUROC curves from representative cases in Experiment 2 – all scenarios are documented in Fig. 9 (Appendix D). All the data splits for a given category (named after the anomaly type used in $\mathbf{W}_{\text{greedy}}$) are compared with the respective curve from Experiment 1 – note that each curve has a different $\mathbf{W}_{\text{eval}}$. The results from nodes from `f0` up to `f4` have been omitted because their *achievable* performances (see Experiment 1 in Sec. 5) are generally worse.

6.3. Experiment 2 Discussion

In Fig. 4a, we observe that Experiment 1 outperforms all the curves based on a single anomaly type in both modes (Bottom-Up and Top-Down). In other words, the GreedyES struggles to generalize well to unseen anomalies. Besides, its adaptability across diverse categories lacks consistency.

While Experiment 1 shows it is possible to obtain a perfect classifier with less than 30 eigencomponents – and adding up to other 70 eigencomponents does not hurt the performance – none of the single-anomaly-type runs were capable of ever reaching such performance. As shown in Appendix D, most scenarios have this behavior with more or less cross-anomaly type variability (*e.g.* category "capsule" has a stronger dependency on the anomaly type than the category "cable").

Fig. 4b shows a more stable behavior in the sense that all the anomaly types have nearly the same curve. Still, GreedyES runs in Experiment 2 are comparable and more often better than (N)PCA, which comes without surprise due to the supervision used in the former. However, even in such cases, it generally fails to achieve the same level of performance seen in Experiment 1.

Bottom-Up vs. Top-Down Bottom-Up often reaches better maximum performance with lower k, while Top-Down is more stable at keeping the baseline performance (no dimension reduction) and shows a less variable behavior. Categories "carpet" and "zipper" (node `f7` in particular) are good examples of such contrast. Other examples include categories "hazelnut", "leather", "transistor", and "wood".

7. Experiment 3: fixed number of images

7.1. Experiment 3 Setup

Instead of basing the choice of components on the whole test set (Experiment 1) or on a single anomaly type (Experiment 2), we now randomly select images from all anomaly types given a fixed minimum number of anomalous images in $\mathbf{W}_{\text{greedy}}$. The data split is repeated with 5 seeds such that $\mathbf{W}_{\text{greedy}}$ has at least 15 anomalous images (equal number of images per anomaly type) and the remaining anomalous images go to $\mathbf{W}_{\text{eval}}$; both $\mathbf{W}_{\text{greedy}}$ and $\mathbf{W}_{\text{eval}}$ use all the normal images from $\mathbf{W}_{\text{test}}$. More details in Table 1 in Appendix J.

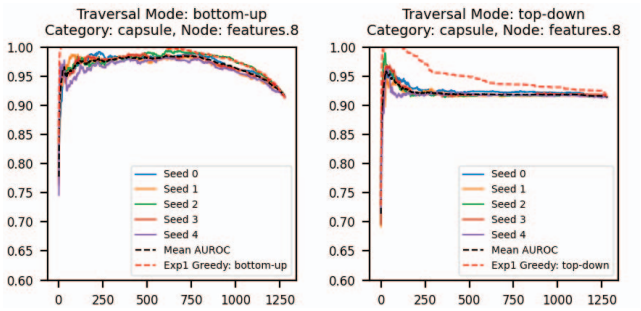
7.2. Experiment 3 Results

Fig. 5 shows a selection of k-vs-AUROC curves from representative cases in Experiment 3 – all scenarios are documented in Fig. 10 (Appendix E). Each curve corresponds to a seed, and their cross-seed mean curve and the curve from Experiment 1 are plotted together for reference. The results from nodes from `f0` up to `f4` have been omitted because their *achievable* performances (see Experiment 1 in Sec. 5) are generally worse.

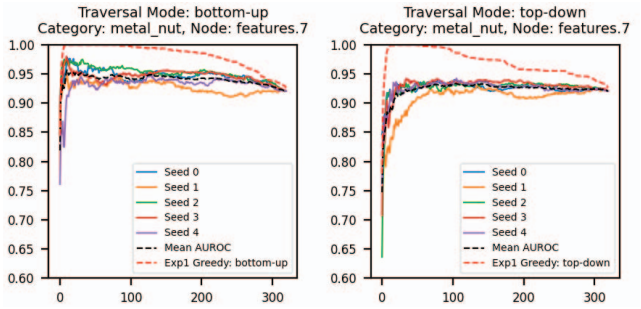
7.3. Experiment 3 Discussion

Compared to Experiment 2, Experiment 3 shows slightly better results with less variance across runs of a same scenario, which is expected because $\mathbf{W}_{\text{greedy}}$ is not biased towards a single anomaly type – two counter examples are worth noting: categories "pill" and "transistor". Still, a similar pattern often arises: while the curve from Experiment 1 reaches 100% AUROC, the others fail to avoid bad components.

Fig. 5b shows a noticeable pattern in Experiment 3. While Experiment 1 reveals a rather important margin for improvement (relative to the baseline without dimension reduction), the ability to generalize with reduced amount of data is very limited, and the discrepancy is usually bigger



(a) capsule, `f8`



(b) metal nut, `f7`

Figure 5: Experiment 3: selection of representative k-vs-AUROC curves. All scenarios in Fig. 10 (Appendix E).

for with the Bottom-Up mode. Again, the GreedyES fails to avoid bad components, although performances are generally better than (N)PCA, which (again) is not surprising because the latter have no supervision at all. Other noticeable examples include categories "capsule", "carpet", "metal nut", "pill", and "screw".

Fig. 5a shows an encouraging example where the Bottom-Up approach is more successful. Most runs achieved substantial performance improvement relative to the baseline (no dimension reduction) with low variability. Category "carpet" with node `f6` and category "leather" with node `f5` also represent well such behavior.

In cases where the baseline typically has more than 95%, this is often the case as well. Although the relative improvement is not as prominent (baseline is already high), the dimensionality reduction – at nearly-constant performance – is considerable, revealing that removing just the redundancy is easier. Some examples include categories "cable" and "zipper" with nodes from `f6` to `f8`, and categories "carpet" and "hazelnut" with node `f8`.

Bottom-Up vs. Top-Down The same pattern observed in Experiment 2 is seen here: Bottom-Up is more embedding-size-efficient, while Top-Down manages to only keep the baseline performance (no dimension reduction). In Experiment 3, however, the two modes do not differ as much, making the Top-Down a safer choice. Examples of this can be seen in categories "capsule" and "hazelnut" (nodes from `f6` to `f8`).

8. Conclusion

The paper presents three experiments evaluating a novel dimension reduction strategy for Anomaly detection (AD) combining multivariate Gaussian (MVG)-based models with eigendecomposition of the covariance matrix and a greedy tree search algorithm.

The first experiment intentionally overfits the test set, revealing it is *possible* to achieves higher performance with smaller embeddings than previous approaches like Principal Component Analysis (PCA) and *Negated* PCA (NPCA). However, the second and third experiments, exploring its generalization capacity, reveal that the algorithm struggles to generalize well, and identifying spurious discriminatory features remains challenging. We hypothesize that the struggles observed in Experiment 2 and Experiment 3 can be mitigated by using a different criteria to select eigencomponents (*i.e.* a different g in Sec. 2) as in [7].

Our analysis shows contradictions with previous theories; in particular, it was observed that the per-eigencomponent variance (λ_i^2) does not correlate with its suitability for anomaly detection. Finally, it must be noted that our conclusions are limited to EfficientNet – although experiments were carried over multiple datasets and nodes – and future work could explore more recent architectures like ResNext [16].

References

[1] Paul Bergmann, Kilian Batzner, Michael Fauser, David Sattlegger, and Carsten Steger. The MVTec Anomaly Detection Dataset: A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. *International Journal of Computer Vision*, 129(4):1038–1059, Apr. 2021. 1, 4

[2] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. MVTec AD – A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 4

[3] Joao P. C. Bertoldo and David Arrustico. Visualization for multivariate gaussian anomaly detection in images. Accepted to the 12th International Conference on Image Processing Theory, Tools and Applications (IPTA 2023). 6

[4] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. PaDiM: A Patch Distribution Modeling Framework for Anomaly Detection and Localization. In Alberto Del Bimbo, Rita Cucchiara, Stan Sclaroff, Giovanni Maria Farinella, Tao Mei, Marco Bertini, Hugo Jair Escalante, and Roberto Vezzani, editors, *Pattern Recognition. ICPR International Workshops and Challenges*, pages 475–489, Cham, 2021. Springer International Publishing. 3

[5] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. 4

[6] F. J. Ferri, P. Pudil, M. Hatef, and J. Kittler. Comparative study of techniques for large-scale feature selection. In Edzard S. GELSEMA and Laveen S. KANAL, editors, *Pattern Recognition in Practice IV*, volume 16 of *Machine Intelligence and Pattern Recognition*, pages 403–413. North-Holland. 1, 2

[7] Zeyu Jiang, João P. C. Bertoldo, and Etienne Decencière. Heuristic hyperparameter choice for image anomaly detection, 2023. Accepted to the 12th International Conference on Image Processing Theory, Tools and Applications (IPTA 2023). 3, 8

[8] Jin-Hwa Kim, Do-Hyeong Kim, Saehoon Yi, and Taehoon Lee. Semi-orthogonal Embedding for Efficient Unsupervised Anomaly Segmentation, 2021. 3

[9] Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. 88(2):365–411. 2

[10] Jie Lin, Song Chen, Enping Lin, and Yu Yang. Deep Feature Selection for Anomaly Detection Based on Pretrained Network and Gaussian Discriminative Analysis. *IEEE Open Journal of Instrumentation and Measurement*, 1:1–11, 2022. 3, 4, 5, 6, 11, 23

[11] Fenglin Liu, Xuancheng Ren, Zhiyuan Zhang, Xu Sun, and Yuexian Zou. Rethinking skip connection with layer normalization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3586–3598. International Committee on Computational Linguistics. 4

[12] Oliver Rippel, Patrick Mertens, Eike König, and Dorit Merhof. Gaussian Anomaly Detection by Modeling the Distribution of Normal Data in Pretrained Deep Features. *IEEE Transactions on Instrumentation and Measurement*, 70:1–13, 2021. 1, 3, 4, 16

[13] O. Rippel, P. Mertens, and D. Merhof. Modeling the Distribution of Normal Data in Pre-Trained Deep Features for Anomaly Detection. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6726–6733, Los Alamitos, CA, USA, Jan. 2021. IEEE Computer Society. 1, 2, 3, 16

[14] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14318–14328. 3, 16

[15] Mingxing Tan and Quoc Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 6105–6114. PMLR, May 2019. 4

[16] Ross Wightman, Hugo Touvron, and Herve Jegou. ResNet strikes back: An improved training procedure in timm. In *NeurIPS 2021 Workshop on ImageNet: Past, Present, and Future*. 8