WHICH WAY IS BROKEN? LANGUAGE-FREE ZERO-SHOT ANOMALY DETECTION WITH FROZEN VISION FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Zero-shot anomaly detection (ZSAD) aims to identify and localize unseen defects without access to reference target samples. Prior approaches rely heavily on Vision-Language Models (VLMs) by matching visual patch tokens with hand-crafted or learnable text prompts. However, this language-grounded setup creates a fundamental bottleneck: visual defects are open-ended and fine-grained, making them difficult to describe accurately in text. Recent language-free methods avoid text, but they change the vision model itself. We show that neither is needed. Frozen vision models naturally separate normal and defective patches along a direction that is shared across datasets. We learn a few such directions for normal and defective patches directly on the frozen features, and score each patch by how much it is closer to the corresponding set. During training, each direction is treated as a small spread of nearby directions rather than a single point. Since defects on unseen products point in slightly different directions, such a direction is more likely to transfer. The method uses no text, changes no model weights, and needs very little training data. Across nine benchmarks in strict zero-shot settings, it matches or exceeds language-based and vision-only methods on AUROC, and improves F1-max by 6.8 points at the pixel level and 1.6 points at the image level. It also performs well in the data-constrained regime, trained on only two images per class, it already matches or outperforms prior methods trained on the full data. It works with six different vision backbones and benefits directly from stronger ones. The source code will be publicly released.

1 INTRODUCTION

Unlike traditional anomaly detection (Defard et al., 2021; Roth et al., 2022), Zero-shot anomaly detection (ZSAD) detects and segments unseen defects on objects it has never seen, without access to reference images from target data during training and testing (Ma et al., 2025; Qu et al., 2025; Jeong et al., 2023). Most recent methods build on vision–language models (VLMs) such as CLIP (Radford et al., 2021b), they compare each image patch with text such as "a photo of a damaged object" (Jeong et al., 2023; Zhou et al., 2024). This design has two limitations. First, hand-crafted prompts ask the user to name the object or to guess the kinds of defects in advance, yet real defects are open-ended and hard to predict. Second, the text encoder ties these methods to VLM backbones. VLMs are trained to match whole images with captions, not to capture fine-grained surface detail, so they are weak at localizing small defects (Maninis et al., 2025). Stronger vision-only models, such as DINOv3 (Siméoni et al., 2025), cannot be used at all because they lack a text encoder.

Learnable prompts remove the need to write prompts by hand (Zhou et al., 2024; Cao et al., 2024). They still depend on text, and still frame each prompt around the words "normal" and "abnormal". Their learned prompt vectors also tend to overfit the auxiliary training data, accuracy on unseen datasets peaks early in training and then drops or fluctuates (Qu et al., 2025; Hou et al., 2026). Bayes-PFL (Qu et al., 2025) reduces this by learning a distribution over prompts instead of a single prompt, but it remains tied to CLIP and its text encoder.

A few recent methods drop the language module and work directly with vision foundation models (VFMs). They do so by changing the model. AnomalyVFM (Fučka et al., 2026) fine-tunes the backbone with low-rank adapters (LoRA) on a large set of synthetic defects, about five times more

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

training images than standard ZSAD methods use. VisualAD (Hou et al., 2026) inserts learnable tokens into the frozen backbone, where they mix with the image patches through self-attention. These tokens interfere with the already-pretrained patches via cross-attention, thereby changing the patch features themselves. Both approaches alter the representation that made the foundation model useful in the first place.

To solve these problems, we ask a simpler question: *can we detect and localize anomalies with a frozen VFM, without text and without changing any of its weights?* We find that we can. Frozen VFMs already separate normal and defective patches along a shared direction in their feature space, and this direction points the same way across objects, datasets, and all six VFMs we test (section 3, section B.2). Detecting anomalies, therefore, reduces to finding this direction. We learn a few reference directions for normal patches and a few for defective ones, directly on the frozen features, and score each patch by how much closer it is to the defect directions than to the normal ones. During training, each direction is found in the hyperspherical latent space of foundation models using the von Mises–Fisher sampler. The backbone's weights are never updated, and no token is added to its input.

Because the direction already exists in the model, finding it takes little data and few parameters. Our method trains only about 60k parameters and, when trained on just two images per class, already outperforms prior methods trained on the full auxiliary set. Because it needs no text, it works with all six VFMs we tried and benefits directly from stronger backbones.

In this work, our main contributions are:

- We show that frozen VFMs separate normal and defective patches along a shared direction that is consistent across objects, datasets, and backbones. This makes text prompts and text encoders unnecessary for ZSAD.
- We propose a simple method that learns this direction directly on frozen features. It uses no text, updates no backbone weights, and adds no tokens to the input.
- Across nine benchmarks, our method matches or exceeds language-based and vision-only methods on AUROC, and improves on average F1-max by 6.8 points at the pixel level and 1.6 points at the image level. On the same DINOv3 backbone, it improves pixel-level F1-max by 9.1 points over the language-free ZSAD (AnomalyVFM) while training on 1.7k real images rather than 10k synthetically generated ones.
- Our method remains accurate with very little data: trained on two images per class (3.5% of the auxiliary set), it, on average, outperforms all prior methods trained on the full set. To the best of our knowledge, this is the first work that shows zero-shot detection in a data-constrained regime.

2 Related Work

Classic anomaly detection methods learn what *normal* looks like for one product and flag anything that departs from it. Methods differ mainly in how they represent normality, reconstruction models flag regions they fail to reconstruct Bergmann et al. (2019b); Akcay et al. (2018); Zavrtanik et al. (2021), normalizing flows and density models flag features of low likelihood Gudovskiy et al. (2022); Yu et al. (2021); Rudolph et al. (2022), student–teacher networks flag regions where the two disagree Bergmann et al. (2020); Deng & Li (2022), and memory banks such as PaDiM Defard et al. (2021) and PatchCore Roth et al. (2022) flag patches far from every stored normal example. Whatever the representation, normality is learned per product, so each new product needs its own large number of normal images. Few-shot methods Huang et al. (2022); Gu et al. (2023) shrink this requirement to a handful of images, but do not remove it. Generalist methods You et al. (2022); Guo et al. (2025); Zhu & Pang (2024); Li et al. (2024) cover many products with a single model, yet still need target-product images at test time. Zero-shot anomaly detection (ZSAD) removes this dependence entirely; a model trained on some products must detect defects on products it has never seen.

Language as the reference. Most ZSAD methods replace the per-product model of normality with text. Built on CLIP Radford et al. (2021a), they score each image patch by comparing it with text embeddings of normal and defective states, so the definition of an anomaly lives in the text

encoder. Progress in this line has come largely from making that text reference richer. Hand-crafted prompts in WinCLIP Jeong et al. (2023), learned object-agnostic prompts in AnomalyCLIP Zhou et al. (2024), prompts conditioned on each image in AdaCLIP Cao et al. (2024), flexible abnormality prompts in FAPrompt Zhu et al. (2025), and a learned distribution over prompts in Bayes-PFL Qu et al. (2025), which reduces overfitting to the auxiliary data. Other work adapts the visual side so that patches better match the text, either by tuning both CLIP branches (AA-CLIP Ma et al. (2025)) or by separating foreground from background (FB-CLIP Hu et al. (2026)).

Text-based methods have three problems. They only work with vision–language models, which are trained to match a single global token per image with its caption. The patch features receive no direct training signal, so they capture small local defects poorly (Maninis et al., 2025). Their prompts reduce every defect to "normal" or "abnormal", but real defects vary without limit. Their learned prompts also tend to overfit the training data Qu et al. (2025); Hou et al. (2026). Learning directly from visual features solves the first two problems, and because we learn only a few directions, there is little to overfit.

Adapting vision foundation models. Recent methods remove language and instead build the reference within a vision foundation model (VFM), but both do so by changing the backbone. AnomalyVFM Fučka et al. (2026) changes its *weights* by adding trainable LoRA adapters and training them on a large set of synthetic defects. VisualAD Hou et al. (2026) changes its *inputs* by inserting two learnable tokens, for normal and abnormal, which mix with the image patches through self-attention and therefore alter the patch features themselves. Our work takes the remaining option. We change nothing inside the backbone and learn the reference directly in its output feature space, as a few directions for normal and defective patches. This rests on our finding that the frozen features already separate the two along a shared direction (section 3).

3 Preliminary Analysis

3.1 Latent Anomaly Geometry in Frozen VFMs

Before designing a method, we ask a basic question: does a frozen VFM already tell normal and defective patches apart?

Let $x_n^{(j)} \in \mathbb{R}^D$ denote the feature vector of patch $n \in \{1, \dots, N\}$ at transformer block j of a frozen vision backbone. We apply ℓ_2-normalization, $\hat{x}_n^{(j)} = x_n^{(j)} / \|x_n^{(j)}\|_2$, so that every patch lies on the unit hypersphere $\mathbb{S}^{D-1}$.

To answer the question, we use an *oracle* direction. Let $\mathcal{P}_{\text{normal}}$ and $\mathcal{P}_{\text{anomaly}}$ be the sets of normal and defective patches given by the pixel masks of a dataset. We average the normalized features of each set,

$$\mu_c = \frac{1}{|\mathcal{P}_c|} \sum_{n \in \mathcal{P}_c} \hat{x}_n^{(j)}, \quad c \in \{\text{normal}, \text{anomaly}\}, \tag{1}$$

and define e_1 as the unit direction between the two means:

$$e_1 = \frac{\mu_{\text{anomaly}} - \mu_{\text{normal}}}{\|\mu_{\text{anomaly}} - \mu_{\text{normal}}\|_2}. \tag{2}$$

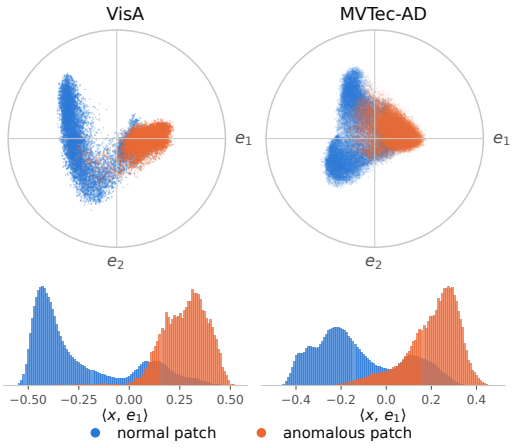


Figure 1: Patch features of a frozen DINOv3, Layer 24, projected onto the oracle direction e_1 and the display axis e_2.

This direction is an oracle because it uses the labels of the data it is tested on. It therefore tests whether the separation exists, not what a zero-shot method can achieve.

To visualize the features in two dimensions, we add a second axis e_2 that captures the largest remaining variation. We first remove the e_1 component from each patch,

$$r_n = \hat{x}_n^{(j)} - \langle \hat{x}_n^{(j)}, e_1 \rangle e_1, \tag{3}$$

mean-center the residuals, $\tilde{r}_n = r_n - \bar{r}$ with $\bar{r} = \frac{1}{N}\sum_{n=1}^{N} r_n$, and compute their covariance $C_r = \frac{1}{N}\sum_{n=1}^{N} \tilde{r}_n \tilde{r}_n^\top$. We take e_2 as the leading unit eigenvector of C_r, that is, $C_r e_2 = \lambda_{\max} e_2$. Since every residual is orthogonal to e_1, so is e_2. Each patch is then plotted at $\big(\langle \hat{x}_n^{(j)}, e_1 \rangle, \langle \hat{x}_n^{(j)}, e_2 \rangle\big)$.

Figure 1 shows that the answer is yes, on both MVTec-AD and VisA, normal and defective patches form two distinct groups along e_1, although the backbone was never trained for anomaly detection. Two questions remain. First, the oracle uses labels of the target data, which zero-shot detection does not have; we show in section 6 that directions learned on one dataset transfer to unseen ones. Second, one direction is not enough to cover every case; different kinds of defects can point in slightly different directions, and normal appearance varies too. Section 4 therefore learns several directions for normal and defective patches from auxiliary data only.

4 METHODOLOGY

Building on our preliminary analysis, which demonstrates that frozen Vision Foundation Models (VFMs) naturally separate normal and anomalous features by their angular orientation, we formulate zero-shot anomaly detection (ZSAD) as a problem of detecting the opponent directions on a unit hypersphere. By ℓ_2-normalizing high-dimensional embeddings, we intentionally discard the Euclidean norm, isolating angular direction as the sole carrier of semantic identity.

Because our tokens are ℓ_2-normalized, their radial extent is constant by construction and carries no information (Ledoit & Wolf, 2004). Therefore, our objective is to evaluate how well local patch tokens align with a set of learned reference directions for both normal and anomalous states. Crucially, we achieve this without fine-tuning the backbone or exposing the model to target-domain defects. These normalized directional embeddings, $\hat{x}_n^{(j)} \in \mathbb{S}^{D-1}$, form the foundation of our framework and serve as the direct inputs for the directional modeling (section 4.1) and scoring mechanisms that follow.

4.1 DIRECTIONAL MODELING VIA VON MISES-FISHER

For each selected transformer block $j \in J$, we define Q reference directions describing normal appearance and Q directions describing anomalous appearance. Let $\hat{d}_{j,q}^{c} \in \mathbb{S}^{D-1}$ denote the q^{th} reference direction for state c at block j. Each reference direction lies on the same unit sphere $\mathbb{S}^{D-1}$ as the features $\hat{x}_n^{(j)}$. During training, we treat it as a random direction drawn from a von Mises–Fisher (vMF) distribution (Fisher, 1953; Banerjee et al., 2005), the maximum-entropy distribution on the sphere for a given mean direction. The distribution is parameterized by a mean direction $\hat{\mu}_{j,q}^{c} \in \mathbb{S}^{D-1}$ and a concentration $\kappa_{j,q}^{c} \geq 0$. Dropping the indices for readability, a realization z of this random direction has density

$$p(z \mid \hat{\mu}, \kappa) = C_D(\kappa) \exp\left(\kappa \hat{\mu}^\top z\right), \quad \text{where} \quad C_D(\kappa) = \frac{\kappa^{D/2-1}}{(2\pi)^{D/2} I_{D/2-1}(\kappa)}. \tag{4}$$

Here, $C_D(\kappa)$ is the normalization constant, which makes the density integrate to one. Computing it requires $I_{D/2-1}(\kappa)$, the modified Bessel function of the first kind. This representation lets each reference direction express not only where it points ($\hat{\mu}$) but also how firmly it is specified (κ). Evaluating this Bessel function is the well-known difficulty of working with vMF densities in high dimension (Plesner et al., 2024). At $D = 1024$, neither direct evaluation nor the exponentially scaled form commonly used in practice stays representable across the concentrations our model learns (section A.2). Dedicated solvers exist for this regime (Plesner et al., 2024), but they target numerical evaluation rather than gradient-based optimization. Because $C_D(\kappa)$ enters our objective only through the regularizer, we never evaluate it directly and instead compute its logarithm with a closed-form approximation based on a uniform asymptotic expansion (eq. (9), section A.3).

Sampling Directions During Training. During training, we sample each direction instead of using it as a fixed point. A sample $z \sim \mathrm{vMF}(\hat{\mu}, \kappa)$ splits into a part along $\hat{\mu}$ and a part orthogonal to it:

$$z = w\hat{\mu} + \sqrt{1 - w^2}\, v, \qquad v \sim \mathrm{Uniform}\big(\{v \in \mathbb{S}^{D-1} : v^\top \hat{\mu} = 0\}\big), \tag{5}$$

where $w \in [-1, 1]$ is the cosine between z and $\hat{\mu}$, with density $p(w) \propto (1 - w^2)^{(D-3)/2} \exp(\kappa w)$. This density has no closed-form inverse cumulative distribution function, so we draw w by rejection sampling (Wood, 1994) (section A.4 algorithm 1).[1]

With the differentiable sampling mechanism established for a generic variable z, we can now instantiate the specific reference vectors required by our scoring objective. The reference direction $\hat{d}^c_{j,q}$ utilized in the training forward pass is assigned as a freshly drawn sample:

$$\hat{d}^c_{j,q} = z^c_{j,q} \sim \mathrm{vMF}\big(\hat{\mu}^c_{j,q}, \kappa^c_{j,q}\big). \tag{6}$$

By continuously perturbing $\hat{d}^c_{j,q}$ during training, the model learns robust angular boundaries resistant to overfitting.

Spherical Prior Regularization. The concentrations $\kappa^c_{j,q}$ are free parameters. Left unconstrained, the task loss drives them to grow without bound ($\kappa \to \infty$), collapsing each direction to a fixed point (Bishop & Nasrabadi, 2006; Davidson et al., 2018). To prevent this, we penalize the KL divergence between the vMF of each direction and the uniform distribution on the sphere, $\mathrm{Uniform}(\mathbb{S}^{D-1})$. Because the uniform distribution assigns equal density to every direction, this penalty depends only on κ (Li et al., 2025). It never pulls a reference direction away from where the data places it; it only keeps some spread unless the training loss favors concentrating (Davidson et al., 2018; Xu & Durrett, 2018):

$$D_{\mathrm{KL}}\left(\mathrm{vMF} \parallel \mathrm{Uniform}\right) = \kappa \frac{I_{D/2}(\kappa)}{I_{D/2-1}(\kappa)} + \log C_D(\kappa) + \log\left(\frac{2\pi^{D/2}}{\Gamma(D/2)}\right), \tag{7}$$

where the last term is the log surface area of the unit sphere. We denote by $\mathcal{L}^c_{\mathrm{KL}}$ this divergence averaged over the $K|\mathcal{L}|$ directions of state c.

At $D = 1024$, the Bessel functions in eq. (7) cannot be represented in floating point, so we replace them with two closed-form approximations. First, we use a closed-form upper bound on the Bessel ratio (Amos, 1974; Sra, 2012) (section A.4):

$$\frac{I_{D/2}(\kappa)}{I_{D/2-1}(\kappa)} \approx \sqrt{1 + \rho^2} - \rho, \quad \text{where} \quad \rho = \frac{D-1}{2\kappa}. \tag{8}$$

Second, we compute the logarithm of the Bessel function in $\log C_D(\kappa)$ with the uniform asymptotic expansion for large order (Olver, 1954; Abramowitz & Stegun, 1964) (section A.3). With $\nu = D/2 - 1$ and $s = \sqrt{\nu^2 + \kappa^2}$,

$$\log I_\nu(\kappa) \approx s + \nu \log\left(\frac{\kappa}{\nu + s}\right) - \frac{1}{2}\log(2\pi) - \frac{1}{2}\log s + \log\left(1 + \frac{3\kappa^2 - 2\nu^2}{24\, s^3}\right). \tag{9}$$

Both are built from elementary functions. Together, these approximations ensure accurate, overflow-free gradient flow through the KL divergence in high-dimensional latent spaces.

4.2 Dual-Branch Anomaly Scoring and Optimization

Residual alignment. Each token $\hat{x}^{(j)}_n$ is compared with the two sets of reference directions: the anomalous directions $\hat{d}^{\mathrm{anom}}_{j,q}$ and the normal directions $\hat{d}^{\mathrm{norm}}_{j,q}$. During training these are the sampled directions of eq. (6); at test time we use their means $\hat{\mu}^c_{j,q}$. The residual alignment score A_n is the difference between the token's strongest alignment with each set, averaged over the selected blocks and scaled by a learned inverse temperature τ_{scale}:

$$A_n = \frac{\tau_{\text{scale}}}{|J|} \sum_{j \in J} \left(\max_q \left\langle \hat{x}^{(j)}_n, \hat{d}^{\mathrm{anom}}_{j,q} \right\rangle - \max_q \left\langle \hat{x}^{(j)}_n, \hat{d}^{\mathrm{norm}}_{j,q} \right\rangle \right). \tag{10}$$

Intuitively, normal and defective tokens of the same object look largely alike. The alignment with the anomalous set alone therefore also rises for normal tokens. Subtracting the alignment with the normal set removes most of this shared part, and leaves what makes a token look defective.

[1] We use the rationalized form of Wood's constant b, which is algebraically equivalent and better conditioned for large κ.

Dual branches. Localization and detection are different objectives, so we keep two independent copies of this construction, one pair of normal and anomalous banks computes eq. (10) for the pixel branch, and a second, separately parameterized pair computes it for the image branch. The pixel branch places its scores (A_n) on the token grid and bilinearly upsamples them to the input resolution, giving the anomaly map M_{pixel}. The image branch sorts its own scores and averages the top η fraction into a single image score. This keeps small defects from being diluted, as they would be by averaging over all tokens (section 6.5).

Optimization. The backbone is frozen, so the only trainable parameters are the directions $\mu^c_{j,q}$, their concentrations $\kappa^c_{j,q}$, and the scale τ_{scale}, separately for each branch. The pixel branch is trained with Binary Focal (Lin et al., 2017) and Dice (Sudre et al., 2017) losses on the pixel scores, and the image branch with a Binary Focal loss on the pooled image score. Focal loss handles the imbalance between the few defective pixels and the many normal ones, while Dice loss rewards overlap of the whole defect region, which is a per-pixel loss does not. The total loss is

$$\mathcal{L} = \mathcal{L}^{\text{pix}}_{\text{Focal}} + \mathcal{L}^{\text{pix}}_{\text{Dice}} + \alpha\, \mathcal{L}^{\text{img}}_{\text{Focal}} + \beta \sum_{b \in \{\text{pix, img}\}} \left(\mathcal{L}^{\text{norm},b}_{\text{KL}} + \mathcal{L}^{\text{anom},b}_{\text{KL}}\right), \tag{11}$$

where $\mathcal{L}^{c,b}_{\text{KL}}$ is eq. (7) averaged over the directions of state c in branch b. Weight decay is applied to μ and τ_{scale} but not to $\log \kappa$, since decaying $\log \kappa$ would pull every concentration towards $\kappa = 1$, which is nearly uniform at $D = 1024$.

5 Experimental Setup

5.1 Datasets and Zero-Shot Protocol

We evaluate our framework across nine diverse anomaly detection benchmarks encompassing manufacturing defects and synthetic textures: MVTec AD (Bergmann et al., 2019a), VisA (Zou et al., 2022), BTAD (Mishra et al., 2021), MPDD (Jezek et al., 2021), KSDD (Tabernik et al., 2019), KSDD2 (Božič et al., 2021), DAGM (Wieler & Hahn, 2007), RealIAD (Wang et al., 2024) and DTD (Aota et al., 2023).

Following the standard strict Zero-Shot Anomaly Detection (ZSAD) (Zhou et al., 2024; Fučka et al., 2026; Hou et al., 2026; Hu et al., 2026), our model never observes any images from the target datasets during training. To learn universal boundaries for normality and abnormality, the directional banks are trained exclusively on an auxiliary dataset split. Specifically, when evaluating on MVTec-AD, the model is trained entirely on the VisA dataset; for all other benchmarks, the model is trained on MVTec-AD.

5.2 Competitors

We benchmark our framework against two distinct paradigms of current zero-shot state-of-the-art models: Language-Driven ZSAD: AdaCLIP (Cao et al., 2024), AA-CLIP (Ma et al., 2025), Bayes-PFL (Qu et al., 2025), and FB-CLIP (Hu et al., 2026). These models rely on Vision-Language Models (VLMs) and require explicit text prompts to localize anomalies. Language-Free Vision Adaptations: AnomalyVFM (Fučka et al., 2026) and VisualAD (Hou et al., 2026). These methods utilize Vision Foundation Models (VFMs) but rely on either parameter tuning (LoRA) or the injection of learnable prompt tokens directly into the attention mechanism. For language-free models, the backbone is changed to DINOV3/L16 as ours. All baselines are evaluated using their official checkpoints with default hyperparameters, except for VisualAD, as we tuned them for a new backbone.

5.3 Evaluation Metrics

Following (Fučka et al., 2026; Cao et al., 2024; Hou et al., 2026; Qu et al., 2025), Performance is measured at two granularities: anomaly localization (pixel-level) and anomaly detection (image-level). For both, we report the Area Under the Receiver Operating Characteristic curve (AUROC) and the maximum F1-score (F1-max).

5.4 Implementation Details and Hyperparameters

We employ a frozen pre-trained DINOv3 (ViT-L/16) backbone on 512×512 pixel inputs, extracting multi-scale patch features from transformer layers $J = \{16, 18, 24\}$. Across the four directional banks (pixel/image, normal/anomalous), we allocate $Q = 5$ learnable directions each, initializing concentration parameters to $\kappa = 100$. Directional parameters are optimized for 15 epochs using AdamW with a batch size of 16, applying parameter-specific learning rates of 10^{-2} for directional vectors ($\boldsymbol{\mu}$) and log-concentrations ($\log \kappa$), alongside 10^{-3} for the inverse temperature scalar (τ_{scale}). Weight decay is fixed at 10^{-2} for all parameters except $\log \kappa$, which strictly receives no decay to prevent artificial dragging toward the uniform prior. The joint objective balances the image-level branch with $\alpha = 0.8$ and the spherical KL divergence with $\beta = 0.001$. At inference, image-level anomaly scores are computed using top-k pooling with ratio $\eta = 0.02$, with all experiments conducted on a single 24 GB RTX 3090 GPU and 64 GB of RAM.

6 Results

6.1 Comparison with State-of-the-art

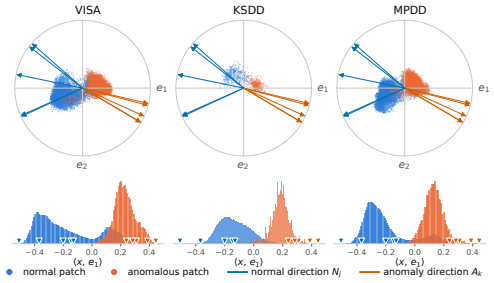
Table 1 compares our method with language-based and language-free methods on nine benchmarks. Our method gives the most precise defect maps; it raises the average pixel-level F1-max from 48.2 to 55.0, a gain of 6.8 points over the best prior method, and improves it on eight of the nine datasets. Pixel-level AUROC improves less (96.4 to 97.0), because most methods are already close to their upper limit. At the image level, our method raises the average AUROC from 92.2 to 93.8 and F1-max from 88.9 to 90.5. Furthermore, with the direct comparison to the best performing language-free models, we improve 7.6 points in pixel F1-max and 5.5 points in image F1-max. The qualitative results comparing the localization maps of the top-performing models are shown in Figure 2.

Table 1: Zero-shot anomaly detection results on pixel- and image-level metrics for Language Driven and Language-Free frameworks. *AnomalyVFM uses a large synthetic dataset (∼10,000 images, 100 objects), while others use standard auxiliary data (∼1,700 images, 15 objects). Ours averages 5 seeds (macro-average std dev: pixel AUROC ±0.08, pixel F1 ±0.43, image AUROC ±0.20, image F1 ±0.25).

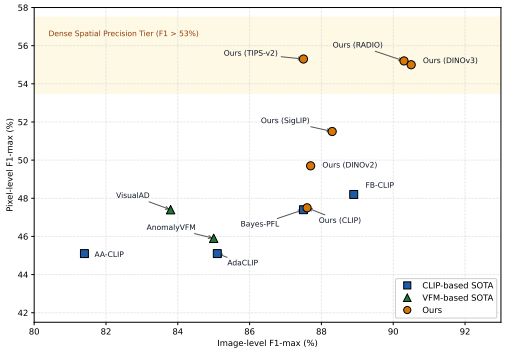
Datasets	AdaCLIP ECCV'24	AA-CLIP CVPR'25	Bayes-PFL CVPR'25	FB-CLIP CVPR'26	*AnomalyVFM CVPR'26	VisualAD CVPR'26	Ours
Pixel-level Results (AUROC, F1-max)							
MVTec-AD	(89.6, 43.9)	(91.4, 46.4)	(91.1, 49.3)	(92.1, 45.4)	(91.6, 49.2)	(89.3, 47.6)	(**92.4**, **53.4**)
VisA	(95.9, 36.4)	(94.8, 30.2)	(95.9, 34.5)	(96.2, 35.5)	(94.7, 31.3)	(93.2, 28.4)	(**96.3**, **37.2**)
MPDD	(95.6, 28.8)	(96.7, 30.0)	(**97.7**, 32.6)	(96.9, 31.9)	(96.7, 29.1)	(95.8, 27.9)	(96.2, **33.5**)
BTAD	(93.6, 49.0)	(96.3, 55.1)	(95.6, 52.3)	(96.1, 49.6)	(89.8, 45.0)	(95.6, 55.2)	(**96.8**, **56.6**)
KSDD	(89.8, 15.2)	(87.1, 28.0)	(97.6, 30.5)	(97.3, 47.4)	(96.4, 21.6)	(98.1, 35.4)	(**99.3**, **55.2**)
KSDD2	(98.5, 67.0)	(99.5, 63.4)	(96.2, 63.1)	(97.9, 64.4)	(98.9, 67.9)	(98.9, 61.7)	(**99.6**, **75.3**)
DAGM	(95.8, **64.4**)	(96.2, 53.3)	(96.3, 59.2)	(96.6, 59.1)	(**98.4**, 62.3)	(93.9, 56.2)	(95.1, 60.2)
DTD-Synthetic	(96.7, 59.3)	(95.8, 59.6)	(98.3, 64.7)	(98.2, 61.3)	(99.1, 68.3)	(98.8, 70.0)	(**99.9**, **74.2**)
RealIAD	(96.5, 41.6)	(96.4, 39.9)	(96.8, 40.8)	(96.7, 39.4)	(95.8, 38.5)	(**97.1**, 43.0)	(**97.1**, **49.0**)
Average	(94.7, 45.1)	(94.9, 45.1)	(96.2, 47.4)	(96.4, 48.2)	(95.7, 45.9)	(95.9, 47.4)	(**97.0**, **55.0**)
Image-level Results (AUROC, F1-max)							
MVTec-AD	(92.5, 93.0)	(90.5, 90.4)	(92.2, 92.8)	(92.1, 92.5)	(**93.5**, **93.8**)	(87.0, 90.6)	(92.6, **93.8**)
VisA	(79.6, 79.3)	(84.6, 78.8)	(87.0, 83.9)	(89.5, 85.3)	(89.4, 86.1)	(78.2, 79.7)	(**90.2**, **86.9**)
MPDD	(73.6, 79.9)	(75.1, 79.8)	(81.1, 83.8)	(76.8, 79.4)	(78.1, 83.7)	(70.3, 76.0)	(**82.8**, **84.9**)
BTAD	(94.0, 92.0)	(94.8, 93.7)	(94.2, 91.8)	(95.1, 93.4)	(95.3, 91.3)	(94.6, **93.8**)	(**97.0**, 91.8)
KSDD	(90.6, 71.4)	(69.3, 57.1)	(92.9, 76.1)	(97.0, 87.4)	(85.8, 53.6)	(92.0, 68.7)	(**98.2**, **90.1**)
KSDD2	(95.9, 86.7)	(95.9, 84.4)	(97.8, 89.7)	(**98.2**, **91.8**)	(96.5, 86.0)	(94.2, 86.8)	(97.8, 89.9)
DAGM	(98.7, 97.2)	(93.2, 79.4)	(99.2, 96.8)	(99.1, 97.4)	(99.3, 98.1)	(91.0, 89.1)	(**99.7**, **98.9**)
DTD-Synthetic	(92.7, 93.2)	(90.4, 92.8)	(93.8, 94.8)	(97.2, 96.2)	(99.2, 96.3)	(94.4, 94.7)	(**99.6**, **99.2**)
RealIAD	(79.2, 73.5)	(81.3, 76.4)	(85.2, 77.8)	(84.5, 76.9)	(83.6, 76.3)	(80.0, 74.6)	(**86.0**, **78.9**)
Average	(88.5, 85.1)	(86.1, 81.4)	(91.5, 87.5)	(92.2, 88.9)	(91.2, 85.0)	(86.9, 83.8)	(**93.8**, **90.5**)

6.2 Do the Directions Transfer Across Datasets?

Section 3 showed that an oracle direction, computed from each dataset's own labels, separates normal and defective patches. A zero-shot method cannot use such labels. Figure 3a tests whether the separation survives without them: the axes e_1 and e_2 and the learned directions all come from MVTec-AD alone, and are applied unchanged to VisA, KSDD, and MPDD. Across all three unseen datasets, normal and defective patches still fall on opposite sides of e_1, and the learned normal and



(a) Zero-shot directional projection across datasets.



(b) Localization vs. detection F1-max.

Figure 3: (a) Directions learned on MVTec-AD, applied to unseen datasets. The axes e_1, e_2, and the learned normal (blue) and anomalous (orange) directions are computed on MVTec-AD only; no labels of the target dataset are used. Bottom: scores along e_1, with triangles marking the learned directions. (b) Our method on six frozen backbones (circles) against language-based (squares) and language-free (triangles) prior methods. Averages over the nine target datasets.

anomalous directions point towards their respective matching groups. The ablation study (Oracle direction e1) also supports this claim and shows that, without training, the model's oracle direction can detect anomalies, whereas learned directions detect them more effectively. The same pattern holds for all six backbones (section B.2). The direction that separates defects is therefore shared across datasets, which is why directions learned on auxiliary data transfer to unseen products. It is worth mentioning that although the arrows appear overlapped in a 2D orthographic plot, they are far apart. Detailed geometric analysis is provided in section C.1.

6.3 Learning from Few Images

Because the separating direction already exists in the frozen backbone (section 3), our method should need little data to find it. Table 2 tests this, we train on K normal and K defective images per MVTec-AD class and test on the other eight datasets. With only two images per class, 60 images or 3.5% of the auxiliary set, our method already matches or exceeds the best prior method trained on the full set on all four metrics, and improves pixel-level F1-max from 48.6 to 52.7. With three images per class, pixel-level F1-max (55.3) approximately reaches the level of our fully trained model. With a single image per class, pixel-level results remain strong, but image-level results vary widely between draws of the training images.

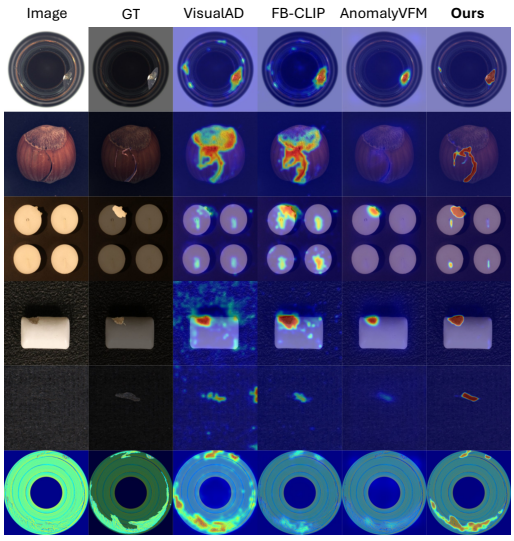


Figure 2: Qualitative comparison with SOTA. Rows: input image with the ground-truth defect outlined, followed by the anomaly maps of each method and ours. For display only, the scores of every method are normalized per class in the same way.

6.4 VFMs as Backbones

Because our method needs no text encoder, it can run on any frozen vision backbone. Figure 3b compares it on six backbones with prior methods. On CLIP, the backbone most language-based methods are built on, our method is on par with them in F1-max without using any text: its pixel-level F1-max of 47.5 lies next to Bayes-PFL (47.4) and just below FB-CLIP (48.2). With

Table 2: Training with few images. Our method is trained on K normal and K defective images per MVTec-AD class and tested on the other eight datasets. Mean ± standard deviation over three random draws of the training images.

shots	imgs	% of train data	pxAUROC	pxF1	imAUROC	imF1
1/class	30	1.7%	96.5 ± 0.55	50.4 ± 2.03	77.1 ± 14.47	77.4 ± 11.07
2/class	60	3.5%	97.0 ± 0.37	52.7 ± 1.07	92.3 ± 0.54	88.9 ± 0.49
3/class	90	5.2%	97.1 ± 0.13	55.3 ± 0.56	91.5 ± 0.47	89.0 ± 0.32
5/class	150	8.7%	97.1 ± 0.14	54.8 ± 0.50	92.9 ± 0.44	89.3 ± 0.27
10/class	300	17.4%	97.3 ± 0.14	55.1 ± 0.53	92.8 ± 0.42	89.7 ± 0.25
FULL	1,725	100.0%	97.5	55.2	93.9	90.1

Table 3: Ablation study. Models trained on MVTec-AD and evaluated zero-shot on eight industrial datasets (mean).

	Pixel-level		Image-level	
Method	AUROC	F1	AUROC	F1
Oracle direction e_1	95.7	45.0	84.9	82.1
w/o anomaly bank	97.0	53.1	86.1	84.5
w/o normal bank	97.1	54.1	92.4	88.8
Mean over bank	97.1	52.0	90.5	87.8
w/o KL term	96.9	51.6	92.8	89.3
Single layer (24)	96.6	52.4	91.4	87.6
Mean pooling (Detection)	97.5	55.2	87.8	84.4
Max pooling (Detection)	97.5	55.2	92.4	88.6
Full model	97.5	55.2	93.9	90.1

stronger vision backbones, it moves clearly ahead, on five of the six backbones, its pixel-level F1-max exceeds every prior method, and replacing CLIP with DINOv3 alone raises it from 47.5 to 55.0. With DINOv3 and RADIO, our method also leads at the image level (90.5 and 90.3 F1-max, against 88.9 for FB-CLIP). Per-dataset results and further analysis are given in section B.2.

6.5 Ablation Study

6.5.1 Learned directions.

A single oracle direction $e1$, computed from the MVTec auxiliary data, is 10.2 points below our method in pixel-level F1-max and 8.9 points in image-level AUROC (table 3). One direction is therefore not enough, as argued in section 3. Both sets of directions are needed, but not equally, removing the anomalous directions during evaluation lowers image-level AUROC by 7.7 points, while removing the normal ones lowers it by 1.4. Defects are thus best captured by explicit anomaly directions, not only by their distance from normal appearance.

6.5.2 Scoring.

Taking the best-matching direction in each set, rather than the average, adds 3.2 points of pixel-level F1-max and 3.4 points of image-level AUROC. Using one-layer features (layer 24) instead of multi-layer features lowers the pixel- and image-level F1 scores by 2.7 and 2.5 points, respectively. For the image score, averaging the top 2% of patch scores outperforms both the mean over all patches, which dilutes small defects (6.0 points in image AUROC), and the single maximum (1.4 points in image AUROC). The KL-divergence term also affects the learning process of directions. Without the KL-Divergence term, the κ rapidly collapses to a single point across the epochs, and it interferes with the sampling process, reducing 3.6 F1 pixel and 0.8 F1 image points.

6.6 Inference Efficiency

Our method adds almost no computational cost to the frozen backbone. It trains only 61.5K parameters, about 90 times fewer than the lightest prior method (FB-CLIP, 5.56M), and it needs the least compute (0.62 TFLOPs) and GPU memory (1.25 GB) of all methods. Its latency (63.5 ms per image) matches that of the language-free methods and is 1.5 to 4.6 times lower than that of language-based methods, which must also run a text encoder. Full results are given in section C.2.

7 Conclusion

We presented a language-free zero-shot anomaly detector that works directly on frozen vision foundation models. We showed that these models already separate normal and defective patches along a direction that is shared across datasets and backbones. Our method learns a few such directions for normal and for defective patches from auxiliary data, and applies them unchanged to unseen datasets. During training, each direction is randomly perturbed with a von Mises–Fisher sampler, which makes the defect maps sharper. The method uses no text, changes no backbone weights, and trains only 61.5K parameters. Across nine benchmarks, it improves the average pixel-level F1-max by 6.8 points and image-level F1 by 1.6 points over the best prior method, and when trained on only two images per class, it already matches or outperforms prior methods trained on the full data. Because it needs no text, it works with any vision backbone.

AI USE STATEMENT

In this work, we used generative AI tools to polish the writing, find related work, proofread the text, and format mathematical notation. Generative AI was also used to learn new concepts and to debug code. We did not use generative AI tools for ideation or experimentation. We have carefully reviewed all AI-assisted work by verifying against the literature, cross-checking concepts from multiple sources, and conducting dual proofreading by the authors. We take full responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, volume 55 of *Applied Mathematics Series*. National Bureau of Standards, Washington, D.C., 1964.

Samet Akcay, Amir Atapour-Abarghouei, and Toby P. Breckon. Ganomaly: Semi-Supervised Anomaly Detection via Adversarial Training. In *Asian Conference on Computer Vision (ACCV)*, pp. 622–637, 2018.

Donald E Amos. Computation of modified Bessel functions and their ratios. *Mathematics of computation*, 28(125):239–251, 1974.

Toshimichi Aota, Lloyd Teh Tzer Tong, and Takayuki Okatani. Zero-Shot Versus Many-Shot: Unsupervised Texture Anomaly Detection. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023.

Arindam Banerjee, Inderjit S Dhillon, Joydeep Ghosh, Suvrit Sra, and Greg Ridgeway. Clustering on the unit hypersphere using von mises-Fisher distributions. *Journal of Machine Learning Research*, 6(9), 2005.

Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019a.

Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving Unsupervised Defect Segmentation by Applying Structural Similarity to Autoencoders. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*, pp. 372–380, 2019b.

Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed Students: Student-Teacher Anomaly Detection with Discriminative Latent Embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4183–4192, 2020.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Jakob Božič, Domen Tabernik, and Danijel Skočaj. Mixed supervision for surface-defect detection: From weakly to fully supervised learning. *Computers in Industry*, 129:103459, 2021.

Bingyi Cao, Koert Chen, Kevis-Kokitsi Maninis, Kaifeng Chen, Arjun Karpur, Ye Xia, Sahil Dua, Tanmaya Dabral, Guangxing Han, Bohyung Han, Joshua Ainslie, Alex Bewley, Mithun Jacob, René Wagner, Washington Ramos, Krzysztof Choromanski, Mojtaba Seyedhosseini, Howard Zhou, and Andre Araujo. Tipsv2: Advancing vision-language pretraining with enhanced patch-text alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29325–29335, June 2026.

Yunkang Cao, Jiangning Zhang, Luca Frittoli, Yuqi Cheng, Weiming Shen, and Giacomo Boracchi. Adaclip: Adapting clip with hybrid learnable prompts for zero-shot anomaly detection. In *European conference on computer vision*, pp. 55–72. Springer, 2024.

Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. Hyperspherical Variational Auto-Encoders. In Amir Globerson and Ricardo Silva (eds.), *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI 2018)*, pp. 856–865. AUAI Press, 2018. URL http://auai.org/uai2018/proceedings/papers/309.pdf.

Nicola De Cao and Wilker Aziz. The Power Spherical distrbution. *Proceedings of the 37th International Conference on Machine Learning, INNF+*, 2020.

Thomas Defard, Aleksandr Setkov, Angel Angelov, Chethan Li, and Abhinav Dhall. Padim: A Patch Distribution Modeling Framework for Anomaly Detection and Localization. In *International Conference on Pattern Recognition (ICPR)*, pp. 475–489, 2021.

Hanqiu Deng and Xingyu Li. Anomaly Detection via Reverse Distillation from One-Class Embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9737–9746, 2022.

Ronald Fisher. Dispersion on a sphere. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, pp. 295–305, 1953.

Matic Fučka, Vitjan Zavrtanik, and Danijel Skočaj. Anomalyvfm – Transforming Vision Foundation Models into Zero-Shot Anomaly Detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 35555–35566, 6 2026.

Zhihao Gu, Taihai Yang, and Lizhuang Ma. Prototype-Aware Contrastive Knowledge Distillation for Few-Shot Anomaly Detection. In *Proceedings of the 34th British Machine Vision Conference (BMVC)*, 2023.

Denis Gudovskiy, Shun Ishizaka, and Kazuki Kozuka. C-FLOW-AD: Real-Time Unsupervised Anomaly Detection with Localization via Conditional Normalizing Flows. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 98–107, 2022.

Jia Guo, Shuai Lu, Weihang Zhang, Fang Chen, Huiqi Li, and Hongen Liao. Dinomaly: The less is more philosophy in multi-class unsupervised anomaly detection. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20405–20415. IEEE, 2025.

Yanning Hou, Peiyuan Li, Zirui Liu, Yitong Wang, Yanran Ruan, Jianfeng Qiu, and Ke Xu. Visualad: Language-Free Zero-Shot Anomaly Detection via Vision Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21346–21356, 6 2026.

Ming Hu, Yongsheng Huo, Mingyu Dou, Jianfu Yin, Peng Zhao, Yao Wang, Cong Hu, Bingliang Hu, and Quan Wang. Fb-CLIP: Fine-Grained Zero-Shot Anomaly Detection with Foreground-Background Disentanglement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 35659–35669, 6 2026.

Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yanfeng Wang. Registration Based Few-Shot Anomaly Detection. In *European Conference on Computer Vision (ECCV)*, pp. 303–319, 2022.

Jongheon Jeong, Yang Zou, Taewan Kim, Dongqing Zhang, Avinash Ravichandran, and Onkar Dabeer. Winclip: Zero-/few-shot anomaly classification and segmentation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19606–19616. IEEE, 2023.

Stepan Jezek, Martin Jonak, Radim Burget, Pavel Dvorak, and Milos Skotak. Deep learning-based defect detection of metal parts: evaluating current methods in complex conditions. In *13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, pp. 66–71, 2021.

Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis*, 88(2):365–411, 2004.

Yiting Li, Adam Goodge, Fayao Liu, and Chuan-Sheng Foo. Promptad: Zero-shot anomaly detection using text prompts. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 1093–1102, 2024.

Zhixiang Li, Zhiwen Luo, Nizar Bouguila, Weifeng Su, and Wentao Fan. Disentangled representation learning for multi-view clustering via von Mises–Fisher hyperspherical embedding. *Neural Networks*, 191:107802, 2025.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Wenxin Ma, Xu Zhang, Qingsong Yao, Fenghe Tang, Chenxu Wu, Yingtai Li, Rui Yan, Zihang Jiang, and S Kevin Zhou. Aa-clip: Enhancing zero-shot anomaly detection via anomaly-aware clip. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4744–4754. IEEE, 2025.

Kevis-Kokitsi Maninis, Kaifeng Chen, Soham Ghosh, Arjun Karpur, Koert Chen, Ye Xia, Bingyi Cao, Daniel Salz, Guangxing Han, Jan Dlabal, and others. Tips: Text-image pretraining with spatial awareness. In *International Conference on Learning Representations*, volume 2025, pp. 64256–64279, 2025.

Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. Vt-ADL: A vision transformer network for image anomaly detection and localization. In *2021 IEEE 30th international symposium on industrial electronics (ISIE)*, pp. 01–06. IEEE, 2021.

Christian Naesseth, Francisco Ruiz, Scott Linderman, and David Blei. Reparameterization gradients through acceptance-rejection sampling algorithms. In *Artificial Intelligence and Statistics*, pp. 489–498. PMLR, 2017.

F. W. J. Olver. The Asymptotic Expansion of Bessel Functions of Large Order. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 247 (930):328–368, 1954.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Andreas Plesner, Hans Henrik Brandenborg Sørensen, and Søren Hauberg. Accurate computation of the logarithm of modified bessel functions on gpus. In *Proceedings of the 38th ACM International Conference on Supercomputing*, pp. 213–224, 2024.

Zhen Qu, Xian Tao, Xinyi Gong, Shichen Qu, Qiyu Chen, Zhengtao Zhang, Xingang Wang, and Guiguang Ding. Bayesian prompt flow learning for zero-shot anomaly detection. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 30398–30408. IEEE, 2025.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, pp. 8748–8763, 2021a.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and others. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021b.

Mike Ranzinger, Greg Heinrich, Jan Kautz, and Pavlo Molchanov. Am-radio: Agglomerative vision foundation model reduce all domains into one. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12490–12500. IEEE, 2024.

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)*, pp. 14318–14328, 2022.

Marco Rudolph, Tom Wehrbein, Bodo Rosenhahn, and Bastian Wandt. Fully Convolutional Cross-Scale-Flows for Image-Based Defect Detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1088–1097, 2022.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, and others. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Suvrit Sra. A short note on parameter approximation for von Mises-Fisher distributions: and a fast implementation of I s (x). *Computational Statistics*, 27(1):177–190, 2012.

Carole H Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *International Workshop on Deep Learning in Medical Image Analysis*, pp. 240–248. Springer, 2017.

Domen Tabernik, Samo Šela, Jure Skvarč, and Danijel Skočaj. Segmentation-based deep-learning approach for surface-defect detection. *Journal of Intelligent Manufacturing*, 31(3):759–776, 2019.

Chengjie Wang, Wenbing Zhu, Bin-Bin Gao, Zhenye Gan, Jiangning Zhang, Zhihao Gu, Shuguang Qian, Mingang Chen, and Lizhuang Ma. Real-IAD: A Real-World Multi-View Dataset for Benchmarking Versatile Industrial Anomaly Detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Matthias Wieler and Tobias Hahn. Weakly Supervised Learning for Industrial Optical Inspection. In *DAGM Symposium*, 2007.

Andrew TA Wood. Simulation of the von Mises Fisher distribution. *Communications in statistics-simulation and computation*, 23(1):157–164, 1994.

Jiacheng Xu and Greg Durrett. Spherical latent spaces for stable variational autoencoders. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 4503–4513, 2018.

Zhiyuan You, Lei Cui, Yujun Shen, Kai Yang, Xin Lu, Yu Zheng, and Xinyi Le. A unified model for multi-class anomaly detection. *Advances in Neural Information Processing Systems*, 35:4571–4584, 2022.

Jiawei Yu, Ye Zheng, Xiang Wang, Wei Li, Yida Shen, and Chunhua Zhao. Fastflow: Unsupervised Anomaly Detection and Localization via 2d Normalizing Flows. *arXiv preprint arXiv:2111.07677*, 2021.

Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Draem – A Discriminatively Trained Reconstruction Embedding for Surface Anomaly Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8330–8339, 2021.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11941–11952. IEEE, 2023.

Qihang Zhou, Guansong Pang, Yu Tian, Shibo He, and Jiming Chen. Anomalyclip: Object-agnostic prompt learning for zero-shot anomaly detection. In *International Conference on Learning Representations*, volume 2024, pp. 49705–49737, 2024.

Jiawen Zhu and Guansong Pang. Toward generalist anomaly detection via in-context residual learning with few-shot sample prompts. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17826–17836. IEEE, 2024.

Jiawen Zhu, Yew-Soon Ong, Chunhua Shen, and Guansong Pang. Fine-grained abnormality prompt learning for zero-shot anomaly detection. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22241–22251. IEEE, 2025.

Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-Difference Self-supervised Pre-training for Anomaly Detection and Segmentation. In *European Conference on Computer Vision (ECCV)*, 2022.

A Numerical Treatment of the von Mises–Fisher Distribution

Every quantity in section 4.1 that involves the normalizer $C_D(\kappa)$ depends on the modified Bessel function $I_{D/2-1}(\kappa)$. At the dimension we work in, this function cannot be evaluated on a computer at all. This appendix explains why, gives the two closed forms we use instead, and reports how accurate they are.

A.1 Why the concentrations are large

It is worth first explaining a number that looks surprising. Our learned concentrations reach 2×10^5, whereas hyperspherical latent-variable models in the literature typically report κ in the tens or hundreds. This is not a difference in how tightly our directions are specified. It is a consequence of dimension.

To make κ easier to read, we convert it into an angle. We measure the angle between each sample and the mean direction, and report the typical angle, $\theta = \arccos\big(A_D(\kappa)\big)$, where $A_D(\kappa)$ is the average cosine between them. For large κ, this angle is about $\sqrt{(D-1)/\kappa}$. Keeping the same angle in a higher dimension, therefore needs a proportionally larger κ:

	Angular spread			
κ	$D = 3$	$D = 64$	$D = 512$	$D = 1024$
1×10^2	$8.1°$	$42.8°$	$79.1°$	$84.4°$
1×10^3	$2.6°$	$14.3°$	$39.0°$	$52.3°$
1×10^4	$0.8°$	$4.5°$	$12.9°$	$18.2°$
2×10^5	$0.2°$	$1.0°$	$2.9°$	$4.1°$

At $D = 3$, $\kappa = 100$ is a tight spike. At $D = 1024$ it is spread over $84°$, which is almost indistinguishable from uniform. The intuition is that in high dimensions nearly all of a sphere's surface lies close to the equator of any chosen axis, so holding samples inside a narrow cone means working against a great deal of available area.

Expressed as angles, the concentrations our model learns:

	κ	Spread
Initialization	100	$84.4°$
Learned minimum	360	$71.5°$
Learned median	12,263	$16.4°$
Learned maximum	197,046	$4.1°$

We report angular spread alongside κ wherever concentrations are discussed, since the raw parameter is not comparable across dimensions. Note also that the $547\times$ range in κ corresponds to only a $17\times$ range in angle.

A.2 Why we never evaluate the Bessel function

The values involved are far outside what floating-point arithmetic can hold. With $\nu = D/2 - 1 = 511$, we computed $I_\nu(\kappa)$ in exact arithmetic at the two ends of our range:

$$I_{511}(10^2) \approx 2.83 \times 10^{-294}, \qquad I_{511}(2 \times 10^5) \approx 3.66 \times 10^{86855}. \tag{12}$$

A single-precision float spans roughly 10^{-308} to 10^{308}. The first value is within a few orders of magnitude of underflowing to zero; the second exceeds the largest representable number by more than eighty-six thousand orders of magnitude.

The usual remedy is the exponentially scaled function $\tilde{I}_\nu(\kappa) = I_\nu(\kappa)e^{-\kappa}$, which removes the dominant growth and is what most implementations use. At the dimensions common in hyperspherical latent-variable models, a few dozen, this is entirely sufficient. At ours it is not. We measured it directly in single precision:

κ	$\tilde{I}_{511}(\kappa)$	Usable?
1×10^1	0 exactly	No
1×10^2	0 exactly	No
3.6×10^2	0 exactly	No
1×10^3	0 exactly	No
1×10^4	8.55×10^{-9}	Yes
2×10^5	4.64×10^{-4}	Yes

Below $\kappa \approx 10^3$ the scaled value is not small but exactly zero, because $I_{511}(100)e^{-100}$ falls beneath the smallest positive double. Its logarithm is then undefined. This matters because the range is one our model occupies: the least concentrated direction in the trained model sits at $\kappa = 360$.

We therefore work entirely in logarithms and never evaluate the Bessel function at any point. The two quantities the objective requires, $\log C_D(\kappa)$ and the ratio $A_D(\kappa) = I_{D/2}(\kappa)/I_{D/2-1}(\kappa)$, are each obtained from a separate closed form, given in the next two sections. Both are built from elementary functions, so they are differentiable by automatic differentiation without any custom gradient implementation.

A.3 The logarithm of the Bessel function

For $\log I_\nu(\kappa)$ we use the uniform asymptotic expansion for large order (Olver, 1954; Abramowitz & Stegun, 1964), truncated after the first correction term. This expansion is the right choice here because it is accurate uniformly in the ratio κ/ν, so one formula covers concentrations from 10^2 to 10^6 with no case analysis and no switching between regimes. Writing $\nu = D/2 - 1$ and $s = \sqrt{\nu^2 + \kappa^2}$ gives eq. (9) in the main text. Because the expression is built only from square roots and logarithms, its derivative with respect to κ is supplied by automatic differentiation without any hand-written gradient.

The constant in eq. (9) appears as $\frac{1}{2}\log(2\pi)$ rather than the $\frac{1}{2}\log(2\pi\nu)$ of the form usually quoted. The two are the same expression: writing $t = \kappa/\nu$, the prefactor term satisfies $-\frac{1}{4}\log(1+t^2) = -\frac{1}{4}\log(s^2) + \frac{1}{2}\log\nu$, and the $\frac{1}{2}\log\nu$ cancels against the ν inside $\frac{1}{2}\log(2\pi\nu)$. We note this so the formula is not read as a misprint of the standard one.

We compared the expansion against a sixty-digit reference at $D = 1024$. The absolute error is the quantity that matters, since $\log C_D(\kappa)$ enters the divergence additively:

κ	exact $\log I_\nu(\kappa)$	expansion	absolute error
1×10^2	-675.918527	-675.918527	2.2×10^{-8}
1×10^3	867.649969	867.649969	1.2×10^{-9}
1×10^4	9981.422041	9981.422041	6.9×10^{-10}
2×10^5	199992.325222	199992.325222	$< 10^{-16}$
1×10^6	999992.042746	999992.042746	1.2×10^{-10}

The largest error, about 2×10^{-8}, occurs at the least concentrated end and is roughly eleven significant figures of agreement. Across the concentrations our trained model occupies the expansion agrees to twelve significant figures or better. Both are far below any scale the optimiser can resolve, the KL term is weighted by $\beta = 10^{-3}$ and the gradients that reach κ are themselves of order 10^{-8} or smaller (section A.5), so an error of 10^{-8} in $\log C_D(\kappa)$ has no effect on training.

A.4 THE RATIO OF BESSEL FUNCTIONS

The KL divergence in eq. (7) requires the ratio $A_D(\kappa) = I_{D/2}(\kappa)/I_{D/2-1}(\kappa)$, known as the mean resultant length. It measures how tightly samples cluster around the mean direction, running from 0 for the uniform distribution to 1 for a point mass. We evaluate it through a closed form (Amos, 1974; Sra, 2012) rather than by computing two Bessel functions and dividing, so that no Bessel function is evaluated anywhere in our implementation.

The shape of the expression follows from the standard recurrence for modified Bessel functions, taken at order $\nu + 1$:

$$I_\nu(\kappa) - I_{\nu+2}(\kappa) = \frac{2(\nu+1)}{\kappa}\, I_{\nu+1}(\kappa). \tag{13}$$

Dividing through by $I_{\nu+1}(\kappa)$ and writing $R_\nu = I_{\nu+1}(\kappa)/I_\nu(\kappa)$ for the successive ratio gives $R_\nu^{-1} - R_{\nu+1} = 2(\nu+1)/\kappa$, or equivalently

$$R_\nu = \left(\frac{2(\nu+1)}{\kappa} + R_{\nu+1}\right)^{-1}. \tag{14}$$

Consecutive ratios change very slowly at high order, so substituting $R_{\nu+1} \approx R_\nu$ turns this recursion into a quadratic, whose positive root is $\sqrt{1+\rho^2} - \rho$. Our target is $A_D(\kappa) = R_{D/2-1}$, so matching $\nu + 1 = D/2$ in this self-consistent solution would give $\rho = D/(2\kappa)$.

We do not use that value. Amos (1974) establishes a tighter bound for this ratio, obtained with the order shifted by one half, $\alpha - \frac{1}{2} = \frac{D-1}{2}$ for $\alpha = D/2$, and it is this form we adopt:

$$A_D(\kappa) \;\approx\; \sqrt{1+\rho^2} - \rho, \qquad \rho = \frac{D-1}{2\kappa}. \tag{15}$$

The half-unit shift matters more than its size suggests. The two constants are comparable at $\kappa = 10^3$, with relative errors of 2.0×10^{-4} and 2.4×10^{-4}, but the Amos form becomes progressively more accurate above that: at $\kappa = 2 \times 10^5$ it reaches 6.4×10^{-9} against 2.5×10^{-6} for the uncorrected fixed point. Since our trained concentrations lie between 3.6×10^2 and 2.0×10^5, this is the regime that matters. We draw the distinction explicitly so that eq. (15) is not mistaken for a transcription slip in the recurrence result above.

A further property follows from eq. (15) being an upper bound rather than a two-sided approximation: the error has a known sign. The expression never underestimates $A_D(\kappa)$, and since the ratio enters the divergence as $\kappa A_D(\kappa)$, the KL penalty is never understated as well. Any error therefore acts in the conservative direction, discouraging rather than permitting over-concentration.

We compared eq. (15) against a sixty-digit reference at $D = 1024$:

κ	exact $A_D(\kappa)$	bound	relative error
1×10^1	0.0097646957	0.0097742372	9.8×10^{-4}
1×10^2	0.0967439949	0.0968350895	9.4×10^{-4}
1×10^3	0.6115999686	0.6117240427	2.0×10^{-4}
1×10^4	0.9501548828	0.9501573067	2.6×10^{-6}
2×10^5	0.9974457640	0.9974457704	6.4×10^{-9}
1×10^6	0.9994886306	0.9994886308	2.6×10^{-10}

The bound is least accurate exactly where accuracy matters least. Its largest error, roughly one part in a thousand, occurs at concentrations so low that the distribution is already close to uniform and the KL penalty is correspondingly small. Across the range the model actually occupies it is accurate to eight significant digits or better.

A.5 HOW GRADIENTS REACH THE CONCENTRATION

While the concentration κ enters the KL divergence directly in closed form, it influences the task loss exclusively through the sampled reference directions. It is therefore worth setting out exactly

Algorithm 1 Rejection sampling for w

Require: Feature dimension D, concentration κ, maximum draws M

$b \leftarrow \frac{D-1}{2\kappa+\sqrt{4\kappa^2+(D-1)^2}}$
$x_0 \leftarrow \frac{1-b}{1+b}$
$c_0 \leftarrow \kappa x_0 + (D-1)\ln(1-x_0^2)$
$a \leftarrow \frac{D-1}{2}$
$w \leftarrow x_0$ {fallback, returned if no draw is accepted}
for $i = 1$ **to** M **do**
  Sample $y \sim \mathrm{Beta}(a, a)$
  Candidate: $w^* \leftarrow \frac{1-(1+b)y}{1-(1-b)y}$
  Sample $u \sim \mathrm{Uniform}(0, 1)$
  Acceptance statistic: $A \leftarrow \kappa w^* + (D-1)\ln(1-x_0 w^*) - c_0$
  if $A \geq \ln(u)$ **then**
    $w \leftarrow w^*$
    break
  end if
end for
return w

how pathwise gradients reach κ during the forward pass, and putting concrete numbers on an approximation that is standard in directional deep learning but has not previously been quantified at high dimension.

The sampled scalar projection w is not treated as a detached constant. In algorithm 1, the candidate draw

$$w^* = \frac{1-(1+b)y}{1-(1-b)y}, \qquad b = \frac{D-1}{2\kappa + \sqrt{4\kappa^2 + (D-1)^2}}, \tag{16}$$

is a smooth, differentiable function of b, which in turn is a differentiable function of κ. Because the proposal variates $y \sim \mathrm{Beta}(a, a)$ and $u \sim \mathrm{Uniform}(0, 1)$ are independent of the distribution parameters, gradients backpropagate into κ along the standard pathwise route through $w^*\hat{\mu}$. The only operation that interrupts the computational graph is the discrete accept/reject decision, which carries no gradient.

Wood's proposal is built from a symmetric $\mathrm{Beta}\big(\frac{D-1}{2}, \frac{D-1}{2}\big)$ variable y, which keeps the acceptance rate stable in high dimension (Wood, 1994; Davidson et al., 2018). The random variables y, the uniform acceptance variable u, and v in eq. (5) do not depend on κ; κ enters only through b in the candidate w^*. Gradients therefore reach κ, like $\hat{\mu}$, by the ordinary pathwise route, and only the Boolean accept decision, which carries no gradient, is treated as a constant.

The omitted correction term. Because the acceptance probability of the rejection sampler depends on κ, the distribution of accepted draws is not strictly a pushforward of the proposal density. An exactly unbiased gradient estimator requires an additional score-function correction term (Naesseth et al., 2017). In practice, this term exhibits high variance, leading Davidson et al. (2018) to omit it when training hyperspherical VAEs. While De Cao & Aziz (2020) cite this uncorrected bias as a conceptual drawback of vMF reparameterization, the actual magnitude of this bias has remained unmeasured in high dimensions ($D \gg 1$).

Quantifying the gradient bias. Because the vMF density belongs to the exponential family with natural parameter κ and sufficient statistic w, the true gradient of the expected projection is given by the variance:

$$\frac{\mathrm{d}\mathbb{E}[w]}{\mathrm{d}\kappa} = \mathrm{Var}(w) = 1 - \frac{D-1}{\kappa} A_D(\kappa) - A_D(\kappa)^2. \tag{17}$$

We measure the empirical gradient computed by automatic differentiation across algorithm 1 (without the score correction) over 2×10^5 draws at $D = 1024$, and compare it against the exact theoretical derivative:

κ	autograd $\mathrm{d}\mathbb{E}[w]/\mathrm{d}\kappa$	exact $\mathrm{Var}(w)$	relative error
1×10^2	9.494×10^{-4}	9.504×10^{-4}	0.10%
1×10^3	2.785×10^{-4}	2.786×10^{-4}	0.02%
1×10^4	4.855×10^{-6}	4.854×10^{-6}	0.03%
2×10^5	1.2756×10^{-8}	1.2755×10^{-8}	0.01%

The uncorrected pathwise gradient matches the true analytical derivative to within one part in a thousand at low concentrations, and to within 0.03% across our operating regime. At foundation-model dimensions ($D = 1024$), the bias introduced by omitting the score-function correction is practically negligible.

To verify that this gradient provides an active learning signal, we train the model with the spherical prior disabled ($\beta = 0$). Initialized at $\kappa = 100$, the mean concentration increases to 549 after one epoch and reaches 9.1×10^5 by epoch 15. The task loss actively drives reference directions to concentrate, while the spherical regularizer (eq. (7)) penalizes low entropy; their equilibrium determines the learned angular dispersion.

Acceptance efficiency. Contrary to concerns that rejection sampling stalls at large κ (De Cao & Aziz, 2020), the acceptance rate remains stable: the first-draw acceptance frequency is 99.6% at $\kappa = 10^2$, 85.5% at 10^3, 72.5% at 10^4, and 71.1% at 2×10^5. The sampler terminates within two draws across all tested configurations.

B Backbone Analysis

Our method needs no text encoder, so it can run on any frozen vision backbone. This section compares six of them, shows that the separating direction of section 3 appears in all six, and looks at the directions the method learns.

B.1 Setup

Table 4 lists the backbones. They cover three kinds of pre-training: self-supervised learning (DINOv2 (Oquab et al., 2023), DINOv3(Siméoni et al., 2025)), image–text alignment (CLIP(Radford et al., 2021a), SigLIP(Zhai et al., 2023), TIPS-v2(Cao et al., 2026)), and distillation from several teacher models (RADIO(Ranzinger et al., 2024)). Five backbones are ViT-L; SigLIP uses the comparable SoViT-400m, and all are used with the same hyperparameters and training protocol as in the main experiments. For SigLIP, which has 27 blocks instead of 24, we use blocks at the same relative depths.

Table 4: Frozen backbones used with our method. Blocks are given as used / total; for SigLIP they are chosen at the same relative depths as for the 24-block models.

Backbone	Variant	Patch	D	Blocks $\mathcal{L}$	Resolution	Pre-training
DINOv3	ViT-L/16	16	1024	{16, 18, 24} / 24	512^2	Self-supervised
DINOv2	ViT-L/14	14	1024	{16, 18, 24} / 24	518^2	Self-supervised
RADIO	C-RADIOv3-L (ViT-L/16)	16	1024	{16, 18, 24} / 24	512^2	Multi-teacher distillation
TIPS-v2	ViT-L/14	14	1024	{16, 18, 24} / 24	518^2	Vision–language (spatial)
SigLIP	SoViT-400m/14	14	1152	{18, 20, 27} / 27	518^2	Vision–language
CLIP	ViT-L/14	14	1024	{16, 18, 24} / 24	518^2	Vision–language

B.2 Results

Results on all nine datasets are reported in table 5. DINOv3 and RADIO give the best overall results. DINOv3 is first, and RADIO is second in pixel-level AUROC (97.0 and 96.8), image-level AUROC (93.8 and 92.5), and image-level F1-max (90.5 and 90.3). In pixel-level F1-max, TIPS-v2 is first (55.3), RADIO is second (55.2), and DINOv3 is third (55.0), while SigLIP, DINOv2, and CLIP reach 51.5, 49.7, and 47.5. This suggests that our method benefits most from newer backbones whose training puts more emphasis on dense patch-level features.

Table 5: Backbone comparison under the cross-dataset zero-shot protocol.

Datasets	DINOv3	DINOv2	TIPS-v2	SigLIP	CLIP	RADIO
Pixel-level Results (AUROC, F1-max)						
MVTec-AD	(92.4, 53.4)	(91.6, 50.7)	(91.3, 51.6)	(89.9, 45.3)	(90.1, 42.8)	(**92.5**, **54.0**)
VisA	(**96.3**, 37.2)	(95.7, 37.2)	(95.4, 35.0)	(95.1, 32.8)	(95.4, 30.2)	(96.1, **38.5**)
MPDD	(96.2, 33.5)	(**96.7**, 37.2)	(96.6, **37.9**)	(96.5, 36.3)	(96.5, 33.6)	(**96.7**, 36.1)
BTAD	(**96.8**, 56.6)	(94.6, 49.8)	(93.4, 56.0)	(91.5, 52.6)	(90.8, 44.4)	(95.9, **57.3**)
KSDD	(**99.3**, **55.2**)	(99.0, 42.1)	(98.9, 52.1)	(97.1, 50.4)	(96.9, 52.2)	(99.1, 51.0)
KSDD2	(**99.6**, **75.3**)	(99.2, 73.0)	(99.2, 73.0)	(95.7, 61.5)	(96.6, 65.7)	(99.2, 72.1)
DAGM	(**95.1**, 60.2)	(89.3, 44.1)	(94.2, **64.8**)	(94.7, 62.8)	(90.2, 51.0)	(94.6, 62.6)
DTD-Synthetic	(**99.9**, 74.2)	(96.7, 64.5)	(98.9, 77.9)	(98.6, 74.2)	(96.6, 65.7)	(99.2, **78.1**)
RealIAD	(97.1, 49.0)	(97.4, 48.5)	(97.5, **49.3**)	(97.2, 47.8)	(96.9, 42.2)	(**97.7**, 47.4)
Average	(**97.0**, 55.0)	(95.6, 49.7)	(96.2, **55.3**)	(95.1, 51.5)	(94.4, 47.5)	(96.8, 55.2)
Image-level Results (AUROC, F1-max)						
MVTec-AD	(92.6, **93.8**)	(91.3, 92.7)	(**92.8**, 92.8)	(92.5, 93.2)	(88.9, 91.4)	(92.7, 93.6)
VisA	(**90.2**, **86.9**)	(86.7, 84.1)	(85.3, 82.7)	(87.1, 84.5)	(83.7, 81.6)	(88.5, 85.0)
MPDD	(**82.8**, **84.9**)	(77.8, 82.3)	(74.6, 82.6)	(75.3, 83.2)	(72.4, 79.7)	(75.7, 83.0)
BTAD	(**97.0**, **91.8**)	(90.8, 86.9)	(93.7, 90.0)	(92.8, 88.1)	(92.4, 90.1)	(93.4, 89.7)
KSDD	(**98.2**, 90.1)	(95.9, 80.7)	(94.2, 76.6)	(96.4, 86.2)	(97.6, 89.8)	(97.8, **91.7**)
KSDD2	(97.8, 89.9)	(97.2, 87.5)	(95.5, 87.9)	(95.4, 84.9)	(95.9, 88.7)	(**98.4**, **91.1**)
DAGM	(**99.7**, **98.9**)	(98.4, 96.3)	(98.0, 97.0)	(98.7, 97.2)	(98.4, 96.2)	(98.9, 97.9)
DTD-Synthetic	(99.6, 99.2)	(98.8, 98.2)	(99.6, 99.3)	(98.9, 98.7)	(96.2, 95.1)	(**99.7**, **99.5**)
RealIAD	(86.0, 78.9)	(87.0, 80.7)	(83.7, 78.3)	(85.6, 79.1)	(82.8, 75.8)	(**87.7**, **80.8**)
Average	(**93.8**, **90.5**)	(91.5, 87.7)	(90.8, 87.5)	(91.4, 88.3)	(89.8, 87.6)	(92.5, 90.3)

B.3 Comparison on CLIP.

CLIP is the backbone that language-based methods are built on, so it allows the closest comparison with them. On CLIP, our method reaches a pixel-level F1-max of 47.5 and an image-level F1-max of 87.6, on par with Bayes-PFL (47.4 and 87.5) and just below FB-CLIP (48.2 and 88.9), without using any text. Its pixel-level AUROC (94.4) is lower than theirs. Dropping text thus has little overhead on the same backbone and is faster than the language-based models.

B.4 The separating direction appears in every backbone

Figure 4 repeats the analysis of section 3 for all six backbones. The axes e_1 and e_2 and the learned directions come from MVTec-AD, and are applied on VisA with zero-shot settings. In every backbone, normal and defective patches fall on opposite sides of e_1. This also holds on VisA, which was not used to compute the axes. The two groups overlap most for CLIP, which is also the weakest backbone in table 5. In all six cases, the learned anomalous and normal directions lie at the two ends of the e_1 axis.

B.5 Zero-shot transfer to unseen datasets

Figure 5 applies the same axes and directions, learned on MVTec-AD, to seven unseen datasets for DINOv3, RADIO, and CLIP. With DINOv3 and RADIO, normal and defective patches separate on all seven datasets, with little overlap. However, with CLIP, the two groups overlap strongly on BTAD, KSDD2, DAGM, and DTD-Synthetic. These are also the datasets on which CLIP scores lowest in table 5, suggesting that CLIP is the weakest backbone for our method.

B.6 Structure of the learned directions.

Figure 6 shows the Gram matrix of the pixel branch, for every backbone and auxiliary data. Two patterns hold throughout. First, the anomalous and normal directions point away from each other: almost all similarities between the two sets are negative or close to zero; hence, we called them "opponent directions." Secondly, directions within the same set are positively aligned but not identical,

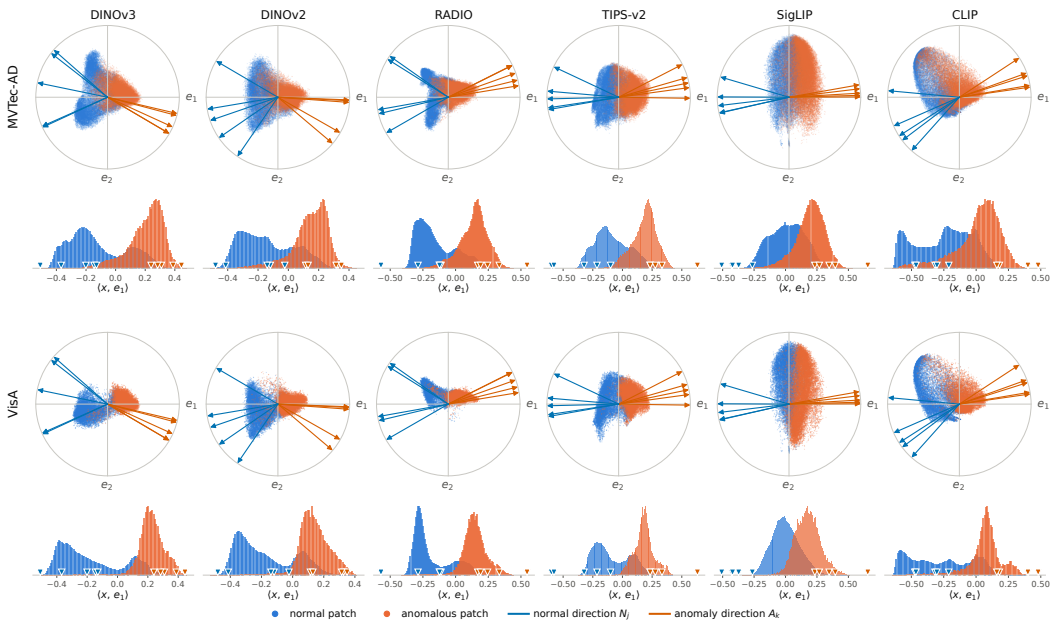


Figure 4: Six frozen backbones. The axes e_1, e_2, and the learned normal (blue) and anomalous (orange) directions are computed on MVTec-AD and applied to VisA in zero-shot settings.

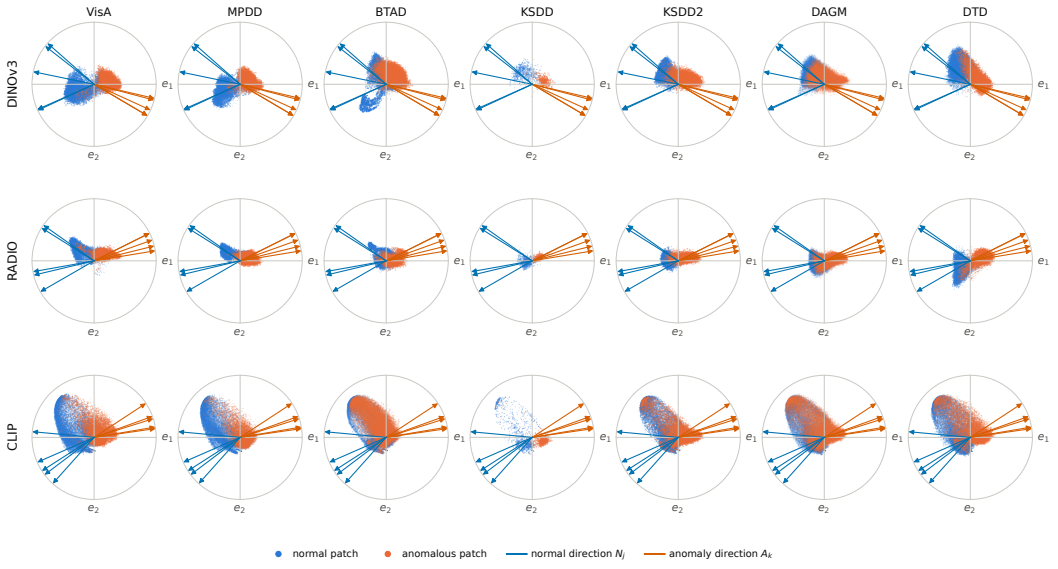


Figure 5: Transfer to seven unseen datasets for DINOv3, RADIO and CLIP. Axes and directions are computed on MVTec-AD only; no target labels are used.

with similarities ranging from about 0.1 to 0.8. In other words, all anomalous directions lean towards one side of the anomaly axis of section 3, and all normal directions lean towards the opposite side. However, its members still differ from each other, which is why several directions work better than one (section 6.5). The same structure appears whether the directions are learned on MVTec-AD or on VisA.

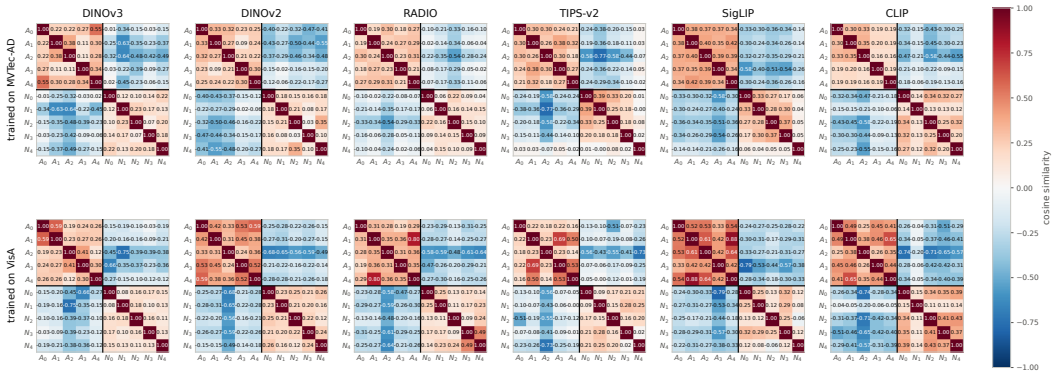


Figure 6: Gram matrix of directions for the pixel branch, layer 24, for six backbones trained on auxiliary data.

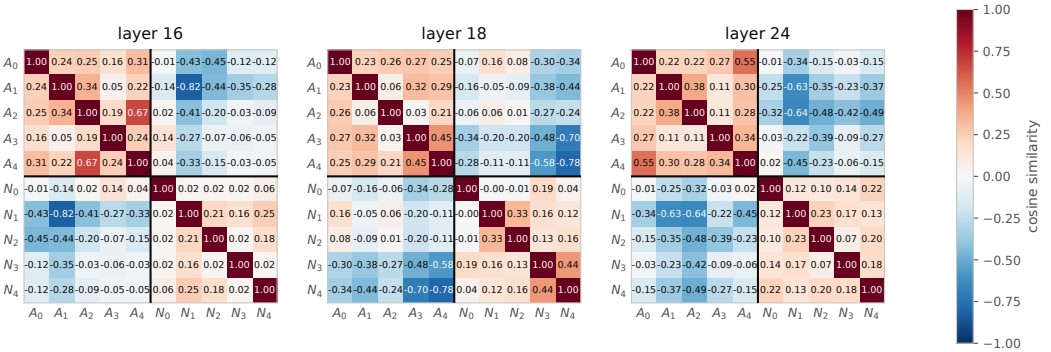


Figure 7: Gram matrix of pixel-branch directions at each block trained on MVTec-AD. Directions are positively aligned within each bank and mostly point away from the other bank.

C Analysis of Main Model

C.1 Geometric Interpretation

We project patches and learned directions of the main model (DINOv3, pixel branch) onto a two-dimensional frame built from labeled MVTec-AD patches at each block $j \in J$ (Figure 8). At every block, normal patches fall mostly on the negative side of e_1 and anomalous patches on the positive side, as shown in other backbones analysis (section B.2). This also holds for VisA (zero-shot), although e_1 comes from MVTec-AD alone.

Every anomalous direction has $\langle d_j^{\text{anom}}, e_1 \rangle > 0$ and every normal direction $\langle d_j^{\text{norm}}, e_1 \rangle < 0$ (triangles in Figure 8). These projections stay below about 0.55 in magnitude, so the model does not copy e_1. The gap between the two banks along e_1 grows with depth, from about 0.08 at block 16 to 0.37 at block 24.

The Gram matrices in Figure 7 show the same picture. Directions within a bank are similar (mean 0.24 to 0.28 for anomalous, 0.10 to 0.16 for normal) but not copies (at most 0.67), while the two banks point away from each other (mean -0.27 to -0.19). The normal bank is more spread out, which also shows in its wider fan of arrows in Figure 8. A likely reason is that normal appearance varies more across categories than the change a defect causes. Since the score takes a maximum over each bank (eq. (10)), a wider normal bank can cover more kinds of normal surface.

Why the score subtracts. Because the two banks lie on opposite sides of e_1, a small step $\delta > 0$ of a patch along e_1 changes $\langle x, d_j^{\text{anom}} \rangle$ by $\delta \langle e_1, d_j^{\text{anom}} \rangle > 0$ and $\langle x, d_j^{\text{norm}} \rangle$ by $\delta \langle e_1, d_j^{\text{norm}} \rangle < 0$, to first order. The first maximum in eq. (10) rises, the second falls, and the two terms add instead of

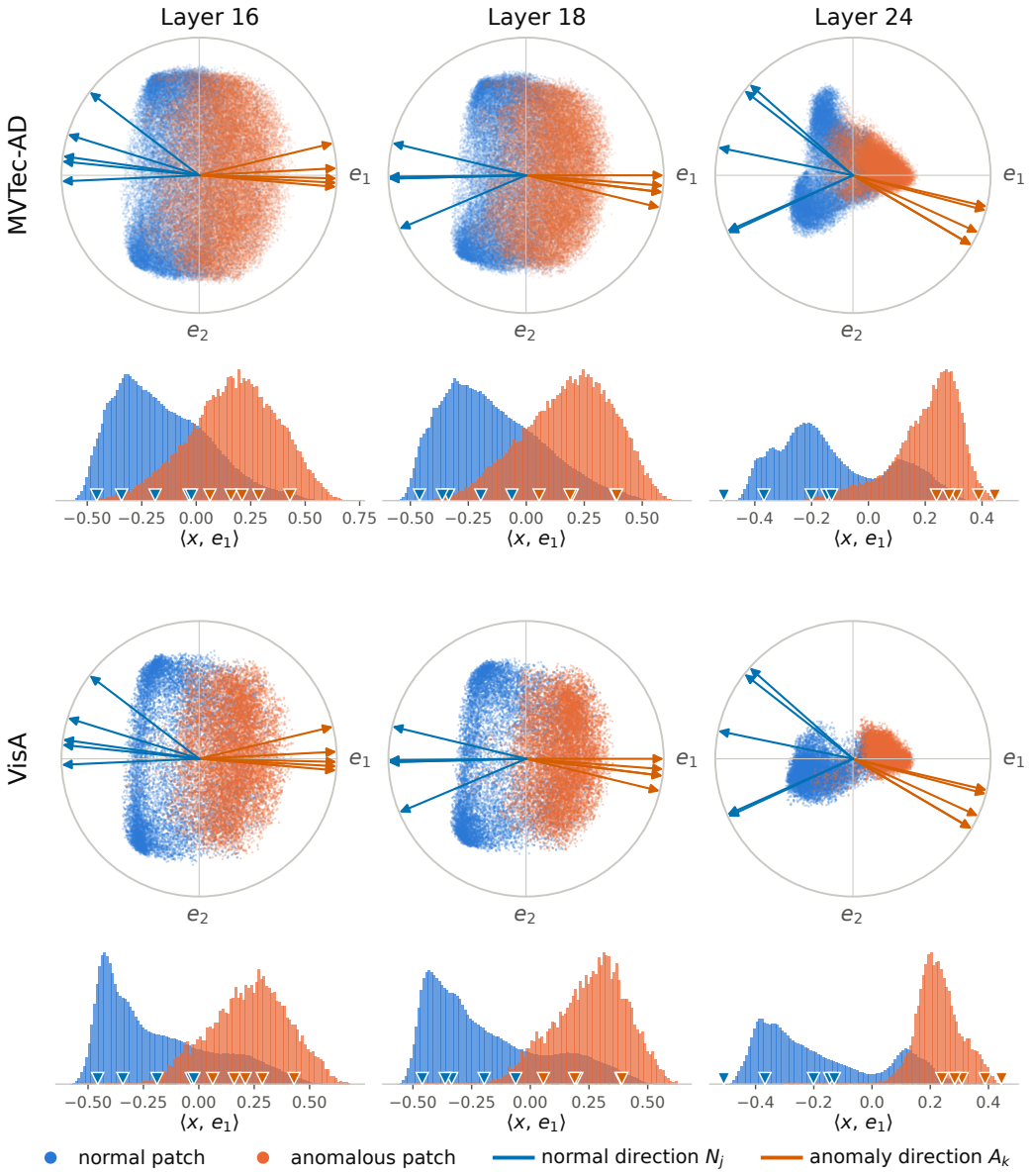


Figure 8: Patches and learned directions projected onto the frame (e_1, e_2), which is built from labeled MVTec-AD patches at each block. Both rows use the same frame and the same model trained on MVTec-AD. The frame (e_1, e_2) and learned directions are transferred zero-shot on the ViSA dataset.

cancelling. A shift that moves both banks by the same amount is removed by the subtraction. This fits the ablation, where dropping the anomalous bank lowers image AUROC by 7.7 points (table 3).

This view has one limitation: the arrows appear to point in identical directions. However, the frame has two axes, while the tokens live in $D = 1024$ dimensions, so Figure 8 shows only part of the geometry. The similarities in Figure 7, however, are computed in the full space.

C.2 COMPUTATIONAL COST

Comparison of the cost during inference of all models, all methods are measured on the same 24GB RTX 3090 GPU, for a single image with batch size 1. Latency is averaged over 10 images after 10 warm-up runs. Best value in each row in **bold**, and the second best is underline:

Metric	AdaCLIP	AA-CLIP	Bayes-PFL	FB-CLIP	AnomalyVFM	VisualAD	Ours
Frozen Parameters	428.76M	428.76M	429.35M	435.92M	**303.15M**	**303.15M**	**303.15M**
Learnable Parameters	10.67M	12.58M	27.37M	5.56M	35.47M	31.52M	**61.50K**
Total Parameters	439.42M	441.34M	456.72M	441.48M	338.63M	334.68M	**303.21M**
Latency (ms)	174.15	92.65	295.06	241.30	**61.41**	63.38	63.53
TFLOPs	2.68	1.04	1.82	1.31	1.63	0.67	**0.62**
GPU Memory (GB)	1.99	2.06	2.54	2.61	1.55	1.42	**1.25**

Almost all of our costs come from the frozen backbone. The learned directions add 61.5K parameters, and scoring a patch only requires comparing it with 30 directions per branch, which is negligible next to a forward pass of the backbone. As a result, our latency (63.5 ms) is close to that of the other two methods built on a ViT-L vision backbone, AnomalyVFM (61.4 ms) and VisualAD (63.4 ms), even though they train about 500 times more parameters.

The language-based methods are slower for two reasons. They carry a text encoder of about 125M parameters in addition to the image encoder, and some of them run it several times per image: Bayes-PFL, for example, samples ten sets of prompts at inference (Qu et al., 2025), which explains its latency of 295 ms. Our method needs no text encoder and runs the backbone once, so it also has the smallest total model (303.2M parameters) and the lowest GPU memory (1.25 GB).