LANGUAGE-ASSISTED FEATURE TRANSFORMATION FOR ANOMALY DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper introduces LAFT, a novel feature transformation method designed to incorporate user knowledge and preferences into anomaly detection using natural language. Accurately modeling the boundary of normality is crucial for distinguishing abnormal data, but this is often challenging due to limited data or the presence of nuisance attributes. While unsupervised methods that rely solely on data without user guidance are common, they may fail to detect anomalies of specific interest. To address this limitation, we propose **L**anguage-**A**ssisted **F**eature **T**ransformation (LAFT), which leverages the shared image-text embedding space of vision-language models to transform visual features according to user-defined requirements. Combined with anomaly detection methods, LAFT effectively aligns visual features with user preferences, allowing anomalies of interest to be detected. Extensive experiments on both toy and real-world datasets validate the effectiveness of our method.

1 INTRODUCTION

Anomaly detection (AD) is the task of distinguishing abnormal data that deviates from the norm. In most scenarios where anomaly detection is applied, normal data is relatively easy to obtain, while abnormal data is scarce or sometimes impossible to obtain in advance. Thus, typical anomaly detection methods rely on normal data provided by users to learn what constitutes normal. However, when the training data is biased or does not cover the diverse variations, modeling the boundary of normality becomes a significant challenge (Lee & Wang, 2020; Cohen et al., 2023). In practical applications, models may need to prioritize or disregard certain attributes of the data. For instance, when inspecting products in images, a user might focus solely on the product's shape, ignoring attributes like color or lighting conditions. Moreover, distinguishing anomalies becomes more difficult when attributes are entangled, as seen in the Waterbirds dataset, where the background and bird features are entangled (Sagawa et al., 2019).

To address this issue, various methods have been proposed that use data augmentation or generation techniques to improve the learning of decision boundaries (Zavrtanik et al., 2021; Li et al., 2021; Du et al., 2021). These approaches aim to produce more diverse samples, covering a broader range of the underlying data distribution than what is available. Additionally, some approaches focus on enabling models to learn task-specific feature representations (Chen et al., 2020a;b; Caron et al., 2020), applying them to anomaly detection to better capture feature-level normality (Hyun et al., 2023). However, a limitation of these methods is that they may struggle to generalize to completely unseen data or fail to align with the user's intent in defining normality.

In some scenarios, users may have prior knowledge or specific preferences about the data that they want to integrate into the anomaly detection process. Typically, this is achieved through indirect methods, such as manually applying random color augmentation to ignore certain object colors. Controlling the boundary of normality remains relatively unexplored, and existing approaches often require unrealistic conditions, such as access to anomaly samples or labels (Cohen et al., 2023). To overcome this limitation, we propose leveraging vision-language models to directly integrate user knowledge and preferences into the anomaly detection framework through natural language. By using language, users can more explicitly express their desired concepts, providing greater control over the definition of normality.

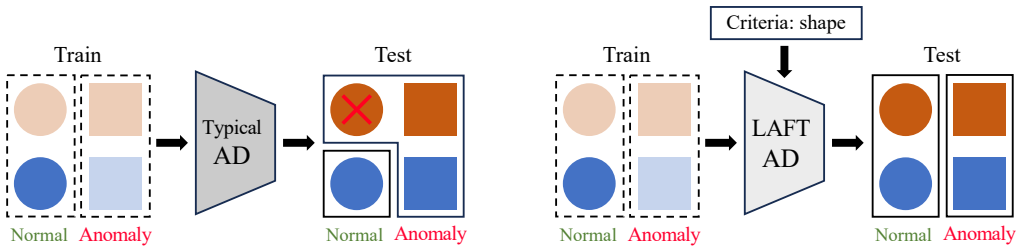

Figure 1: High-level motivation of our method: (**left**) typical image anomaly detection methods treat all test data that differs from the training data as anomalies, while (**right**) our method, LAFT AD, incorporates user preferences into the anomaly detection.

Recent studies have shown the effectiveness of training vision models with large amounts of unlabeled Internet data (Radford et al., 2021; Jia et al., 2021; Desai et al., 2023). By using image-text pairs from the web for pre-training, these models use natural language descriptions to improve the quality of image representations. Through large-scale training, they can correlate visual concepts in images with textual descriptions, aligning image and text features in a shared embedding space. Researchers have applied these models to industrial anomaly detection (Jeong et al., 2023; Cao et al., 2023b; Chen et al., 2023; Zhu & Pang, 2024) and general image out-of-distribution tasks using zero-shot text prompts (Ming et al., 2022; Miyai et al., 2023). A key benefit is the ability to incorporate human knowledge through text prompts, allowing zero-shot use without requiring additional training images. However, defining the complex normality of an image solely through natural language remains a challenge, and many methods face structural limitations in utilizing available training images. Therefore, we aim to develop a method where normality is primarily defined by image features, similar to other image anomaly detection approaches, with language serving only to refine the boundaries of normality. [TODO: We will add a description of Figure 1 in the final version.]

In this paper, we present Language-Assisted Feature Transformation (LAFT), a method that allows users to control the transformation of image features using natural language without requiring additional training. LAFT leverages the vision-language model CLIP (Radford et al., 2021), using its shared embedding space to link visual and textual features. This connection enables the transformation of visual features based solely on language inputs. We hypothesize that visual concept subspaces exist within this shared embedding space, and introduce the notion of a "concept axis" to represent these subspaces. By computing pairwise differences between textual features, we derive concept difference vectors that define the concept axes. Projecting visual features onto or orthogonal to these axes allows for selective emphasis or suppression of specific image attributes.

Our method offers a training-free approach by using language, whereas most feature transformation methods require extensive training on sufficient data. This property presents two key advantages: it is agnostic to downstream tasks and performs well even in settings where data is particularly scarce. By combining LAFT with a proper anomaly detection method, we can effectively apply LAFT to anomaly detection tasks. Our approach differs from most existing work using CLIP for anomaly detection in that it relies primarily on image features to define normality, with language playing a supporting role. By using language, users can provide their understanding of normality, allowing greater flexibility in incorporating domain knowledge. Furthermore, by defining the boundaries of normality using image features, the model is capable of accurately distinguishing between normal and abnormal images.

We summarize our contributions as follows:

1. We propose Language-Assisted Feature Transformation (LAFT), a novel method that uses natural language to transform image features to fit the given task requirements by leveraging the image-text aligned embedding space of CLIP.
2. We introduce LAFT AD, an anomaly detection method that combines LAFT with a k-nearest neighbor (kNN) classifier, enabling users to selectively focus on or ignore specific image attributes based on their guidance for semantic anomaly detection tasks.
3. We present WinCLIP+LAFT, an extension of WinCLIP that integrates LAFT to improve performance in industrial anomaly detection tasks.
4. We demonstrate the effectiveness of our method on Colored MNIST and extensively evaluate its performance on real-world datasets, including Waterbirds, CelebA, MVTec AD, and VisA.

2 Related Work

Image anomaly detection with vision-language model Since the advent of CLIP (Radford et al., 2021), numerous studies in image anomaly detection have attempted to exploit the generalization capability of this vision-language model. To align visual and textual features for effective out-of-distribution detection, Ming et al. (2022) proposed a scoring method called MCM, with Miyai et al. (2023) later presenting an improved version, GL-MCM. Addressing the limitations of zero-shot, Ming & Li (2023) tried to further enhance performance by using parameter-efficient fine-tuning in downstream tasks. Fort et al. (2021) aimed to improve the model's understanding of normality by providing the CLIP text encoder with candidate anomaly labels. In addition, Esmaeilpour et al. (2022) introduced a framework for training a label generator based on the CLIP image encoder to generate possible anomaly labels. For industrial anomaly detection, Jeong et al. (2023) introduced WinCLIP, a zero-/few-shot anomaly detection model that efficiently extracts and aggregates features at multiple levels, aligning them with textual information. Similarly, Chen et al. (2023) proposed APRIL-GAN, and Zhu & Pang (2024) developed InCTRL, both of which adapt CLIP image features using additional adapter layers to better align them for anomaly detection, although these approaches require additional pre-training of the adapter layers.

Adjusting the normality boundary Few studies have specifically addressed the challenge of adjusting the normality boundary in anomaly detection. Cohen et al. (2023) introduced Red PANDA, an anomaly detection method that disentangles relevant attributes in images while ignoring nuisance factors. However, achieving this disentangled feature representation requires labeled data for each nuisance attribute. Reiss et al. (2023) emphasized that overly expressive feature representations can ultimately degrade performance, highlighting a trade-off between sufficient representation and over-expressiveness in anomaly detection. Hendrycks et al. (2018) introduced outlier exposure, an approach that uses auxiliary data to help models generalize more effectively to unseen anomalies.

Extracting task-specific features Several strategies have been developed to enhance the adaptability and robustness of features extracted from backbone models. Some studies focus on fine-tuning pre-trained feature extraction backbones or generating task-specific features through feature transformations. Ruff et al. (2018) introduced a method that transforms normal data into a hypersphere representation for anomaly detection, and Reiss et al. (2021) proposed an early stopping strategy to prevent feature collapse. Chen et al. (2020a;b) utilized contrastive pre-training to facilitate feature agreement, and Caron et al. (2020) employed prototype vectors for contrastive training of similar features. Following this line of research, Hyun et al. (2023); Reiss & Hoshen (2023); Tack et al. (2020) extended contrastive learning approaches for anomaly detection. Zhao et al. (2023) suggested using the backbone of vision-language pre-trained diffusion models and training a text adaptor to extract task-specific features with text prompts for downstream tasks.

3 Preliminaries

In our scenario, a training set, represented as $\mathcal{D}_{\text{train}}$, consists of normal samples only, and a test set $\mathcal{D}_{\text{test}}$ consists of both normal and anomalous samples. For a two-stage anomaly detection model consisting of a feature extractor f and an anomaly classifier g, the feature extractor f maps the input image x to a feature $v = f(x)$, and the anomaly classifier g maps the feature v to an anomaly score $s = g(v)$. Then, the anomaly score s_i is used to determine the prediction of the anomaly label $\hat{y}_i$.

The attributes of an image x extracted by the feature extractor are denoted as $a = \{a^1, \cdots, a^m\}$, and the anomaly label is denoted as y. Each attribute a^j $(j = 1, \cdots, m)$ denotes any characteristics within the feature extracted from the image, such as the shape of the object, the color, or the background. The m attributes can be divided into relevant attributes $a^{\text{rel}} = \{a^j\}_{1 \le j \le n}$ and irrelevant (nuisance) attributes $a^{\text{irr}} = \{a^j\}_{n < j \le m}$ for desired anomaly detection tasks. For example, when detecting anomalies in the shape of objects, the shape is relevant, while the color is irrelevant. To properly detect anomalies, the prediction of the model should be invariant to the irrelevant attributes.

There are two ways to achieve this invariance:

1. Provide enough data that covers the possible values for each a^j, so that the classifier g can properly ignore irrelevant attributes a^{irr} in the feature v. This is the most desirable solution, and many data augmentation and generation methods have been proposed. However, it is often impossible to collect or hard to generate such data.

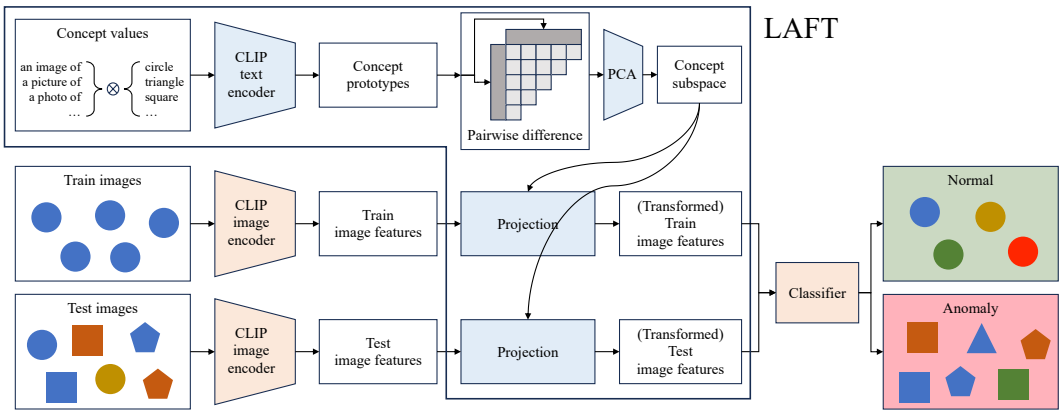

Figure 2: Overview of our method, LAFT, a transformation module, and LAFT AD, combining LAFT with a kNN classifier. Our approach uses CLIP's text and image encoders without any additional training. The key idea is to use text prompts containing concept values to construct a concept subspace for the target attribute. This process involves computing pairwise differences of concept prototypes and extracting robust concept axes via PCA. Once the concept subspaces are created, the shared embedding space can be used to transform image features suitable for anomaly detection.

2. Make the feature extractor f only extract the relevant attributes a^{rel} and do not include the irrelevant attributes a^{irr} in the feature v. Fine-tuning the feature extractor or adding a transformation T to the feature extractor is a common way to achieve this. But it is often hard to design such training procedures to achieve the invariance.

Our goal is to design a transformation T that transforms the feature v into a new feature $v' = T(v)$ that the anomaly classifier g can use to detect anomalies without being affected by the irrelevant attributes a^{irr} using only the user's natural language without any additional training data or labels.

This can be achieved by two approaches:

Guide Make a transformation T_{guide} that includes only the relevant attributes $a^j \in a^{\text{rel}}$. Some attributes in a^{rel} may be correlated, so the transformed feature may not include all relevant attributes.

Ignore Make a transformation T_{ignore} that excludes all irrelevant attributes $\forall a^j \in a^{\text{irr}}$. In many cases this is harder to achieve than the above approach, because the transformation should be able to remove all irrelevant attributes.

That is, we want our transformation T to represent the relevant attributes in a manner unaffected by the irrelevant attributes:

$$p(a^{n+1}, \cdots, a^m) = p(a^{n+1}, \cdots, a^m \mid T(v)). \tag{1}$$

We also want the transformed feature v' to be informative, containing enough information about relevant attributes. Here, $I(;)$ represents the mutual information between the two arguments:

$$I((a^1, \cdots, a^n);\, v) \sim I((a^1, \cdots, a^n);\, T(v)). \tag{2}$$

In practice, invariance can be measured by the accuracy of predicting anomaly label y from the transformed feature $T(v)$. But we can assess the informativeness by measuring the accuracy of predicting the relevant attribute utilized to define anomalies. Empirical evaluations of these measures for our datasets can be found in Experiments. With such a representation, we may later evaluate anomalies independently, devoid of any bias caused by the irrelevant attribute we aim to disregard.

CLIP (Radford et al., 2021) embeds the features in a unit sphere subspace in Euclidean space $\mathbb{R}^n$. An embedding vector of an image is correlated to the text embedding describing the image. This means that we can construct the transform with the CLIP text encoder. We assume that all relevant and irrelevant features can be encoded with the text description, so that natural language assists in the manipulation of the vector in the CLIP shared embedding space.

4 METHOD

Anomaly detection often faces data scarcity, especially for abnormal data, making it difficult for models to define the normality boundary. In this situation, users may have prior knowledge or preferences about what should be considered normal. Therefore, we aim to design:

- A method that can be used when the user has knowledge of normality and wants to control it. Typical anomaly detection methods consider all test data different from the training data as anomalies, but we want our method to be used only with a language.
- A method that effectively handles anomalies that are challenging to express solely in natural language. Other methods using vision-language models require normality to be expressed entirely in language prompts for guidance, which limits the ability to capture complex normality. We want our method to utilize image features to define normality.

Note that our method is not intended to be used in situations where the user has no knowledge or preferences, or to automatically identify relevant attributes.

Our method uses CLIP's text and image encoders without additional training. The core idea is to construct a ***concept subspace*** for target attributes using text prompts containing ***concept values***. After constructing the subspace, it transforms image features by projecting them onto the subspace, taking advantage of CLIP's shared embedding space. This involves computing pairwise differences between textual features containing ***concept prototypes*** to extract robust concept axes. In this section, we provide a detailed explanation of our method, as illustrated in Figure 2.

4.1 TEXT PROMPT

To guide the model in focusing on or ignoring specific attributes of an image, it is essential to provide it with appropriate textual prompts. Following Ming et al. (2022), we assume that the text contains ***concept prototypes*** representing the attributes. Thus, we provide the method with a list of prompts composed of templates and values, as commonly done in CLIP-based methods (Radford et al., 2021; Ming et al., 2022; Jeong et al., 2023). The key difference in our approach is that we use the actual values of the desired attribute (e.g., "circle," "square") rather than the attribute name itself (e.g., "shape"). For instance, to capture the concept of hair color, we can construct the prompt as:

- "a photo of a person with *brown hair*"
- "a potrait of a man with *black hair*"
- "an image of a *blond* child"

By using the actual values of the desired attribute in the prompts, we aim for the method to capture the difference between the concept prototypes of the attribute. Providing values for this attribute that are not present in the training set, but are likely to appear during testing, helps construct a more comprehensive subspace for that concept. As with other language-based methods, multiple types of templates can be provided to mitigate the bias introduced by any single template. We examine the effect of various sets of concept values in the Ablation Study, with the prompts used in our experiments detailed in the Appendix.

4.2 FIND CONCEPT SUBSPACE

Mikolov (2013) showed that simple arithmetic operations between text embeddings can capture meaningful relationships (e.g., vec(`biggest`) − vec(`big`) ≈ vec(`smallest`) − vec(`small`)). This finding showed that text embeddings not only represent texts in a vector space, but also encode the underlying relationships between them. Moreover, they observed that high-dimensional vectors, when trained on large datasets, are capable of capturing subtle semantic relationships. Similarly, CLIP's text embeddings support arithmetic operations to compute differences between concept prototypes, allowing for the comparison of these concepts in the embedding space.

Building on this approach, the method constructs a subspace of the concept within CLIP's embedding space. Specifically, it identifies the axes of this subspace that capture the variance between concept prototypes, as represented by difference between the prompts. For prompts t_i and t_j, where $1 \leq i < j \leq n$, we compute the pairwise differences of the text features:

$$\Delta v_{ij} := E_{\text{text}}(t_i) - E_{\text{text}}(t_j) \tag{3}$$

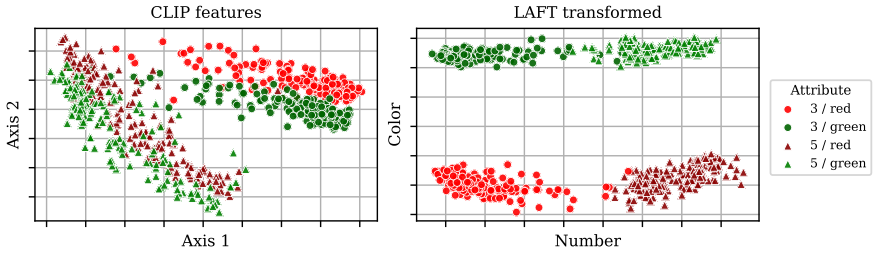

Figure 3: Projection of image features from CLIP's image encoder (**left**) and transformed image features using LAFT (**right**). Without guidance, the image features may not align with the intended attributes. After applying LAFT, the features become more aligned with the desired attributes.

where n represents the number of prompts, and E_{text} denotes CLIP's text encoder. But directly using these vectors is not preferable because the prompts may contain template noise Zhou et al. (2022). To address this, we apply PCA to extract the principal axes from these vectors:

$$\{c_k\}_{1\leq k\leq d} := \text{PCA}(\{\Delta v_{ij}\}_{1\leq i<j\leq n}, d) \tag{4}$$

where d is the number of components, and $\{c_k\}$ represents the d principal axes, collectively referred to as the ***concept axes***. Throughout this paper, we typically select d between 8 and 32 when guiding an attribute, and between 32 and 384 when ignoring an attribute. As discussed in the Preliminaries, ignoring attributes is generally more challenging than guiding them, so a larger number of components is used. For the impact of d, please refer to the Ablation Study.

4.3 Feature Transformation with Projection

For each image feature $v_i = f(x_i)$ encoded by CLIP's image encoder, we project the features onto the concept subspace:

$$v_i' = T_{\text{guide}}(v_i) := \sum_{k=1}^{d} \frac{\langle v_i, c_k \rangle}{\langle c_k, c_k \rangle} c_k, \tag{5}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. This projection retains only the relevant attributes of the image feature, as irrelevant attributes are nearly orthogonal to the concept axes. Conversely, we can remove the irrelevant attributes using orthogonal projection. Let $\bar{c}_k$ represent the concept axes associated with the irrelevant attributes. Then we can project orthogonally onto the concept subspace as follows:

$$\bar{v}_i' = T_{\text{ignore}}(v_i) := v_i - \sum_{k=1}^{d} \frac{\langle v_i, \bar{c}_k \rangle}{\langle \bar{c}_k, \bar{c}_k \rangle} \bar{c}_k, \tag{6}$$

which manually cancels out the vectors of irrelevant attributes. This completes the description of the feature transformation method, LAFT.

4.4 Anomaly Scoring

Many anomaly detection methods employ k-nearest-neighbors (kNN) for anomaly scoring (Cohen & Hoshen, 2020; Roth et al., 2022). This approach is effective because normal data tends to be densely concentrated, while anomalous data is typically sparsely distributed in the feature space. In our method, LAFT AD, we also use kNN to estimate the density of normal data around each test sample, assuming that the features have been processed by LAFT for *semantic anomaly detection*.

We start by extracting features for each normal sample: $v_i' = T(f(x_i)),\ \forall x_i \in \mathcal{D}_{\text{train}}$. Next, for each test sample, we infer its feature: $v_j' = T(f(x_j)),\ \forall x_j \in \mathcal{D}_{\text{test}}$. Finally, we score each test sample based on its kNN distance from the normal data:

$$s_j = g(v_j') := \frac{1}{k} \sum_{v_i' \in N_k(v_j')} S_{\cos}(v_j', v_i'), \tag{7}$$

where $N_k(v_i')$ denotes the k nearest features to v_j' in the normal data, and $S_{\cos}(\cdot, \cdot)$ represents cosine similarity. We use $k = 30$ for kNN throughout the paper without optimization.

However, this method may not be suitable for industrial anomaly detection tasks, where anomalies are often small and subtle. To address this, we extend WinCLIP (Jeong et al., 2023) to incorporate LAFT for anomaly scoring, which we refer to as WinCLIP+LAFT for *industrial anomaly detection*. A detailed explanation of this extension is provided in Experiment.

5 EXPERIMENTS

Datasets To validate our approach, we used the colored version of MNIST (LeCun et al., 2010), Waterbirds (Sagawa et al., 2019), and CelebA (Liu et al., 2015) datasets for semantic anomaly detection (SAD). We defined normal and anomalous values for each dataset attribute and divided the training split into 2^m subsets. For example, in the Colored MNIST dataset, we designated digits 0-4 as normal and 5-9 as anomalous, with the color red as normal and green and blue as anomalous. We then used one subset as the training set, considering it normal across all m attributes (e.g., digits 0-4 and the color red). This is similar to the setup commonly used in many studies for a single attribute (primarily based on class labels) (Ruff et al., 2020; Tack et al., 2020; Esmaeilpour et al., 2022; Cohen et al., 2023; Reiss & Hoshen, 2023; Cao et al., 2023a; Zhu & Pang, 2024). To further demonstrate the practicality of our method, we also used the MVTec AD (Bergmann et al., 2019) and VisA (Zou et al., 2022) datasets for industrial anomaly detection (IAD). For more details, see Experimental Setup.

Baselines For semantic anomaly datasets, we do not compare with typical image-only AD methods for two reasons: (1) they require attribute-specific processing (e.g., color augmentation) to incorporate user prior knowledge, which limits generalizability to other contexts, and (2) without guidance, these methods detect images that differ from the training set, resulting in high false positive rates in our settings. Instead, to simulate image-only AD methods, we compare with simple ***k*NN** and **LinearProbe** with additional training data, directly using image features from CLIP.

kNN computes the distance between the test image features and the training image features for anomaly scoring. As discussed in Method, many well-known anomaly detection (AD) methods rely on kNN-based anomaly scoring, making it an important baseline. kNN using the same training subset as the other methods can be viewed as a no-guidance version of LAFT AD, allowing us to evaluate the effectiveness of LAFT. And kNN using additional normal training subset depending on the target attribute represents a image-only method with attribute-specific image processing. Since we can't simply apply image augmentation for each dataset and attribute (e.g. how to augment the background in Waterbirds), we assume that the additional data is very well *augmented* images for the target attribute. To evaluate CLIP image encoder's performance, we provide full training data including normal and anomalous images for LinearProbe to train a linear classifier to predict the class of the test image (Radford et al., 2021).

We also evaluate CLIP-based zero-shot and few-shot AD methods. For zero-shot AD, we use Maximum Concept Matching (**MCM**; Ming et al., 2022), which requires only prompts for normal images to perform anomaly scoring, and Zero-shot outlier exposure (**ZOE**; Fort et al., 2021), which uses prompts for normal images and candidate prompts for anomalous images. And **CLIPN** (Wang et al., 2023) is a zero-shot method that uses pre-trained "no" prompts and "no" text encoder to make prompts for anomalies. We also consider **WinCLIP** (Jeong et al., 2023), which supports both zero-/few-shot AD. The zero-shot version of WinCLIP is similar to ZOE in terms of anomaly scoring, and the few-shot version (WinCLIP+) requires a few normal images. Lastly, we consider **InCTRL** (Zhu & Pang, 2024), a few-shot AD method similar to WinCLIP+. And for industrial anomaly datasets, we also compare with **PatchCore** (Roth et al., 2022). [TODO: For more details, see Baselines.]

Prompts We use the actual class names from the dataset for concept values, if available, and add other candidate labels to simulate unseen classes. For example, for the number attribute in the Colored MNIST dataset, we use '0' to '20' and 'zero' to 'twenty' as number attributes, even though the dataset only includes 0 to 9. We referenced the prompts provided by CLIP (Radford et al., 2021). [TODO: We will add more details on how to construct prompts in the final version.] For more details, please refer to the prompts section in Appendix.

Table 1: Anomaly detection performance on Colored MNIST and Waterbirds datasets. Standard deviations are computed over five different seeds, with results for deterministic cases omitted.

		Colored MNIST: Number			Waterbirds: Bird		
Guidance	Method	AUROC ↑	AUPRC ↑	FPR95 ↓	AUROC ↑	AUPRC ↑	FPR95 ↓
Baseline							
Normals	kNN	0.897 ± 0.004	0.895 ± 0.004	0.440 ± 0.003	0.834	0.918	0.447
+ Unseen normals	kNN	0.974 ± 0.001	0.964 ± 0.001	0.094 ± 0.001	0.842	0.921	0.422
+ Anomalies	LinearProbe	0.998 ± 0.000	0.998 ± 0.000	0.008 ± 0.000	0.954	0.982	0.167
Guide							
Language	MCM	0.693	0.578	0.537	0.899	0.958	0.357
	ZOE	0.914	0.924	0.440	0.924	0.972	0.338
	CLIPN-C	0.645 ± 0.043	0.619 ± 0.068	0.776 ± 0.026	0.773 ± 0.005	0.897 ± 0.000	1.000 ± 0.000
	CLIPN-A	0.724 ± 0.073	0.684 ± 0.094	0.698 ± 0.044	0.841 ± 0.001	0.930 ± 0.002	0.540 ± 0.000
	WinCLIP	0.912	0.923	0.460	0.923	0.971	0.335
Image + Language	WinCLIP+	0.936 ± 0.002	0.943 ± 0.001	0.353 ± 0.012	0.923 ± 0.001	0.971 ± 0.000	0.337 ± 0.007
	InCTRL	0.578 ± 0.001	0.618 ± 0.002	0.935 ± 0.001	0.845 ± 0.005	0.934 ± 0.002	0.638 ± 0.013
	LAFT AD (Ours)	**0.985** ± 0.000	**0.984** ± 0.000	**0.069** ± 0.001	**0.956**	**0.984**	**0.206**
Ignore							
Image + Language	LAFT AD (Ours)	0.967 ± 0.001	0.962 ± 0.003	0.144 ± 0.004	0.847	0.921	0.399

Table 2: Anomaly detection performance on CelebA dataset. Standard deviations are computed over five different seeds, with results for deterministic cases omitted.

		Hair color			Eyeglasses		
Guidance	Method	AUROC ↑	AUPRC ↑	FPR95 ↓	AUROC ↑	AUPRC ↑	FPR95 ↓
Baseline							
Normals	kNN	0.835	0.966	0.608	0.824	0.204	0.486
+ Unseen normals	kNN	0.837	0.967	0.608	0.847	0.205	0.428
+ Anomalies	LinearProbe	0.979	0.997	0.119	0.995	0.973	0.003
Guide							
Language	MCM	0.880	0.979	0.566	0.081	0.034	1.000
	ZOE	0.934	0.989	0.388	0.929	0.624	0.342
	CLIPN-C	0.773 ± 0.019	0.953 ± 0.005	0.705 ± 0.000	0.012 ± 0.000	0.033 ± 0.000	1.000 ± 0.000
	CLIPN-A	0.890 ± 0.002	0.980 ± 0.001	0.523 ± 0.000	0.011 ± 0.000	0.033 ± 0.000	1.000 ± 0.000
	WinCLIP	0.934	0.989	0.394	0.934	0.652	0.322
Image + Language	WinCLIP+	0.934 ± 0.001	0.989 ± 0.000	0.391 ± 0.005	0.941 ± 0.009	0.628 ± 0.034	0.286 ± 0.042
	InCTRL	0.876 ± 0.006	0.976 ± 0.003	0.691 ± 0.021	0.890 ± 0.030	0.308 ± 0.090	0.303 ± 0.035
	LAFT AD (Ours)	**0.950**	**0.992**	**0.298**	**0.981**	**0.807**	**0.059**

5.1 Semantic Anomaly Detection

We used the colored version of the MNIST dataset (LeCun et al., 2010), similar to Arjovsky et al. (2019), to demonstrate our concept in the simplest way. We created a dataset that divides each digit of the MNIST and colors each split with red, green, and blue. In this way, the image of a colored MNIST consists of two attributes: number and color. We mark the numbers 0 to 4 as normal and the numbers 5 to 9 as abnormal. In addition, we label red as normal and green and blue as abnormal colors. In this setting, the training set consists of 0 to 4 and red images. Then, we use five different seeds to split the training set for coloring each digit. Figure 3 shows a brief overview of our desired transformation using concept axes. If we choose an axis (number or color) to project the image features, we can simply use kNN to detect only the desired anomalies.

Table 1 presents the main results on Colored MNIST. The table is divided into three groups: *no guidance*, *guide* (Lang., Img. + Lang.), and *ignore*, as discussed in Preliminaries. The *no guidance* group refers to methods that do not provide guidance and serves as a reference for many typical anomaly detection methods. The *guide* group consists of methods that can be instructed to focus on a target attribute, where models are given prompts related to the attribute corresponding to the label (e.g., number prompts for number anomalies). Specifically, methods in the *guide* (Lang.) group rely solely on language, while those in the *guide* (Img. + Lang.) group use both image and language to define normality. However, except for our method, guidance in these methods is only used to calculate image-text similarity and is not applied to image-image similarity. The *ignore* group represents a method that disregard attributes other than the target, where models are provided prompts of irrelevant attributes (e.g., color prompts for number anomalies). Ignoring irrelevant attributes is a unique feature of our method, but it is generally a more challenging task.

As shown in the table, guidable methods generally outperform non-guidable methods, with our method achieving the best overall performance. ZOE's performance is lower than ours because it

Table 3: Anomaly detection AUROC on MVTec AD and VisA datasets in few-shot settings. We use five different sets of reference samples from the training set for each method.

	# Shots				
Method	K = 0	K = 1	K = 2	K = 4	K = 8
MVTec AD					
PatchCore*	×	0.834 ± 0.030	0.863 ± 0.033	0.888 ± 0.026	-
InCTRL	×	0.913 ± 0.027	0.932 ± 0.018	0.936 ± 0.016	0.938 ± 0.013
WinCLIP/+	0.904	0.933 ± 0.018	0.947 ± 0.008	0.950 ± 0.005	0.953 ± 0.004
+ LAFT General (Ours)	**0.932**	**0.946** ± 0.021	**0.958** ± 0.009	**0.961** ± 0.003	**0.963** ± 0.003
+ LAFT Category (Ours)	0.929	0.944 ± 0.012	0.957 ± 0.006	0.960 ± 0.003	0.962 ± 0.003
VisA					
PatchCore*	×	0.799 ± 0.029	0.816 ± 0.040	0.853 ± 0.021	-
InCTRL	×	0.850 ± 0.043	**0.867** ± 0.027	**0.884** ± 0.020	**0.894** ± 0.017
WinCLIP/+	0.756	0.826 ± 0.021	0.843 ± 0.016	0.854 ± 0.016	0.865 ± 0.011
+ LAFT General (Ours)	**0.806**	0.852 ± 0.012	0.861 ± 0.010	0.867 ± 0.010	0.873 ± 0.009
+ LAFT Category (Ours)	0.812	**0.858** ± 0.011	**0.867** ± 0.010	0.876 ± 0.011	0.885 ± 0.010

is provided with inaccurate prompts for anomalous images (e.g., '13'). This problem, highlighted in Ming et al. (2022), shows that methods relying on image-text similarity in CLIP are highly sensitive to inaccurate prompts. WinCLIP performs similarly to ZOE because multi-scale features do not benefit semantic anomaly detection. While WinCLIP+ performs better than WinCLIP by using reference images, its performance is still below ours. InCTRL performs poorly likely because its feature adapter was trained on the MVTec AD dataset, which differs significantly from the Colored MNIST dataset, aligning with the findings in Zhu & Pang (2024). In contrast, our method uses prompts only to transform the image features, while normality is determined by the images themselves, leading to superior performance. In addition, we observe that when our method is used to guide one attribute, other attributes are ignored, as discussed further in Additional Experiments.

The Waterbirds dataset (Sagawa et al., 2019) is widely used in studies of spurious correlations and disentangling representations. It consists of two primary attributes: bird type (waterbird / landbird) and background (water / land). Naturally, the training set has a very strong correlation between birds and backgrounds, whereas the test set has an equal ratio of birds to backgrounds. We specify waterbirds and water backgrounds as the normal training set. Table 1 summarizes the results for this dataset. The trends observed in the Colored MNIST experiment are largely consistent, demonstrating the applicability of our method to real-world datasets. The key difference is that ignoring one attribute (background) does not directly improve performance on the other attribute (bird).

To verify that our method works in multi-attribute datasets, we use the CelebA dataset (Liu et al., 2015), which contains over 200K celebrity images with 40 attribute labels. For the normal training set, we select two attributes: Hair color and Eyeglasses. The results are displayed in Table 2. The trends are consistent with the previous experiments, demonstrating the effectiveness of our method.

5.2 Industrial Anomaly Detection

To demonstrate the practical applicability of our method beyond semantic anomaly detection, we evaluated its performance on the widely used MVTec AD (Bergmann et al., 2019) and VisA (Zou et al., 2022) datasets in few-shot settings. However, anomalies in industrial anomaly detection datasets often consist of small defects that are difficult to distinguish using only image-level representations. Instead of using LAFT AD, which is designed for semantic anomaly detection tasks, we propose WinCLIP+LAFT, a model that applies LAFT to WinCLIP to extract multi-scale features using CLIP. We apply LAFT to WinCLIP's window, image, and text embeddings, all of which reside in CLIP's shared embedding space, allowing seamless integration.

Typically, some zero-shot or few-shot methods based on CLIP rely on training additional adapter layers to transform CLIP's image features for anomaly detection tasks. For example, InCTRL pretrains feature adapters on specific datasets (such as MVTec AD) to effectively compute image-image similarity before applying them to different datasets. In contrast, our method uses prompts to transform image features while preserving CLIP's core features, allowing us to extract features suitable for anomaly detection tasks without the need for additional training of the adapter layer.

For the proof of concept, we used prompts similar to those used in WinCLIP for LAFT (LAFT General) and category-specific prompts (LAFT Category). For LAFT General, we constructed prompts

Table 4: Anomaly detection performance on the Colored MNIST and Waterbirds datasets with various prompts. Standard deviations are computed over five different seeds.

	Concept values			Colored MNIST: Number			Waterbirds: Bird		
Prompt	Seen	Unseen	Aux.	AUROC ↑	AUPRC ↑	FPR95 ↓	AUROC ↑	AUPRC ↑	FPR95 ↓
Guide									
Only normals	○	×	×	0.962 ± 0.002	0.957 ± 0.002	0.166 ± 0.002	0.943	0.978	0.226
Partial anomalies	○	△	×	0.984 ± 0.000	0.983 ± 0.000	0.082 ± 0.001	0.955	0.983	**0.187**
Exact anomalies	○	○	×	**0.985** ± 0.000	**0.984** ± 0.000	0.076 ± 0.001	**0.959**	**0.986**	0.196
All candidates	○	○	○	**0.985** ± 0.000	**0.984** ± 0.000	**0.069** ± 0.001	0.956	0.984	0.206
Ignore									
Only seen normals	○	×	×	0.945 ± 0.002	0.941 ± 0.004	0.250 ± 0.005	0.848	**0.924**	0.406
Partial unseen normals	○	△	×	0.959 ± 0.002	0.951 ± 0.003	0.167 ± 0.004	**0.851**	**0.924**	0.381
Exact normals	○	○	×	**0.970** ± 0.001	**0.965** ± 0.003	**0.125** ± 0.003	0.850	0.922	**0.373**
All candidates	○	○	○	0.967 ± 0.001	0.962 ± 0.003	0.144 ± 0.004	0.847	0.921	0.399

using only state words and category names, without providing additional knowledge based on the category of the inspection image as in WinCLIP. However, in order to identify a more precise concept subspace, we use some more text templates and general state words such as 'malformed {}'. For LAFT Category, we used prompts that include category-specific knowledge (e.g. anomaly class names), such as 'bottle with large breakage' for the bottle category. See Prompts for more details.

The results are presented in Table 3, which summarizes the average performance across all categories. For PatchCore, we used the results provided by WinCLIP (Jeong et al., 2023). As shown in the results, WinCLIP+LAFT consistently outperforms WinCLIP in both zero-shot and few-shot scenarios. The results show that LAFT improves WinCLIP's ability to compute image-text similarity, and it also improves the computation of image-image similarity between the query and reference images. [TODO: We will add more detailed analysis in the final version.] ~~Moreover, WinCLIP+LAFT outperforms InCTRL even when using the authors' pre-trained weights, which train adapter layers directly on MVTec AD.~~ See Full Results for detailed results on each category.

5.3 Ablation Study on Prompt Quality

An important consideration when using LAFT is how to provide the user's prior knowledge. In anomaly detection, we generally have a good understanding of the current training data, but the unseen test data remains unknown. Therefore, we investigated how the performance of LAFT changes depending on the quality of the concept values provided, as shown in Table 4. In the table, *Seen* refers to the concept values for the current training data, *Unseen* refers to the concept values for the unseen test data, and *Aux.* denotes concept values that are not present in the dataset.

For example, in Colored MNIST, if the guiding attribute is the number, *Seen* represents the values 0-4, *Unseen* corresponds to 5-9, and *Aux.* refers to values like 10-20. Similarly, if the ignored attribute is color, *Seen* includes red, *Unseen* covers green and blue, and *Aux.* includes colors such as yellow and purple. The symbol ○ indicates that all concept values are used, while △ indicates that only half of the concept values are utilized.

The results show that the performance of LAFT is robust to the quality of the concept values when at least partial concept values are provided. Furthermore, the performance is not significantly affected when the completely not included concept values are provided. Also, providing concept values that are not included at all does not significantly affect the performance. These characteristics show that LAFT can be effectively used in anomaly detection where only limited information is known.

6 Conclusion

In this paper, we introduce Language-Assisted Feature Transformation (LAFT), a novel feature transformation method designed to integrate user knowledge and preferences into the anomaly detection framework via natural language, without the need for additional data or training. By utilizing the shared embedding space of the vision-language model CLIP, LAFT can align visual features with user-provided text prompts to guide or ignore specific attributes in the image. We also presented LAFT AD, an anomaly detection method that integrates LAFT with a k-nearest neighbor classifier, and WinCLIP+LAFT, an extension of WinCLIP that incorporates LAFT for industrial anomaly detection. This combination allows users to adjust the normality boundary of the model by providing text prompts to detect desired anomalies. Through experiments on synthetic and real-world datasets, we demonstrate the effectiveness of our proposed method.

REFERENCES

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

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)*, 2019.

Tri Cao, Jiawen Zhu, and Guansong Pang. Anomaly detection under distribution shift. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023a.

Yunkang Cao, Xiaohao Xu, Chen Sun, Yuqi Cheng, Zongwei Du, Liang Gao, and Weiming Shen. Segment any anomaly without training via hybrid prompt regularization. *arXiv preprint arXiv:2305.10724*, 2023b.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2020a.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020b.

Xuhai Chen, Yue Han, and Jiangning Zhang. A zero-/few-shot anomaly classification and segmentation method for cvpr 2023 vand workshop challenge tracks 1&2: 1st place on zero-shot ad and 4th place on few-shot ad. *arXiv preprint arXiv:2305.17382*, 2023.

Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *arXiv preprint arXiv:2005.02357*, 2020.

Niv Cohen, Jonathan Kahana, and Yedid Hoshen. Red panda: Disambiguating image anomaly detection by removing nuisance factors. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Ramakrishna Vedantam. Hyperbolic Image-Text Representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2023.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Xuefeng Du, Zhaoning Wang, Mu Cai, and Yixuan Li. Vos: Learning what you don't know by virtual outlier synthesis. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Sepideh Esmaeilpour, Bing Liu, Eric Robertson, and Lei Shu. Zero-shot out-of-distribution detection based on the pre-trained model clip. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Alex Fang, Albin Madappally Jose, Amit Jain, Ludwig Schmidt, Alexander Toshev, and Vaishaal Shankar. Data filtering networks. *arXiv preprint arXiv:2309.17425*, 2023.

Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. Eva-02: A visual representation for neon genesis. *Image and Vision Computing*, 2024.

Stanislav Fort, Jie Ren, and Balaji Lakshminarayanan. Exploring the limits of out-of-distribution detection. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. Datacomp: In search of the next generation of multimodal datasets. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606*, 2018.

Jeeho Hyun, Sangyun Kim, Giyoung Jeon, Seung Hwan Kim, Kyunghoon Bae, and Byung Jun Kang. Reconpatch: Contrastive patch representation learning for industrial anomaly detection. *arXiv preprint arXiv:2305.16713*, 2023.

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. URL https://doi.org/10.5281/zenodo.5143773.

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

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2010.

Wei-Yu Lee and Yu-Chiang Frank Wang. Learning disentangled feature representations for anomaly detection. In *2020 IEEE International Conference on Image Processing (ICIP)*, 2020.

Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Xiaofan Li, Zhizhong Zhang, Xin Tan, Chengwei Chen, Yanyun Qu, Yuan Xie, and Lizhuang Ma. Promptad: Learning prompts with only normal samples for few-shot anomaly detection. *arXiv preprint arXiv:2404.05231*, 2024.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2015.

Tomas Mikolov. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Yifei Ming and Yixuan Li. How does fine-tuning impact out-of-distribution detection for vision-language models? *arXiv preprint arXiv:2306.06048*, 2023.

Yifei Ming, Ziyang Cai, Jiuxiang Gu, Yiyou Sun, Wei Li, and Yixuan Li. Delving into out-of-distribution detection with vision-language representations. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Atsuyuki Miyai, Qing Yu, Go Irie, and Kiyoharu Aizawa. Zero-shot in-distribution detection in multi-object settings using vision-language foundation models. *arXiv preprint arXiv:2304.04521*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.

Tal Reiss and Yedid Hoshen. Mean-shifted contrastive loss for anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Tal Reiss, Niv Cohen, Liron Bergman, and Yedid Hoshen. Panda: Adapting pretrained features for anomaly detection and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2806–2814, 2021.

Tal Reiss, Niv Cohen, and Yedid Hoshen. No free lunch: The hazards of over-expressive representations in anomaly detection. *arXiv preprint arXiv:2306.07284*, 2023.

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)*, 2022.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. Eva-clip: Improved training techniques for clip at scale. *arXiv preprint arXiv:2303.15389*, 2023.

Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Hualiang Wang, Yi Li, Huifeng Yao, and Xiaomeng Li. Clipn for zero-shot ood detection: Teaching clip to say no. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

Vitjan Zavrtanik, Matej Kristan, and Danijel Skocaj. Dræm–a discriminatively trained reconstruction embedding for surface anomaly detection. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

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

Wenliang Zhao, Yongming Rao, Zuyan Liu, Benlin Liu, Jie Zhou, and Jiwen Lu. Unleashing text-to-image diffusion models for visual perception. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision (IJCV)*, 2022.

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

Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. *arXiv preprint arXiv:2207.14315*, 2022.

A Limitations and Discussion

Table 5: Anomaly detection performance on the CelebA dataset. We transform the image and text features using LAFT to ignore the Gender attribute. Underlined numbers indicate the performance of the suppressed attribute, and bold numbers indicate the performance of the non-suppressed attribute.

Method	Hair color			Gender		
	AUROC ↑	AUPRC ↑	FPR95 ↓	AUROC ↑	AUPRC ↑	FPR95 ↓
MCM	0.880	0.979	0.567	0.969	0.980	0.137
+ LAFT	**0.938**	**0.990**	**0.354**	<u>0.324</u>	<u>0.500</u>	<u>0.984</u>
ZOE	0.934	0.989	0.388	0.994	0.996	0.025
+ LAFT	**0.951**	**0.992**	**0.306**	<u>0.500</u>	<u>0.612</u>	<u>0.947</u>

Ignore attributes using LAFT Unlike in a simple Colored MNIST dataset, we observe that ignoring one attribute using LAFT does not directly improve the performance of the other attribute in real-world datasets. However, as seen in Appendix C, the LAFT actually suppresses the attribute to be ignored. As mentioned in the Preliminaries, it is hard to remove all attribute-related information in the embedding space using only text prompts. Alternatively, guiding the attribute is relatively easy, because LAFT only needs to capture the primary information about the attribute.

Using LAFT with other methods Our proposed method, LAFT, can be used not only for anomaly detection, but also as a feature transformation module in other tasks or methods. Basically, we expect that it can be applied to any vision model that requires a feature extractor. For simple example, we apply the LAFT method to MCM and ZOE. The results in Table 5 show that we can suppress the Gender attribute. Applying LAFT to other downstream tasks would be a future work.

B Experimental Details

Backbones For semantic anomaly detection datasets (Colored MNIST, Waterbirds, CelebA), we use the CLIP ViT-B/16 (Radford et al., 2021) with the pre-trained checkpoint from Fang et al. (2023). And since InCTRL (Zhu & Pang, 2024) and CLIPN (Wang et al., 2023) require pretrained weights, we used the network with the pretrained weights provided by the authors. For industrial anomaly detection datasets (MVTec AD), we use the CLIP ViT-B/16+ (Gadre et al., 2024), pre-trained on the LAION-400M (Schuhmann et al., 2021) dataset, following the setup used in WinCLIP (Jeong et al., 2023). For a fair comparison, we also adopted the CLIP's image encoder as a feature extractor for the kNN baseline.

Metrics We use three metrics to evaluate the performance of the methods: the Area Under Receiver Operating Characteristics (AUROC), the Area Under the Precision Recall Curve (AUPRC), and the False Positive Rate at the 95% true positive rate (FPR95). AUROC and FPR95 are commonly used for the anomaly detection or out-of-distribution detection task (Ming et al., 2022). And we also use AUPRC because some datasets are imbalanced, with a significant disparity.

Computing resources We use a single NVIDIA RTX 3090 GPU for all experiments.

Hyperparameter The only hyperparameter in LAFT is the number of PCA components d. We typically choose d from 4 to 32 when guiding an attribute and from 32 to 384 when ignoring an attribute. Refer to Ablation Study for the impact of d on the performance. And we use $k = 30$ for the methods using kNN anomaly scoring (kNN and LAFT AD).

Baselines [TODO: We will provide the description and details of the baselines used in the experiments in the final version.]

Prompts To see the actual prompts used in the experiments, please refer to `laft/prompts` of the source code in the supplementary material. We referenced the prompts provided by CLIP (Radford et al., 2021) [1].

- **Colored MNIST**
  - **Number**: "zero", . . . , "twenty", "0", . . . , "20"
  - **Color**: "red", "green", "blue", "yellow", "orange", . . . , "black", "white"

- **Waterbirds**
  - **Bird**: Class names provided by the dataset and Birdsnap class names.
  - **Color**: "land", "bamboo", "forest", "ocean", and similar words.

- **CelebA**
  - **Hair color**: "blond", "black", "brown", "gray", "red", "white", and similar words.
  - **Eyeglasses**: "glasses", "eyeglasses", "sunglasses"
  - **Gender**: "man", "male", "boy", "woman", "female", "girl", "masculine", "feminine"

- **MVTec AD** and **VisA** We use the same prompts as WinCLIP (Jeong et al., 2023) for anomaly scoring. To compute the LAFT concept subspace, we use some more template-/state-level prompts for both LAFT General and LAFT Category. For LAFT Category, we use additional category-level prompts as Li et al. (2024).
  - **Template-level**: Jeong et al. (2023) and "an image of a {}", "a photo of the {}", . . .
  - **State-level**: Jeong et al. (2023) and "{} in perfect condition", "malformed {}", . . .
  - **Category-level (LAFT Category)**: "bottle with large breakage", "carpet with hole", . . .

Dataset Split

- **Colored MNIST** `R` denotes red, `G` denotes green, and `B` denotes blue colored digits. `0-4` and `5-9` denote the digits from 0 to 4 and from 5 to 9, respectively.
  - **Train**: `R/0-4` (16.67%)
  - **Test**: `R/0-4` (16.67%), `R/5-9` (16.67%), `GB/0-4` (33.33%), `GB/5-9` (33.33%)

- **Waterbirds** `Wbird` denotes waterbirds, and `Lbird` denotes landbirds. `Wback` denotes water background, and `Lback` denotes land background.
  - **Train**: `Wbird/Wback` (22.04%)
  - **Test**: `Wbird/Wback` (11.08%), `Wbird/Lback` (11.08%), `Lbird/Wback` (38.92%), `Lbird/Lback` (38.92%)

- **CelebA** `Blond` denotes blond hair, and `Glass` denotes eyeglasses. `-Blond` denotes non-blond hair, and `-Glass` denotes no eyeglasses.
  - **Train**: `Blond/-Glass` (14.66%)
  - **Test**: `Blond/Glass` (13.01%), `Blond/-Glass` (0.31%), `-Blond/Glass` (80.53%), `-Blond/-Glass` (6.15%)

- **MVTec AD** and **VisA** We use the same split as Bergmann et al. (2019) and Zou et al. (2022).

[1] https://github.com/openai/CLIP/blob/main/data/prompts.md

C Additional Experiments

C.1 Guiding and Ignoring Attributes

Table 6: Anomaly detection performance on the Colored MNIST datasets with different target criteria. We use five different seeds to split the training set for coloring each digit. Bold numbers indicate that the performance should be high (relevant), and underlined numbers indicate that the performance should be low (irrelevant).

Criteria	Number			Color		
	AUROC ↑	AUPRC ↑	FPR95 ↓	AUROC ↑	AUPRC ↑	FPR95 ↓
No guidance						
kNN	0.897 ± 0.004	0.895 ± 0.004	0.440 ± 0.003	0.811 ± 0.006	0.895 ± 0.005	0.581 ± 0.002
Guide						
Number	**0.985** ± 0.000	**0.984** ± 0.000	**0.069** ± 0.001	0.527 ± 0.001	0.664 ± 0.002	0.893 ± 0.002
Color	0.512 ± 0.001	0.536 ± 0.001	0.948 ± 0.001	**1.000** ± 0.000	**1.000** ± 0.000	**0.000** ± 0.000
Ignore						
Number	0.637 ± 0.002	0.656 ± 0.002	0.758 ± 0.001	**0.999** ± 0.000	**0.999** ± 0.000	**0.002** ± 0.000
Color	**0.967** ± 0.001	**0.962** ± 0.003	**0.144** ± 0.004	0.615 ± 0.004	0.735 ± 0.005	0.765 ± 0.001

Table 7: Anomaly detection performance on the Waterbirds dataset. Standard deviations are not reported because the method is deterministic. Bold numbers indicate that the performance should be high (relevant), and underlined numbers indicate that the performance should be low (irrelevant).

Criteria	Bird			Background		
	AUROC ↑	AUPRC ↑	FPR95 ↓	AUROC ↑	AUPRC ↑	FPR95 ↓
No guidance						
kNN	0.834	0.918	0.447	0.676	0.641	0.887
Guide						
Bird	**0.953**	**0.982**	**0.195**	0.706	0.726	0.872
Background	0.595	0.845	0.914	**0.976**	**0.972**	**0.101**
Ignore						
Bird	0.639	0.593	0.872	**0.745**	**0.876**	**0.626**
Background	**0.847**	**0.921**	**0.399**	0.500	0.481	0.934

The results in Table 6 and Table 7 show the performance of two attributes when one attribute is either guided or ignored. In datasets with two clearly distinguishable attributes, such as Colored MNIST, ignoring one attribute implicitly guides the other. However, in datasets composed of real-world images, such as Waterbirds, although there may appear to be only two attributes, there may actually be many more. As a result, while the performance of non-ignored attributes may improve slightly, the overall improvement is not significant. Nevertheless, when an attribute is ignored, we can confirm that it is indeed properly disregarded.

C.2 Ablation Study on PCA Components

To investigate the impact of the number of PCA components d on the performance of LAFT, we conduct an ablation study on semantic anomaly datasets. The results are shown in Table 8 and Table 9. We observe that the performance of LAFT is not very sensitive to the number of PCA components for a certain range of d. However, the performance drops significantly when d is too small or too large. This is because a small d cannot capture the concept subspace well, while a large d may include irrelevant information. Except for Eyeglasses attribute in CelebA, the best performance is achieved when d is between 14 and 24 for guiding attributes. For the Eyeglasses attribute, the best performance is when d is 6, which we expect because it is a simple binary classification problem of whether a person wears glasses or not, rather than distinguishing between different types within an attribute (like distinguishing between different numbers or different types of birds).

Table 8: Anomaly detection performance on Colored MNIST and Waterbirds datasets. We scan the number of PCA components d, which is the only hyperparameter of LAFT module.

	Colored MNIST: Number			Waterbirds: Bird		
d	AUROC ↑	AUPRC ↑	FPR95 ↓	AUROC ↑	AUPRC ↑	FPR95 ↓
Guide						
2	0.803	0.795	0.737	0.872	0.950	0.677
4	0.935	0.942	0.423	0.912	0.959	0.357
6	0.963	0.963	0.193	0.936	0.978	0.264
8	0.972	0.973	0.147	0.939	0.978	0.255
10	0.974	0.973	0.132	0.942	0.979	0.238
12	0.977	0.977	0.111	0.946	0.981	0.224
14	0.978	0.977	0.100	0.951	0.982	0.218
16	0.980	0.977	0.105	0.954	0.982	0.210
18	0.982	0.978	0.090	**0.956**	**0.984**	0.206
20	0.984	0.979	0.088	**0.956**	0.983	0.203
24	**0.985**	**0.982**	0.084	0.955	0.982	0.201
28	0.984	0.981	**0.079**	0.954	0.982	**0.193**
32	0.983	0.980	0.086	0.950	0.980	0.213
40	0.978	0.976	0.105	0.949	0.979	0.211
48	0.977	0.975	0.107	0.944	0.976	0.214
64	0.971	0.968	0.131	0.935	0.971	0.230
Ignore						
8	0.948	0.938	0.214	0.839	0.918	0.403
16	0.956	0.946	0.180	0.840	0.918	0.401
32	0.958	0.948	0.170	0.843	0.919	0.387
64	0.961	0.952	0.160	0.843	0.918	0.378
96	0.963	0.955	0.153	0.843	0.918	0.394
128	0.964	0.957	0.148	**0.847**	**0.921**	**0.399**
160	0.965	0.959	0.145	0.846	**0.921**	0.400
192	**0.967**	**0.962**	**0.144**	0.845	0.920	0.401
224	0.966	**0.962**	0.151	0.841	0.918	0.404
256	0.966	0.961	0.158	0.833	0.915	0.420
288	0.965	0.960	0.161	0.824	0.911	0.445
320	0.962	0.959	0.178	0.827	0.914	0.442
352	0.958	0.956	0.203	0.839	0.923	0.438
384	0.948	0.946	0.255	0.832	0.920	0.452

Table 9: Anomaly detection performance on CelebA dataset. We scan the number of PCA components d, which is the only hyperparameter of LAFT module.

	Hair color			Eyeglasses		
d	AUROC ↑	AUPRC ↑	FPR95 ↓	AUROC ↑	AUPRC ↑	FPR95 ↓
Guide						
2	0.526	0.884	0.953	0.659	0.114	0.863
4	0.905	0.983	0.558	0.973	0.764	0.082
6	0.926	0.987	0.384	**0.981**	**0.807**	**0.059**
8	0.940	0.990	0.365	0.979	0.786	0.068
10	0.946	0.991	0.319	0.975	0.750	0.083
12	0.948	0.991	0.317	0.976	0.770	0.094
14	**0.950**	**0.992**	0.298	0.976	0.783	0.098
16	0.948	0.991	0.306	0.973	0.753	0.111
18	0.947	0.991	**0.295**	0.968	0.720	0.135
20	0.946	0.991	0.305	0.953	0.601	0.181
24	0.943	0.991	0.317	0.952	0.571	0.186
28	0.940	0.990	0.339	0.948	0.557	0.210
32	0.939	0.990	0.344	0.942	0.474	0.206
40	0.933	0.989	0.380	0.925	0.406	0.245
48	0.928	0.988	0.385	0.914	0.384	0.285
64	0.918	0.986	0.409	0.897	0.327	0.318

C.3 LAFT WITH VARIOUS VISON-LANGUAGE MODELS

Table 10: Anomaly detection AUROC on semantic anomaly datasets. We compare vision-language models across diverse backbone architectures and training strategies.

	CLIP ViT			CLIP ConvNeXt		EVA02		SigLIP ViT		CoCa ViT	
Method	B-16	L-14	H-14	Base	XXL	B-16	L-14	B-16	L-16	B-32	L-14
Colored MNIST: Number											
kNN	0.897	0.907	0.898	0.842	0.875	0.667	0.731	0.861	0.823	0.854	0.894
MCM	0.693	0.798	0.871	0.739	0.839	0.402	0.640	0.747	0.682	0.621	0.704
ZOE	0.914	0.907	0.933	0.927	0.962	0.834	0.927	0.930	0.959	0.936	0.974
LAFT AD (Ours)	**0.985**	**0.989**	**0.996**	**0.976**	**0.995**	**0.890**	**0.943**	**0.985**	**0.991**	**0.987**	**0.988**
Waterbirds: Bird											
kNN	0.834	0.871	0.859	0.797	0.859	0.860	0.887	0.836	0.792	0.760	0.848
MCM	0.899	0.943	0.947	0.852	0.926	0.888	0.949	0.897	0.938	0.815	0.874
ZOE	0.924	0.938	0.937	0.906	0.935	0.936	0.944	0.933	0.942	0.884	0.925
LAFT AD (Ours)	**0.956**	**0.972**	**0.973**	**0.944**	**0.972**	**0.966**	**0.986**	**0.962**	**0.976**	**0.911**	**0.954**
CelebA: Hair color											
kNN	0.835	0.853	0.837	0.897	0.880	0.883	0.860	0.854	0.834	0.904	0.892
MCM	0.880	0.879	0.872	0.879	0.874	0.867	0.889	0.868	0.821	0.851	0.850
ZOE	0.934	0.943	0.936	0.925	0.932	0.930	0.935	0.915	0.877	0.941	0.901
LAFT AD (Ours)	**0.950**	**0.948**	**0.951**	**0.944**	**0.951**	**0.949**	**0.945**	**0.948**	**0.948**	**0.948**	**0.937**
CelebA: Eyeglasses											
kNN	0.824	0.810	0.750	0.775	0.781	0.762	0.755	0.758	0.700	0.796	0.781
MCM	0.081	0.117	0.132	0.126	0.115	0.216	0.248	0.122	0.149	0.106	0.136
ZOE	0.929	**0.988**	**0.989**	0.948	0.939	**0.987**	**0.979**	**0.992**	**0.992**	0.849	0.911
LAFT AD (Ours)	**0.981**	0.979	0.981	**0.974**	**0.981**	0.979	0.976	0.981	0.981	**0.981**	**0.976**

We compare the performance of LAFT with vision-language models across diverse backbone architectures and training strategies on the semantic anomaly datasets. We use architectures with pre-trained weights available in OpenCLIP (Ilharco et al., 2021). Specifically, we use ViT (Dosovitskiy et al., 2021), ConvNeXt (Liu et al., 2022), and EVA02 (Fang et al., 2024) as the backbone architectures, and we use the pre-trained weights from CLIP (Radford et al., 2021), EVA-CLIP (Sun et al., 2023), SigLIP (Zhai et al., 2023), and CoCa (Yu et al., 2022). The results are shown in Table 10. We observed that across various vision-language models, LAFT AD consistently outperformed the baseline method in most cases.

D FULL RESULTS ON MVTEC AD AND VISA

We provide the full results on MVTec AD and VisA datasets in Table 11 and Table 12, respectively. [TODO: We will add detailed analysis on the full results here in the final version.]

Table 11: Full anomaly detection and localization AUROC (%) on the MVTec AD dataset in few-shot settings (k-shot). We use five different sets of reference samples from the training set for each method. We bold the best performance and underline the second-best performance in each category. The performance of PatchCore is taken from the WinCLIP paper (Jeong et al., 2023).

Anomaly Detection

# K	Method	Bottle	Cable	Capsule	Carpet	Grid	Hazelnut	Leather	Metal Nut	Pill	Screw	Tile	Toothbrush	Transistor	Wood	Zipper	Average
0	WinCLIP	98.6	85.3	68.6	99.3	99.2	**92.4**	**100.**	**95.6**	81.5	71.8	99.9	85.6	89.0	97.6	91.0	90.4
	+ LAFT General (Ours)	**98.8**	**89.7**	**82.5**	**99.8**	99.3	92.3	**100.**	**95.6**	86.3	**84.0**	**100.**	**87.8**	**91.5**	**98.0**	91.7	**93.2**
	+ LAFT Category (Ours)	98.7	87.5	82.1	**99.8**	**99.4**	**92.4**	**100.**	**95.6**	**86.5**	82.1	**100.**	87.5	90.9	**98.0**	**92.2**	92.9
1	PatchCore*	95.3 ±0.8	63.6 ±10.	**97.3** ±0.7	99.0 ±0.9	97.8 ±0.3	**99.4** ±0.4	88.8 ±4.2	67.8 ±2.9	88.3 ±2.7	73.4 ±2.9	81.9 ±2.8	44.4 ±4.6	83.3 ±3.8	78.1 ±6.9	92.3 ±0.5	83.4 ±3.0
	InCTRL	98.8 ±0.9	83.3 ±2.1	68.8 ±9.9	99.4 ±0.6	98.8 ±1.3	97.4 ±0.6	99.9 ±0.1	95.3 ±4.5	89.8 ±2.4	74.3 ±3.0	99.9 ±0.1	**96.2** ±4.0	81.7 ±1.3	97.0 ±0.6	88.8 ±8.7	91.3 ±2.7
	WinCLIP	99.2 ±0.1	89.8 ±0.6	69.7 ±6.0	99.6 ±0.2	99.6 ±0.2	95.8 ±0.7	**100.** ±0.0	**98.5** ±0.5	**92.0** ±0.9	81.1 ±2.8	**100.** ±0.0	95.6 ±3.7	88.1 ±0.3	99.1 ±0.3	91.8 ±3.9	93.3 ±1.3
	+ LAFT General (Ours)	99.4 ±0.2	**91.3** ±0.4	76.3 ±7.4	**99.8** ±0.1	**99.7** ±0.2	95.4 ±0.6	**100.** ±0.0	98.2 ±0.6	91.1 ±0.7	**85.4** ±0.5	**100.** ±0.0	95.2 ±2.9	**91.5** ±0.1	99.1 ±0.3	94.6 ±3.9	**94.5** ±1.2
	+ LAFT Category (Ours)	**99.5** ±0.2	90.2 ±0.5	77.4 ±7.5	**99.8** ±0.1	**99.7** ±0.2	95.6 ±0.7	**100.** ±0.0	98.2 ±0.5	**92.0** ±0.2	83.2 ±1.0	**100.** ±0.0	94.7 ±3.4	91.0 ±0.1	**99.2** ±0.2	**94.7** ±2.8	94.4 ±1.2
2	PatchCore*	96.6 ±0.5	67.7 ±8.3	**97.9** ±0.7	98.5 ±1.0	98.3 ±0.6	**99.2** ±0.3	91.0 ±2.7	72.8 ±7.0	**93.2** ±3.8	77.7 ±8.5	82.9 ±2.9	49.0 ±3.8	85.9 ±3.5	90.0 ±4.3	94.0 ±2.1	86.3 ±3.3
	InCTRL	98.4 ±0.9	87.0 ±3.7	79.2 ±5.2	99.6 ±0.4	99.0 ±1.0	97.8 ±0.9	99.9 ±0.1	95.5 ±3.9	90.0 ±1.4	77.0 ±3.7	99.9 ±0.0	**97.8** ±1.4	85.4 ±3.4	97.1 ±0.4	93.9 ±0.8	93.2 ±1.8
	WinCLIP	99.2 ±0.1	90.0 ±0.8	80.0 ±3.1	99.6 ±0.2	99.6 ±0.3	96.9 ±0.2	**100.** ±0.0	**99.1** ±0.1	92.4 ±0.9	82.9 ±1.1	**100.** ±0.0	96.8 ±1.5	89.3 ±1.5	99.1 ±0.2	94.8 ±0.6	94.6 ±0.7
	+ LAFT General (Ours)	99.4 ±0.1	**91.4** ±0.4	89.0 ±3.4	**99.8** ±0.1	**99.7** ±0.3	96.4 ±0.3	**100.** ±0.0	98.8 ±0.2	91.5 ±0.7	**85.9** ±0.2	**100.** ±0.0	96.7 ±1.2	**91.9** ±0.4	**99.2** ±0.2	97.1 ±0.3	**95.8** ±0.5
	+ LAFT Category (Ours)	**99.6** ±0.1	90.4 ±0.6	90.1 ±3.1	**99.8** ±0.1	**99.7** ±0.3	96.6 ±0.3	**100.** ±0.0	98.8 ±0.2	92.3 ±0.5	83.7 ±0.7	**100.** ±0.0	95.9 ±1.7	91.5 ±0.5	**99.2** ±0.2	**97.3** ±0.3	95.7 ±0.6
4	PatchCore*	96.6 ±0.5	67.7 ±8.3	97.9 ±0.7	98.5 ±1.0	98.3 ±0.6	**99.2** ±0.3	91.0 ±2.7	72.8 ±7.0	**93.2** ±3.8	77.7 ±8.5	82.9 ±2.9	49.0 ±3.8	85.9 ±3.5	90.0 ±4.3	94.0 ±2.1	88.8 ±2.6
	InCTRL	98.4 ±1.0	88.9 ±2.6	75.9 ±5.7	99.6 ±0.3	98.8 ±1.0	97.7 ±0.5	99.9 ±0.1	97.1 ±2.5	91.2 ±1.4	78.3 ±3.5	99.9 ±0.0	**98.7** ±0.9	87.2 ±2.7	97.4 ±0.1	94.2 ±0.9	93.6 ±1.6
	WinCLIP	99.3 ±0.3	90.1 ±0.8	82.5 ±1.0	99.6 ±0.2	99.6 ±0.2	97.2 ±0.3	**100.** ±0.0	**99.3** ±0.3	92.3 ±0.3	83.5 ±1.9	**100.** ±0.0	97.7 ±0.3	90.3 ±1.1	**99.3** ±0.1	94.6 ±0.9	95.0 ±0.5
	+ LAFT General (Ours)	99.5 ±0.1	**91.6** ±0.4	91.9 ±1.0	**99.9** ±0.0	**99.7** ±0.2	96.7 ±0.2	**100.** ±0.0	99.1 ±0.3	91.4 ±0.1	**86.1** ±0.3	**100.** ±0.0	97.0 ±0.9	**92.1** ±0.4	**99.3** ±0.1	97.0 ±0.3	**96.1** ±0.3
	+ LAFT Category (Ours)	**99.6** ±0.1	90.7 ±0.6	**92.5** ±1.0	99.8 ±0.1	99.6 ±0.3	96.8 ±0.3	**100.** ±0.0	99.1 ±0.3	92.3 ±0.2	84.4 ±0.8	**100.** ±0.0	96.1 ±0.3	91.7 ±0.3	**99.3** ±0.1	**97.3** ±0.4	96.0 ±0.3
8	InCTRL	98.2 ±0.7	91.2 ±2.2	71.5 ±4.7	99.8 ±0.1	98.7 ±0.8	**97.5** ±0.2	**100.** ±0.0	97.3 ±1.5	91.3 ±1.2	81.2 ±3.6	**100.** ±0.0	**98.7** ±0.9	88.8 ±1.8	97.6 ±0.2	95.1 ±0.7	93.8 ±1.3
	WinCLIP	99.4 ±0.3	91.2 ±0.9	83.3 ±0.6	99.8 ±0.1	99.4 ±0.1	97.3 ±0.3	**100.** ±0.0	**99.3** ±0.2	**93.0** ±0.2	84.9 ±1.4	**100.** ±0.0	97.4 ±0.9	90.5 ±0.7	99.3 ±0.1	95.1 ±0.8	95.3 ±0.4
	+ LAFT General (Ours)	99.5 ±0.1	**92.0** ±0.4	92.6 ±1.2	**99.9** ±0.0	**99.5** ±0.1	96.7 ±0.3	**100.** ±0.0	99.2 ±0.2	92.1 ±0.1	**86.2** ±0.3	**100.** ±0.0	96.9 ±0.5	**92.3** ±0.3	99.2 ±0.1	97.4 ±0.5	**96.3** ±0.3
	+ LAFT Category (Ours)	**99.7** ±0.1	91.3 ±0.7	**92.9** ±0.7	**99.9** ±0.0	**99.5** ±0.1	96.8 ±0.3	**100.** ±0.0	99.2 ±0.2	92.7 ±0.2	85.0 ±0.8	**100.** ±0.0	96.3 ±1.0	91.9 ±0.3	**99.4** ±0.1	**97.6** ±0.5	96.2 ±0.3

Anomaly Localization

# K	Method	Bottle	Cable	Capsule	Carpet	Grid	Hazelnut	Leather	Metal Nut	Pill	Screw	Tile	Toothbrush	Transistor	Wood	Zipper	Average
0	WinCLIP	64.0 ±0.0	69.1 ±0.0	63.5 ±0.0	71.6 ±0.0	67.4 ±0.0	**70.1** ±0.0	67.2 ±0.0	**57.7** ±0.0	23.5 ±0.0	59.3 ±0.0	**65.3** ±0.0	45.1 ±0.0	85.5 ±0.0	64.4 ±0.0	61.0 ±0.0	62.3 ±0.0
	+ LAFT General (Ours)	64.3 ±0.0	**72.1** ±0.0	58.0 ±0.0	**72.5** ±0.0	67.6 ±0.0	70.0 ±0.0	**68.5** ±0.0	**57.7** ±0.0	**45.1** ±0.0	**62.8** ±0.0	65.2 ±0.0	**49.0** ±0.0	**87.7** ±0.0	**64.7** ±0.0	62.7 ±0.0	64.5 ±0.0
	+ LAFT Category (Ours)	**64.9** ±0.0	70.1 ±0.0	**72.9** ±0.0	71.5 ±0.0	**67.8** ±0.0	69.9 ±0.0	**68.5** ±0.0	55.5 ±0.0	43.2 ±0.0	61.2 ±0.0	**65.3** ±0.0	46.6 ±0.0	87.4 ±0.0	64.6 ±0.0	**62.8** ±0.0	**64.8** ±0.0
1	WinCLIP	94.6 ±0.0	**92.3** ±0.6	89.6 ±1.4	**98.8** ±0.2	**94.9** ±1.5	**97.6** ±0.5	99.2 ±0.0	**87.7** ±1.2	90.3 ±0.6	**93.6** ±0.6	89.7 ±0.8	97.5 ±0.6	89.1 ±1.1	**89.3** ±1.0	**87.4** ±1.2	**92.8** ±0.8
	+ LAFT General (Ours)	**96.7** ±0.1	84.9 ±0.4	90.2 ±1.0	98.3 ±0.2	94.4 ±1.7	**97.6** ±0.5	**99.3** ±0.0	87.4 ±1.2	**91.1** ±0.6	92.4 ±0.5	85.4 ±0.6	97.4 ±0.6	**92.3** ±0.5	89.1 ±1.0	83.5 ±0.8	92.0 ±0.6
	+ LAFT Category (Ours)	**96.7** ±0.1	92.2 ±0.6	**90.4** ±0.8	98.4 ±0.2	94.5 ±1.6	**97.6** ±0.5	99.1 ±0.0	87.5 ±1.2	80.3 ±0.5	90.1 ±0.7	**91.2** ±0.7	**97.6** ±0.6	92.0 ±0.5	**89.3** ±1.0	85.6 ±0.9	92.2 ±0.7
2	WinCLIP	94.8 ±0.2	**92.7** ±0.4	91.7 ±0.2	**98.8** ±0.2	**94.8** ±1.5	98.1 ±0.3	99.2 ±0.0	**89.1** ±1.1	90.8 ±0.6	**94.4** ±0.5	90.1 ±0.2	**97.8** ±0.3	91.3 ±0.6	**89.6** ±0.9	**88.0** ±0.8	**93.4** ±0.5
	+ LAFT General (Ours)	**96.9** ±0.1	84.9 ±0.3	89.0 ±0.4	97.1 ±0.2	94.4 ±1.6	**98.2** ±0.3	**99.3** ±0.0	88.9 ±1.1	**91.6** ±0.5	93.0 ±0.5	90.7 ±0.1	97.6 ±0.2	**93.2** ±0.3	89.5 ±0.9	83.6 ±0.7	92.5 ±0.5
	+ LAFT Category (Ours)	**96.9** ±0.1	92.5 ±0.4	**92.2** ±0.4	98.4 ±0.2	94.2 ±1.6	**98.2** ±0.3	99.1 ±0.0	88.9 ±1.1	80.5 ±0.6	92.6 ±0.5	**91.7** ±0.1	**97.8** ±0.2	93.0 ±0.3	**89.6** ±0.9	**88.0** ±0.7	92.9 ±0.5
4	WinCLIP	94.8 ±0.2	**92.9** ±0.3	92.3 ±0.3	**98.7** ±0.2	**95.3** ±1.0	98.2 ±0.2	99.2 ±0.0	**90.8** ±0.8	91.2 ±0.3	**94.7** ±0.5	90.3 ±0.2	98.1 ±0.3	92.2 ±0.7	**89.7** ±0.6	**88.8** ±0.5	**93.8** ±0.4
	+ LAFT General (Ours)	**97.1** ±0.1	84.9 ±0.2	89.3 ±0.3	98.3 ±0.3	94.9 ±1.2	**98.3** ±0.2	**99.3** ±0.0	90.6 ±0.8	**92.1** ±0.3	93.2 ±0.5	90.8 ±0.2	98.0 ±0.2	**93.7** ±0.3	89.4 ±0.6	84.2 ±0.4	92.9 ±0.4
	+ LAFT Category (Ours)	97.0 ±0.1	92.6 ±0.3	**92.9** ±0.3	97.7 ±0.3	94.9 ±1.1	98.2 ±0.2	99.0 ±0.1	90.6 ±0.8	80.9 ±0.4	92.9 ±0.5	**92.1** ±0.2	**98.2** ±0.2	93.5 ±0.4	89.6 ±0.6	88.6 ±0.4	93.2 ±0.4
8	WinCLIP	94.8 ±0.2	**93.1** ±0.2	92.5 ±0.2	**98.6** ±0.2	**95.3** ±0.2	**98.3** ±0.2	99.2 ±0.0	**91.6** ±0.6	91.5 ±0.2	**95.2** ±0.7	90.3 ±0.2	**98.3** ±0.0	93.2 ±0.5	**90.0** ±0.5	**88.8** ±0.4	**94.0** ±0.3
	+ LAFT General (Ours)	**97.1** ±0.1	84.9 ±0.1	87.5 ±0.3	98.0 ±0.2	94.7 ±0.3	**98.3** ±0.2	99.2 ±0.0	91.5 ±0.6	**92.3** ±0.2	93.4 ±0.9	85.8 ±0.2	98.1 ±0.0	**94.2** ±0.3	89.3 ±0.5	84.0 ±0.3	92.6 ±0.3
	+ LAFT Category (Ours)	97.0 ±0.1	92.9 ±0.2	**93.1** ±0.1	98.2 ±0.2	94.7 ±0.3	**98.3** ±0.2	**99.3** ±0.0	91.4 ±0.6	80.9 ±0.3	93.2 ±0.9	**91.7** ±0.2	**98.3** ±0.0	94.0 ±0.3	**90.0** ±0.5	88.6 ±0.3	93.4 ±0.3

Table 12: Full anomaly detection and localization AUROC (%) on the VisA dataset in few-shot settings (k-shot). We use five different sets of reference samples from the training set for each method. We bold the best performance and underline the second-best performance in each category.

Anomaly Detection

# K	Method	Candle	Capsules	Cashew	Chewinggum	Fryum	Macaroni1	Macaroni2	Pcb1	Pcb2	Pcb3	Pcb4	Pipe Fryum	Average
0	WinCLIP	96.6 ± 0.0	79.3 ± 0.0	92.5 ± 0.0	96.1 ± 0.0	74.5 ± 0.0	74.5 ± 0.0	65.4 ± 0.0	71.2 ± 0.0	44.1 ± 0.0	57.5 ± 0.0	85.2 ± 0.0	70.8 ± 0.0	75.6 ± 0.0
	+ LAFT General (Ours)	**96.7** ± 0.0	**84.2** ± 0.0	**93.3** ± 0.0	97.8 ± 0.0	**84.4** ± 0.0	**80.0** ± 0.0	**66.0** ± 0.0	83.5 ± 0.0	51.4 ± 0.0	65.2 ± 0.0	**90.7** ± 0.0	73.9 ± 0.0	**80.6** ± 0.0
	+ LAFT Category (Ours)	**96.7** ± 0.0	84.1 ± 0.0	**93.3** ± 0.0	**98.6** ± 0.0	82.6 ± 0.0	**80.0** ± 0.0	65.4 ± 0.0	**84.4** ± 0.0	**56.2** ± 0.0	**70.5** ± 0.0	88.1 ± 0.0	**74.0** ± 0.0	81.2 ± 0.0
1	InCTRL	84.2 ± 4.8	77.1 ± 2.5	91.1 ± 2.1	96.7 ± 0.4	**93.8** ± 1.4	**85.5** ± 3.7	**79.0** ± 4.3	76.8 ± 8.8	**74.8** ± 3.9	76.0 ± 4.9	88.3 ± 3.0	**97.1** ± 1.6	85.0 ± 4.3
	WinCLIP	97.4 ± 0.2	82.4 ± 1.1	93.2 ± 0.5	98.4 ± 0.3	89.0 ± 1.0	82.4 ± 1.8	73.4 ± 1.9	81.0 ± 5.7	59.2 ± 1.9	66.3 ± 2.8	79.0 ± 6.7	89.5 ± 1.7	82.6 ± 2.1
	+ LAFT General (Ours)	**97.6** ± 0.2	**85.6** ± 0.8	**94.0** ± 0.4	**99.1** ± 0.2	92.4 ± 1.9	84.9 ± 1.3	73.0 ± 1.8	85.1 ± 0.5	58.0 ± 1.1	70.2 ± 2.0	**89.1** ± 2.6	93.0 ± 1.0	85.1 ± 1.2
	+ LAFT Category (Ours)	**97.6** ± 0.2	85.5 ± 0.8	93.9 ± 0.4	98.9 ± 0.2	92.2 ± 1.3	84.9 ± 1.2	72.1 ± 1.6	**85.3** ± 0.3	64.1 ± 1.2	**76.3** ± 1.8	86.9 ± 3.1	91.3 ± 1.1	**85.8** ± 1.1
2	InCTRL	85.0 ± 4.0	78.9 ± 1.1	92.1 ± 1.6	97.3 ± 0.4	**95.3** ± 0.3	**87.9** ± 1.4	**80.1** ± 2.9	81.1 ± 9.1	**77.9** ± 2.0	**81.7** ± 2.5	86.1 ± 5.7	**97.4** ± 1.3	**86.7** ± 2.7
	WinCLIP	97.6 ± 0.2	83.7 ± 1.0	93.4 ± 0.8	98.6 ± 0.1	90.4 ± 0.5	83.8 ± 1.0	75.1 ± 0.9	83.4 ± 1.8	60.6 ± 2.0	71.1 ± 3.6	84.3 ± 5.3	89.8 ± 1.6	84.3 ± 1.6
	+ LAFT General (Ours)	**97.8** ± 0.1	**86.6** ± 0.9	**94.1** ± 0.5	**99.2** ± 0.1	93.0 ± 1.1	85.9 ± 0.8	74.6 ± 0.8	86.2 ± 0.9	58.7 ± 1.2	73.2 ± 2.5	**90.4** ± 2.6	93.1 ± 0.8	86.1 ± 1.0
	+ LAFT Category (Ours)	**97.8** ± 0.1	86.1 ± 0.9	94.0 ± 0.4	98.9 ± 0.2	93.2 ± 0.6	85.4 ± 0.7	73.1 ± 0.9	**87.7** ± 1.2	64.7 ± 1.5	78.6 ± 1.9	88.9 ± 3.2	91.4 ± 0.8	**86.7** ± 1.0
4	InCTRL	88.5 ± 1.6	80.3 ± 1.4	92.9 ± 0.9	97.5 ± 0.3	**94.9** ± 0.8	**88.0** ± 2.7	**81.7** ± 4.5	86.9 ± 3.5	**78.2** ± 2.3	**84.2** ± 1.8	89.2 ± 4.0	**98.5** ± 0.2	**88.4** ± 2.0
	WinCLIP	97.8 ± 0.1	85.1 ± 1.5	94.3 ± 0.7	98.8 ± 0.1	91.2 ± 0.8	81.7 ± 1.9	76.2 ± 1.2	85.5 ± 1.8	60.7 ± 3.3	76.0 ± 1.5	88.2 ± 4.7	89.8 ± 1.4	85.4 ± 1.6
	+ LAFT General (Ours)	**97.9** ± 0.1	**87.8** ± 1.2	**94.6** ± 0.4	**99.4** ± 0.1	93.4 ± 0.8	84.4 ± 1.1	75.6 ± 1.1	87.3 ± 1.4	58.2 ± 1.8	76.6 ± 1.3	**91.9** ± 2.5	93.4 ± 0.4	86.7 ± 1.0
	+ LAFT Category (Ours)	**97.9** ± 0.1	87.4 ± 1.1	**94.6** ± 0.4	99.1 ± 0.2	93.4 ± 0.6	84.2 ± 1.3	74.5 ± 1.1	**90.1** ± 2.4	65.1 ± 2.2	81.6 ± 0.9	91.1 ± 2.5	91.6 ± 0.5	87.6 ± 1.1
8	InCTRL	89.3 ± 0.8	81.1 ± 1.9	92.2 ± 1.8	97.7 ± 0.3	**95.3** ± 0.7	**89.5** ± 1.8	**80.9** ± 4.9	89.8 ± 1.7	**80.1** ± 1.6	**85.9** ± 1.4	93.4 ± 2.7	**98.1** ± 0.3	**89.4** ± 1.7
	WinCLIP	98.0 ± 0.1	86.4 ± 0.6	94.6 ± 0.9	98.8 ± 0.2	91.7 ± 0.8	82.1 ± 1.2	75.7 ± 2.5	87.8 ± 1.6	63.6 ± 1.2	76.2 ± 1.5	93.5 ± 1.8	90.2 ± 1.0	86.5 ± 1.1
	+ LAFT General (Ours)	**98.1** ± 0.1	**88.8** ± 0.6	94.7 ± 0.5	**99.4** ± 0.2	93.5 ± 1.0	84.9 ± 1.2	75.2 ± 2.2	88.9 ± 1.7	59.9 ± 0.4	76.6 ± 1.0	94.6 ± 1.1	93.5 ± 0.3	87.3 ± 0.9
	+ LAFT Category (Ours)	**98.1** ± 0.1	88.4 ± 0.6	**94.8** ± 0.5	99.1 ± 0.2	93.9 ± 0.7	84.6 ± 0.5	74.9 ± 2.3	**92.9** ± 3.1	66.4 ± 0.6	81.9 ± 1.0	**94.7** ± 1.6	91.7 ± 0.5	88.5 ± 1.0

Anomaly Localization

# K	Method	Candle	Capsules	Cashew	Chewinggum	Fryum	Macaroni1	Macaroni2	Pcb1	Pcb2	Pcb3	Pcb4	Pipe Fryum	Average
0	WinCLIP	**78.7** ± 0.0	77.7 ± 0.0	**61.2** ± 0.0	75.4 ± 0.0	48.3 ± 0.0	65.6 ± 0.0	54.2 ± 0.0	40.1 ± 0.0	38.6 ± 0.0	47.2 ± 0.0	66.8 ± 0.0	25.7 ± 0.0	56.6 ± 0.0
	+ LAFT General (Ours)	78.6 ± 0.0	78.2 ± 0.0	58.1 ± 0.0	74.8 ± 0.0	78.1 ± 0.0	**68.0** ± 0.0	**55.1** ± 0.0	78.6 ± 0.0	42.4 ± 0.0	52.4 ± 0.0	**73.8** ± 0.0	26.1 ± 0.0	63.7 ± 0.0
	+ LAFT Category (Ours)	77.9 ± 0.0	**80.3** ± 0.0	60.8 ± 0.0	**76.9** ± 0.0	**79.0** ± 0.0	67.7 ± 0.0	54.5 ± 0.0	**79.9** ± 0.0	**44.6** ± 0.0	**53.0** ± 0.0	70.6 ± 0.0	**26.3** ± 0.0	**64.3** ± 0.0
1	WinCLIP	**96.2** ± 0.3	83.1 ± 0.9	**96.7** ± 0.6	**98.3** ± 0.1	88.0 ± 1.5	76.6 ± 2.3	72.9 ± 1.3	**93.9** ± 1.0	**87.3** ± 0.9	**83.0** ± 2.2	92.3 ± 0.9	90.4 ± 1.5	**88.2** ± 1.1
	+ LAFT General (Ours)	96.1 ± 0.4	85.4 ± 0.6	96.1 ± 0.8	96.1 ± 0.2	**90.8** ± 0.6	74.5 ± 2.6	**73.3** ± 1.3	91.4 ± 0.5	76.0 ± 1.1	74.7 ± 1.6	**92.4** ± 0.7	**94.9** ± 0.8	86.8 ± 0.9
	+ LAFT Category (Ours)	**96.2** ± 0.4	**85.6** ± 0.5	96.4 ± 0.7	90.2 ± 0.3	83.4 ± 0.3	**77.2** ± 1.3	72.8 ± 1.3	88.6 ± 0.6	84.8 ± 1.0	77.7 ± 1.7	90.5 ± 0.6	89.4 ± 0.8	86.1 ± 0.8
2	WinCLIP	**96.3** ± 0.2	83.7 ± 1.0	**96.8** ± 0.2	**98.3** ± 0.1	89.1 ± 1.1	**77.7** ± 1.5	74.2 ± 0.9	94.5 ± 0.3	**87.6** ± 1.5	**85.0** ± 2.3	**93.5** ± 0.9	89.8 ± 1.3	**88.9** ± 0.9
	+ LAFT General (Ours)	96.2 ± 0.2	85.8 ± 0.6	96.0 ± 0.3	96.2 ± 0.1	**90.8** ± 0.2	75.9 ± 1.6	**74.7** ± 0.9	91.6 ± 0.3	76.0 ± 1.6	75.8 ± 1.7	93.1 ± 0.8	**94.7** ± 0.6	87.2 ± 0.7
	+ LAFT Category (Ours)	96.2 ± 0.2	**86.0** ± 0.5	96.3 ± 0.2	98.2 ± 0.1	83.4 ± 0.2	77.6 ± 0.8	74.0 ± 0.9	**96.6** ± 0.4	85.0 ± 1.6	79.0 ± 1.8	91.2 ± 0.8	89.0 ± 0.7	87.7 ± 0.7
4	WinCLIP	**96.5** ± 0.1	84.2 ± 0.7	**97.0** ± 0.1	**98.3** ± 0.0	89.7 ± 0.7	**77.3** ± 1.8	74.7 ± 0.6	94.9 ± 0.5	**88.3** ± 0.8	**84.9** ± 1.8	**94.2** ± 0.8	89.9 ± 1.5	**89.2** ± 0.8
	+ LAFT General (Ours)	96.3 ± 0.1	86.1 ± 0.4	96.2 ± 0.2	96.1 ± 0.1	**90.9** ± 0.1	75.5 ± 1.9	**75.3** ± 0.6	91.6 ± 0.2	76.4 ± 0.8	75.3 ± 1.2	93.4 ± 0.7	**94.7** ± 0.7	87.3 ± 0.6
	+ LAFT Category (Ours)	96.4 ± 0.1	**86.3** ± 0.4	96.5 ± 0.1	98.2 ± 0.0	83.4 ± 0.1	77.1 ± 1.0	74.6 ± 0.6	**97.0** ± 0.5	85.7 ± 0.8	78.7 ± 1.3	91.5 ± 0.7	89.0 ± 0.7	87.9 ± 0.5
8	WinCLIP	96.6 ± 0.1	84.9 ± 0.5	**97.2** ± 0.1	**98.3** ± 0.0	90.5 ± 0.6	**77.9** ± 1.6	74.9 ± 0.6	95.5 ± 0.4	**89.0** ± 0.5	**85.5** ± 0.9	95.3 ± 0.2	90.1 ± 1.1	**89.6** ± 0.6
	+ LAFT General (Ours)	**96.7** ± 0.1	86.0 ± 0.3	96.7 ± 0.2	96.6 ± 0.1	**90.9** ± 0.2	76.7 ± 1.7	**75.6** ± 0.6	92.0 ± 0.4	76.8 ± 0.6	75.5 ± 0.6	94.0 ± 0.3	**94.9** ± 0.5	87.7 ± 0.5
	+ LAFT Category (Ours)	96.6 ± 0.1	**86.7** ± 0.3	96.6 ± 0.2	90.0 ± 0.1	83.5 ± 0.1	77.1 ± 0.9	74.8 ± 0.6	**97.4** ± 0.4	86.4 ± 0.5	78.9 ± 0.7	**95.4** ± 0.2	89.1 ± 0.5	87.7 ± 0.4

E ALGORITHM

The following pseudocode demonstrates the implementation of LAFT AD, using a syntax similar to NumPy, as the notation used in (Radford et al., 2021).

```
# model: the CLIP model
# prompts: the list of prompts provided by the user
# train_images: the collection of normal images
# test_images: the collection of images to be tested
# d: the number of principle axis

# Compute attribute subspace
text_features = model.encode_text(prompts)
pair_diffs = pairwise_difference(text_features)
basis = pca(pair_diffs, d)

# Encode images
train_features = model.encode_image(train_images)
test_features  = model.encode_image(test_images)

# Guide
train_laft_features = inner_projection(train_features, basis)
test_laft_features  = inner_projection(test_features,  basis)

anomaly_scores = knn(train_laft_features, test_laft_features)

# Ignore
train_laft_features = orthogonal_projection(train_features, basis)
test_laft_features  = orthogonal_projection(test_features,  basis)

anomaly_scores = knn(train_laft_features, test_laft_features)
```