Unsupervised Visual Anomaly Detection with Score-Based Generative Model

Anonymous authors
Paper under double-blind review

Abstract

We consider leveraging the deviated outputs and gradient information from generative models due to out of distribution samples in visual anomaly detection (AD). Visual AD has been critical problems and widely discussed. However, in various applications, abnormal image samples are very rare and difficult to collect. In this paper, we focus on the unsupervised visual anomaly detection and localization tasks and propose a novel score-based generative model applicable to general cases. Our work is inspired by the fact that injected noises to the original image through forward diffusion process may reveal the image defects in the reverse process (i.e., reconstruction). First, due to the differences of normal pixels between the reconstructed and original images, we propose to use a score-based generative model and associated score values as metric to gauge the defects. Second, to accelerate inference process, a novel T scales approach is developed to reduce redundant information from adjacent moments while leverages the information provided by the score model at different moments. These practices allows our model to generalize visual AD in an unsupervised manner while maintain reasonably good performance. We evaluate our method on several datasets to demonstrate its effectiveness.

1 Introduction

Anomaly detection (AD) plays key roles in the variety of applications, including industrial manufacturing (Bergmann et al., 2019; Zavrtanik et al., 2021a; Hou et al., 2021; Li et al., 2021; Reiss et al., 2021b) and medical analysis (Schlegl et al., 2017; 2019; Ouyang et al., 2020; Tang et al., 2021). As anomalous samples are rare in real-world scenarios, they lead to challenges especially for supervised learning models. Alternative solutions through generative models in an unsupervised manner have been prevailing recently, including AutoEncoder (AE) (Kingma & Welling, 2013), Generative Adversarial Network (GAN) (Goodfellow et al., 2014), Flow (Dinh et al., 2016) and their variants. Nonetheless, difficulty remains in applying these methods in high-dimensional data such as images. For example, AE is known for its blurring reconstructions and indistinguishable defects, and GAN or Flow models need additional overhead in developing encoders or dedicated dimensionality reduction modules, which are both time and computational consuming.

We are inspired by the recent score-based generative model (Song et al., 2021) through stochastic differential equation (SDE) and diffusion probabilistic model (Ho et al., 2020a) that achieve state-of-the-art performance in image generation. Our methodology is based on the assumption that *the anomalous data lie in the low probability density region of the normal data distribution*. In addition, we propose a more reliable score-based metric for the anomaly detection, namely, **Score-AD**. Through score-based generative models, two scores are examined, namely, **self-score** and **whole-score**. Self-score measures the distance towards the original data while whole-score measures that towards the high probability density region of the training data. By providing different "stimulation" in the reverse process, two whole-scores can be achieved and their divergence will be leveraged for anomaly detection and localization. Fig.1 shows the framework.

There are three challenges to be solved in our paper. First, like existing works, a generative model is trained first on normal data with the aim of converting abnormal data back to normal ones after going through the model. However, our observations have identified key issues that after the diffusion process, the normal pixels do not exactly match the original image after reverse diffusion process. Therefore, a simple pixel-wise comparison between reconstructed and original data for AD is not

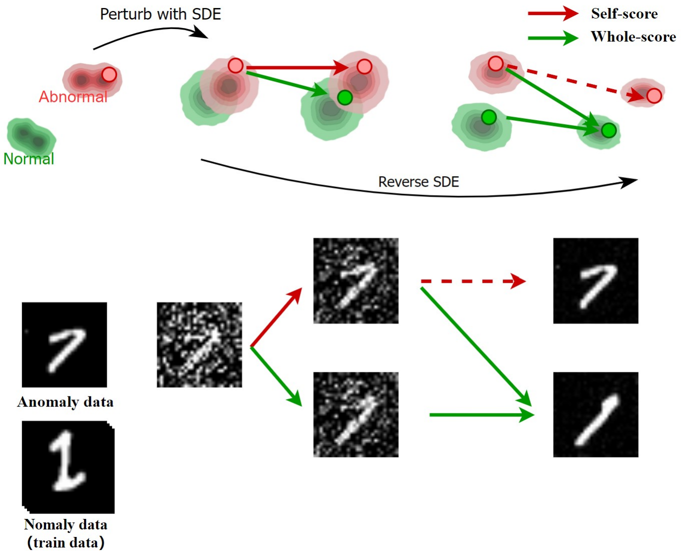


Figure 1: The red dot represents abnormal sample. The red and green contour plots represent the distributions of normal and abnormal data, respectively. Given an anomaly data, on the left, the noise is injected to blur it. When the model iterating through **whole-score**, the defects are recovered into normal mode as indicated by green arrow, while through **self-score**, the noisy image returns to the original image as indicated by the red arrow. The two noisy images in this process, one that the defect is gradually recovering to origin, require a larger **whole-score** value to change the defect than one that have changed a part of the defect.

feasible anymore. Based on the initial assumption that defects lie in the low probability density region of the normal data distribution which can easily satisfy, we propose a new metric through the whole-score to mitigate this issue. Second, the reverse process of score-model is less-efficient for certain setting of hyperparameters, e.g., a larger initial moment t in our case. Instead of launching a large t, we propose to investigate a set of smaller parameters, i.e., $\{t\}$ with only few steps in reverse for each. This ensemble strategy allows us to consider different "reconstructed-original" data pairs and enable a more reliable detection mechanism, termed as T scales in this work. Third, most existing unsupervised AD models rely on pre-trained networks for feature extraction and thus rely external data for good performance. But our goal is to explore the characteristics of the score model applied to unsupervised AD and to provide a simple and effective scheme not dependent on other models. Therefore our score model is just trained on normal data in an unsupervised fashion. We evaluate our method on several datasets to to verify the effectiveness of our method. Specifically, our method achieves the state-of-the-art (SOTA) **98.24 image-level AUC** and **97.78 pixel-level AUC** on the challenging MVTec AD dataset (Bergmann et al., 2019). Besides, we not only explain the principle of our method, but also conduct comparative experiments and ablation studies to analyze our method.

2 Related work

In this section, we mainly review previous AD approaches based on generative model. AE or Variational AutoEncoder (VAE) is trained to generate normal data but fail to reconstruct the abnormal samples. However, the output is often blurred (Hou et al., 2021), or defects are well restored (Zavrtanik et al., 2021b) due to the nature of generalization. To fix these problems, recent works, including memory mechanism (Gong et al., 2019; Hou et al., 2021), SSIM Loss (Wang et al., 2004), Mask strategy (Zavrtanik et al., 2021b), denoising autoencoder (Huang et al., 2019), forgery defect (Zavrtanik et al., 2021a), are developed and discussed. However, these methods are recently superseded by the following competitive generative models.

GANs and its generative and discriminative networks have been leveraged in AD tasks recently. In particular, the generative networks learn to map the noise from a latent space to anomaly-free data

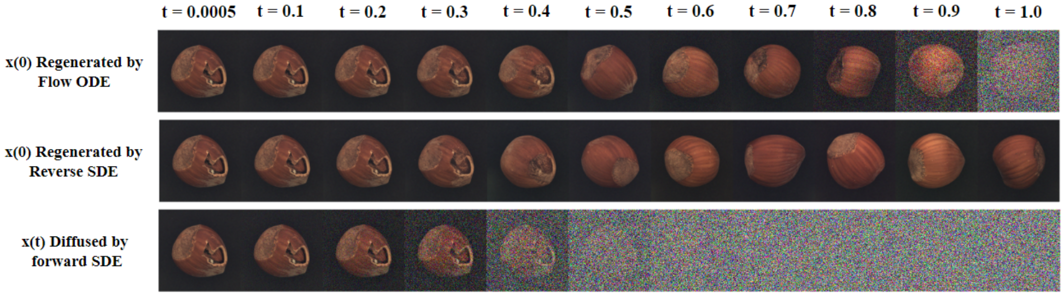

t = 0.0005
t = 0.1
t = 0.2
t = 0.3
t = 0.4
t = 0.5
t = 0.6
t = 0.7
t = 0.8
t = 0.9
t = 1.0
x(0) Regenerated by Flow ODE
x(0) Regenerated by Reverse SDE
x(t) Diffused by forward SDE

Figure 2: Regenerate samples by solving the probability flow ODE and reverse SDE on $[0, t]$ with initial point $\mathbf{x}(t) = \mu(t)\mathbf{x}(0) + \sigma(t)\mathbf{z}(t)$ for $\mathbf{z}(t) \sim \mathcal{N}(0, \mathrm{I})$ trained on MVTec AD dataset.

distribution, while the discriminant network determines whether it comes from anomaly-free data distribution. However, as GANs lacks dedicated encoders to produce hidden variables of the input data, additional efforts are required to develop networks to search for the hidden variable (Schlegl et al., 2019; Akçay et al., 2019; Akcay et al., 2018).

Another line of works is based on the approach “normalized flow” that learns and manages to map the distribution of normal data reversely to a simple Gaussian distribution. The distribution of normal data supposes to be close to the center of the Gaussian kernels (i.e., high-density region), while the abnormal data shall reside in the low probability density region, an indicator for data with defects in the testing phase. However, as the hidden layer dimension must match the data dimension in these methods, when working on data of larger size, e.g., high-resolution images, the model parameters expand quickly. Therefore, the flow-based methods (Rudolph et al., 2020; 2022; Gudovskiy et al., 2021; Yu1 et al., 2022) usually take feature maps extracted by pre-trained network on a large-scale dataset, e.g., ImageNet.

3 Background

3.1 Score-based Generative Model for AD

The AD model discussed in this paper is trained on normal dataset $\mathbf{X}_N$ in an unsupervised fashion while tested on a blend of normal and abnormal dataset $\mathbf{X}_{N+A}$. The hypothesis is that anomaly $\mathbf{x} \in \mathbf{X}_{N+A}$ distributes differently with $\mathbf{X}_N$. Our framework is inspired by the Denoising AutoEncoder (DAE) (Huang et al., 2019). We can extend the usage of DAE by integrating diffusion process and score-based generative models (Meng et al., 2021; Yoon et al., 2021). In particular, we can model a diffusion process by forward SDE to inject a certain amount of noise to the data, and implement the reverse diffusion process as denoising. Ideally, the defects would be treated as noise and recovered to normal data. This procedure is shown in Fig. 2. With the reconstructed data through the reverse diffusion, we are allowed to compare it with original data through certain metrics to detect defects. We will briefly introduce: (1) diffusion process, (2) reverse process, (3) the probability flow ODE in the followings.

Diffusion process. Diffusion process gradually adds noise to the original data $\mathbf{x}$ through forward SDE (Song et al., 2021), and yields a sequence $\{\mathbf{x}(t)\}$ through:

$$\mathrm{d}\mathbf{x}(t) = f(t)\mathbf{x}(t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}(t), \tag{1}$$

where $t \in [0, 1]$ indicates the time stamp, $\mathbf{w}(t)$ denotes a standard Wiener process, and the drift coefficient $f(t)$ and the diffusion coefficient $g(t)$ are fixed. Therefore, it is essentially an ordinary differential equation (ODE) driven by the noise. We can interpret $\mathrm{d}\mathbf{w}(t)$ as an infinitesimal Gaussian noise. The solution to this diffusion process in Eq.(1) is $\{\mathbf{x}(t)\}_{t\in[0,1]}$. Assume $p_t(\mathbf{x})$ denotes the probability density of solution and $p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))$ denotes the transition distribution from $\mathbf{x}(0)$ to $\mathbf{x}(t)$. By definition, $p_{data}(\mathbf{x}) \approx p_0(\mathbf{x})$. Based on Eq.(1), we can continuously add noises to the original data $\mathbf{x}(0) \sim p_0(\mathbf{x})$. This process gradually removes details and structure of the data as t increases, and the distribution of noisy data $p_1(\mathbf{x})$ satisfies a tractable prior distribution $\pi(\mathbf{x})$.

Reverse process. Diffusion process starts from $\mathbf{x}(0)$ and ends up with $\mathbf{x}(t)$. The reverse process aims to recover the original data from $\mathbf{x}(t)$ and get an approximation $\mathbf{x}'(0)$ generated by the reverse of a diffusion process of Eq.(1), which is also a diffusion process and can be achieved by:

$$\mathrm{d}\mathbf{x}(t) = (f(t)\mathbf{x}(t) - g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x}(t)))\mathrm{d}\bar{t} + g(t)\mathrm{d}\bar{\mathbf{w}}(t), \tag{2}$$

where $\mathrm{d}\bar{t}$ represents negative time step, $\bar{\mathbf{w}}$ represents a standard Wiener process in the reversal time direction. Therefore, the objective of score-based generative model transforms to learn the score function $\nabla_{\mathbf{x}}\log p_t(\mathbf{x}(t))$ in Eq.(2). We can estimate $\nabla_{\mathbf{x}}\log p_t(\mathbf{x})$ by training a score-based model $s_\theta(\mathbf{x}(t), t)$ on training dataset $\mathbf{X}_N$, where $s_\theta(\mathbf{x}(t), t)$ adopts a variant of U-net that requires both $\mathbf{x}(t)$ and t inputs. The objective turns to minimize the following loss (Vincent, 2011):

$$\mathcal{L}(\theta; \lambda(\cdot)) := \frac{1}{2}\int_0^1 \mathbb{E}_{p_0(\mathbf{x})p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))}[\lambda(t)||\nabla_{\mathbf{x}}\log p_{0t}(\mathbf{x}(t)|\mathbf{x}(0)) - s_\theta(\mathbf{x}(t), t)||_2^2]\mathrm{d}t, \tag{3}$$

which is equivalent up to a constant that is irrelevant to θ. Additionally, if the drift coefficient $f(t)$ is linear, the $p_{0t}(\mathbf{x}(t)|\mathbf{x}(0)) = \mathcal{N}(\mathbf{x}(t); \mu(t)\mathbf{x}(0), \sigma^2(t)\mathrm{I})$ is a tractable Gaussian distribution, and

$$\mathbf{x}(t) = \mu(t)\mathbf{x}(0) + \sigma(t)\mathbf{z}(t), \tag{4}$$

where $\mathbf{z}(t) \sim \mathcal{N}(0, \mathrm{I})$. Fortunately Variance Exploding (VE), Variance Preserving (VP) and sub-VP SDE introduced in Song et al. (2021) satisfy the linear drift coefficient condition (check more details in Appendix A.3), and therefore, $\nabla_{\mathbf{x}}\log\, p_{0t}(\mathbf{x}(t)|\mathbf{x}(0)) = -\frac{\mathbf{z}(t)}{\sigma(t)}$ of each sample can be solved. Following this, we are allowed to train a score-based model $s_\theta(\mathbf{x}(t), t)$ by sampling $\mathbf{x}(0) \sim p_0(\mathbf{x})$ from training dataset, uniformly random sampling t in [0,1], and getting $\mathbf{x}(t) \sim p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))$.

Probability flow ODE. In addition to the Eq.(2), there is an alternative solution to the reverse diffusion process termed probability flow ODE (Maoutsa et al., 2020; Song et al., 2021), abbreviated as Flow ODE. The Flow ODE shares the same marginal distribution $p_t(\mathbf{x})$ with SDE of Eq.(1) and can be defined as:

$$\mathrm{d}\mathbf{x}(t) = (f(t)\mathbf{x}(t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}}\log p_t(\mathbf{x}(t)))\mathrm{d}t. \tag{5}$$

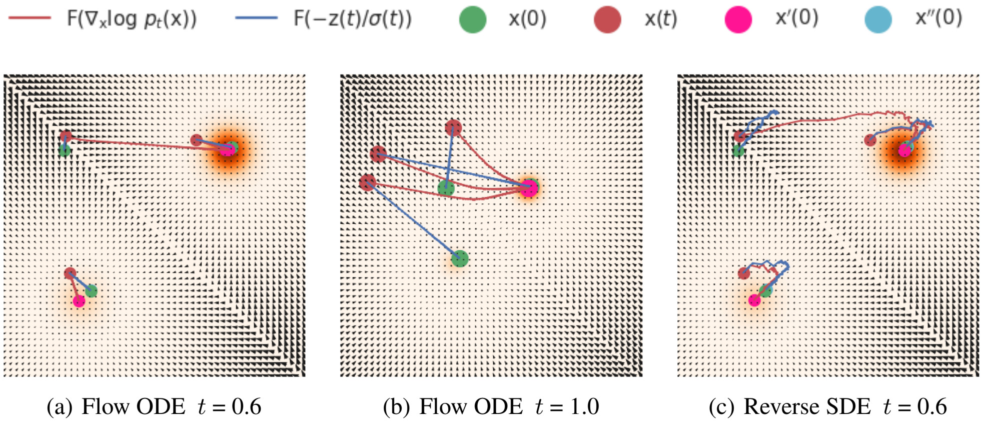
In particular, the Flow ODE does not include random terms but retains the same score function $\nabla_{\mathbf{x}}\log p_t(\mathbf{x}(t))$. Therefore, we can also plug $s_\theta(\mathbf{x}(t), t)$ into Eq.(2) or Eq.(5) to generate samples.

Whole-score and self-score. It can be seen that the output of the trained score model $s_\theta(\mathbf{x}, t) \approx \nabla_{\mathbf{x}}\log\, p_t(\mathbf{x}(t))$ is the gradient pointing to the high-density regions of $\mathbf{X}_N$, and plays a key role in the reverse process. Therefore, we term it as **whole-score** $s_w(\mathbf{x}, t)$. In addition, we use score $s_e(\mathbf{x}, t) = -\frac{\mathbf{z}(t)}{\sigma(t)}$ relative to each sample denoted as **self-score**. Both whole- and self-score will be discussed and used in our proposed Score-AD model.

3.2 Issues and Observations

We regenerate a series of images as shown in Fig. 2. In particular, given a test image, with predetermined $t \in (0, 1]$, we inject noise to $\mathbf{x}(0)$ according to the forward SDEs to achieve $\mathbf{x}(t)$. Simulating reverse process by reverse SDE or Flow ODE, we are allowed to reconstruct images $\mathbf{x}'(0)$. Note with different $t \in \{.0005, 0.1, ...\}$, details and structures are gradually removed from left to right in the third row of Fig. 2. Based on the difference between the original and reconstructed image, we can localize the defects. For example, if using the reconstruction of $t = 0.5$ in the first row of Fig. 2, we may easily locate the defects. However, picking appropriate t is non-trivial. When using smaller t, not all defect images can be well changed into the normal mode. On the other hand, when using larger t values, the normal pixel in the reconstructed image is still somewhat different from the original image in pixel space.

To provide more insights behind these phenomena, we conduct another experiment in Fig. 3 to explore how the score-based model transforms anomalous data to normal pattern, and reasons of deterioration in normal regions of the original image. For demonstration purposes, we consider VE SDE where $\mathrm{d}\mathbf{x}(t) = \sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}}\mathrm{d}\mathbf{w}(t)$. Assume that the distribution of positive data is $\frac{1}{5}\mathcal{N}((-5, -5), \mathrm{I}) + \frac{4}{5}\mathcal{N}((5, 5), \mathrm{I})$, and set 100 time steps in [0,1]. Other details are presented in Appendix B. Following the steps discussed above, we set the initial diffusion time step t and obtain $\mathbf{x}(t)$ by Eq.(4), and then conduct reverse process through Flow ODE in Eq.(5) or reverse SDE in Eq.(2). In Fig. 3, anomalous



(a) Flow ODE $t = 0.6$ (b) Flow ODE $t = 1.0$ (c) Reverse SDE $t = 0.6$

Figure 3: Exploratory experiment based on VE SDE. $p_{data}(\mathbf{x})$ is shown in an orange colormap. The red trajectory represents the reverse diffusion from $\mathbf{x}(t)$ to $\mathbf{x}'(0)$ driven by $\nabla_{\mathbf{x}}\log p_t(\mathbf{x}(t))$, and the blue trajectory is the process from $\mathbf{x}(t)$ to $\mathbf{x}''(0)$ driven by $-\mathbf{z}(t)/\sigma(t)$. (a) $t = 0.6$ and iterate the Flow ODE. (b) $t = 1.0$ and iterate the Flow ODE. (c) $t = 0.6$ and iterate the Reverse SDE.

data points originally located in low probability density move to high probability density regions driven by whole-score $s_w(\mathbf{x}, t)$ and end up as $\mathbf{x}'(0)$. For normal data, as shown in Fig. 3(a) and Fig. 3(c), with a suitable t, e.g., $t = 0.6$, the deviation of the $\mathbf{x}'(0)$ from $\mathbf{x}(0)$ is smaller. When $t = 1.0$, some normal data has trouble returning to the vicinity of $\mathbf{x}(0)$, as shown in Fig. 3(b). The primary reason is $s_w(\mathbf{x}, t)$ is learned to enforces the $\mathbf{x}'(0)$ moving to towards the high probability density region of training data. When $t = 1.0$, mixed Gaussians are fused into a tractable Gaussian distribution $p_1(\mathbf{x}) \approx \pi(\mathbf{x})$. Therefore, some normal points are driven into the other Gaussian cores, which deviate significantly from $\mathbf{x}(0)$.

4 Proposed Method

4.1 Leveraging Whole-scores to Localize Defects

One of the major issues identified in Fig. 3 is the whole-score drive the noisy data towards the high density region, which makes anomalous data, originally located in the low density region, eventually fall into the high density region. In the meanwhile, we shall carefully select t value to retain the overall contour of the distribution, such that the reconstructed normal data remain in the vicinity of the original data after iterations. Otherwise, the normal region of the original data will be changed significantly, as shown in Fig. 2 when $t > 0.5$. This stringent requirement on t makes comparing the difference between the reconstructed and original image in pixel space be a less feasible or reliable solution. However, we find that normal and abnormal data behave differently in the dimension of probability density. From Fig. 3, after sufficient iterations, all normal or abnormal data eventually fall into the high density portion of the normal distribution. Based on assumption that original anomaly data is in the low probability density zone and normal data is originally in the high density region, we propose that employing a metric connected to the probability density of normal data, e.g., whole-score $s_w(\cdot, t) \approx \nabla_{\mathbf{x}}\log p_t(\mathbf{x}(t))$, is effective for AD. As a result, we can feed $\mathbf{x}'(0)$ and $\mathbf{x}(0)$ into score model to assess their whole-score difference.

4.2 Enhancement through Feature Maps

As the score-based model is usually implemented through neural networks, the characteristics of scores can also be reflected by feature maps. Feature maps in deep layers containing more semantic information and shallow layers feature maps for identifying fine-grained information such as lines, colors, and so on. Previous works have already investigated the usefulness of feature maps in different network layers for unsupervised anomaly detection (Wang et al., 2021; Yamada & Hotta, 2021; Yang et al., 2020) as well as semantic segmentation (Baranchuk et al., 2022) through the middle

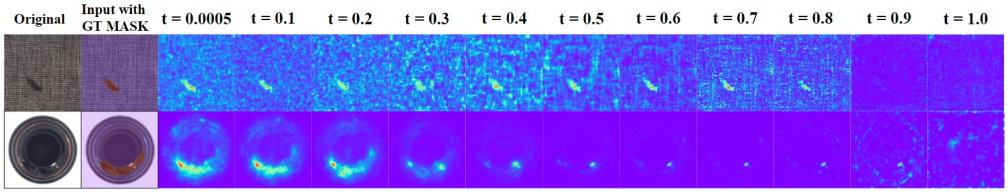


Figure 4: Different semantic information of score model at different moments. Set different initial time t, iterating $r = 1$ steps with the Flow ODE, and present upsampling 64×64 feature map of score model.

layer of score-based model. Therefore, we also adopt the multi-scale feature maps to strengthen the performance through a U-net architecture as the followings.

In order to avoid using more models, we decide to adopt a similar scheme to process feature maps in Yamada & Hotta (2021). Firstly, calculate the Euclidean distance between the two feature maps after performing l_2 normalization on each feature map. Then, feature maps of the same resolution are summed, and all feature maps are scaled up to the same resolution by using the "bilinear" interpolations. The products of all feature maps are taken as the output. In our practice, however, we found that the l_2 normalization compromises the feature maps efficacy and its visual effects. We believe the reason is feature maps of score-based model are not particularly trained for AD in an supervised manner. Without normalization, feature maps of the same resolution may have different magnitude. When they are added together, the output will favor the feature maps of greater magnitude, leading to poor results. Therefore, we skip both l_2 normalization and sum of feature maps of the same resolution. Instead, feature maps with significant visual effects are selected and their point-wise product will be used as outputs, as shown in Fig. 5.

4.3 T Scales

Another issue identified in practice is the time spent in the reverse process given a large t. It takes many iterations with score model to return to $\mathbf{x}'(0)$. We are considering whether we can leverage the feature maps of score model at different moments, but without the full iteration, because goal is to detect defects rather than generate images.

It has been discussed that the score-based model provides semantics at different moments t (Baranchuk et al., 2022), as Fig. 4 shown. Because the feature maps are changing gradually, there may be a lot of redundant information in the feature maps at adjacent moments. Therefore, we are motivated to not perform a full iteration to get the final image, but to iterate r steps (r is a very small integer), then compute the difference of $s_w(\cdot, t_r)$ between $\mathbf{x}'(t_r)$ and $\mathbf{x}''(t_r)$ to be the representative semantic information in a certain time period around t, where $\mathbf{x}''(t)$ represents the true trajectory from $\mathbf{x}(t)$ to $\mathbf{x}(0)$. In order to leverage different information at different moments, we can apply a set of different moments $\{t\}$ of capacity T, and in each t case, we do the same process above. We term this approach as T **scales**, as it will yield T feature maps to be assembled for anomaly map, as shown in Fig. 5.

Assume at step t, we will examine $\mathbf{x}$ at $t_1 > ... > t_i > ... > t_r$ in a sequential manner, where $t_i - t_{i+1} = \Delta t$ will be used as the approximation of $\mathrm{d}t$ in ODE. Without loss of generality, we elaborate the process of computing $\mathbf{x}'(t_i)$ and $\mathbf{x}''(t_i)$ in each t case as follows. First, $\mathbf{x}'(t_i)$ can be achieved by replacing $\nabla_{\mathbf{x}} \log p_t(\mathbf{x}(t))$ in Eq.(2) or Eq.(5) with whole-score $s_w(\cdot, t)$ to build the reverse path as the red path in Fig. 3. Second, $\mathbf{x}''(t_i)$ in adjacent steps can be modeled by replacing the $\nabla_{\mathbf{x}} \log p_t(\mathbf{x}(t))$ in Eq.(2) or Eq.(5) with self-score $s_e(\cdot, t)$ as the true trajectory from $\mathbf{x}(t)$ to $\mathbf{x}(0)$. (The proof can be found in Appendix A.1). Therefore, if we know self-score $s_e(\cdot, t)$ relative to $\mathbf{x}(0)$ at each moment, we can approach the original $\mathbf{x}(0)$ from $\mathbf{x}(t)$, as the blue path in Fig. 3. We show the overall framework of our algorithm in Fig. 5. Specific calculation process and pseudo-code are given in Appendix A.2. The key intuition of Score-AD with T scales as illustrated in Fig. 1.

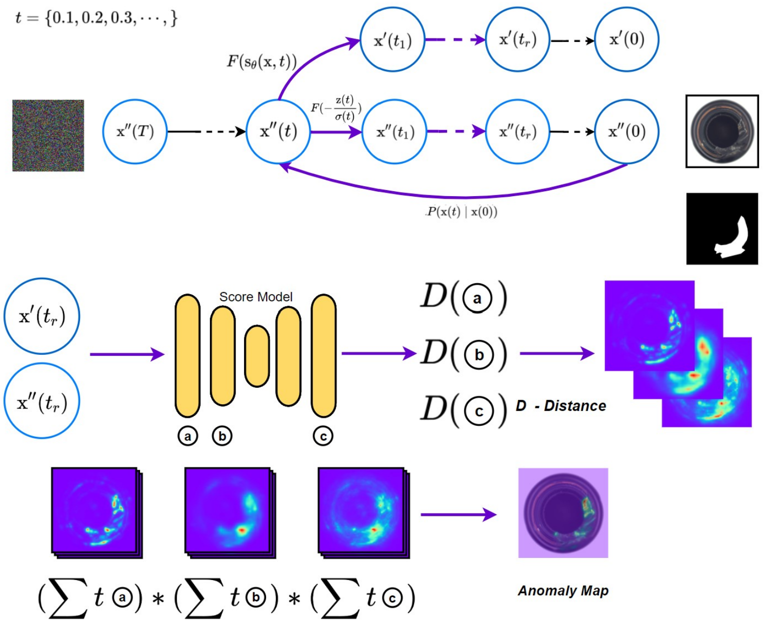


Figure 5: An overview of **Score-AD** for anomaly detection. Set $\{t\}$ and in each moment t case, first inject noise to a test image through $p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))$ Eq.(4). Then solving Eq.(2) or Eq.(5) by plugging $s_w(\mathbf{x}, t)$ and $s_e(\mathbf{x}, t)$ separately into them. After iterating r steps, input two samples into score model to extract feature maps. Add up all feature maps of the same resolution at different $\{t\}$, then after upsampling, the final Anomaly map is obtained by multiplying them up.

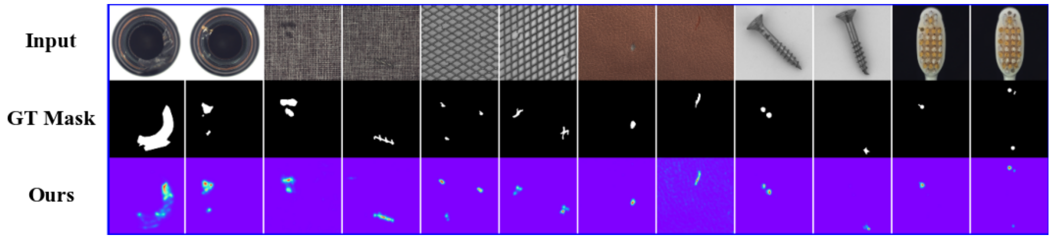


Figure 6: Anomaly location examples of Score-AD on MVTec. The first row is the original anomaly images. The second row is GT MASK. The third row is anomaly map from Score-AD.

5 Experiment

5.1 Datasets

We conducted tests on common benchmarks to validate the effectiveness of our proposed approach, **Score-AD**. We describe in detail the data sets used. **MVTec AD dataset** (Bergmann et al., 2019) contains 5354 high-resolution images, which is specifically utilized in the unsupervised AD task. It contains 10 objects and 5 texture categories, and each category contains 60-320 train samples and about 100 test samples. **BeanTech AD dataset** (Mishra et al., 2021) is a industrial dataset containing 2540 high-resolution images of three products. **MNIST** (LeCun et al., 2010) contains 60k training and 10k test 28 × 28 gray-scale handwritten digit images.

5.2 Experiment setup

All of the images in the aforementioned dataset are resized to 256 × 256 pixels, except for MNIST which is resized to 32 × 32 size. We train a score-based model based on NCSN++ and set 2000 diffusion timesteps for 256 × 256 size and train a score model based on DDPM++ (deep,VP) for

Table 1: Anomaly detection (**left**) and localization (**right**) performance on MVTec AD dataset. Methods achieved for the top two AUROC (%) are highlighted in bold. * means the method based on pre-trained model. ^{i} means Score-AD based on VE SDE, ^{ii} means Score-AD based on VP SDE, ^{iii} means Score-AD based on sub-VP SDE.

Method	RIAD	OCR-GAN	CFlow*	FastFlow*	PaDiM*	PatchCore*	CutPaste	DRÆM	Ours^{i}	Ours^{ii}	Ours^{iii}
carpet	84.2/96.3	99.4/-	**100/99.3**	**100**/99.4	-/99.1	98.7/98.9	93.9/98.3	97.0/95.5	96.9/98.9	86.0/95.5	93.0/98.0
grid	99.6/98.8	99.6/-	97.6/99.0	99.7/98.3	-/97.3	98.2/98.7	**100**/97.5	99.9/**99.7**	**100/99.7**	100/99.6	100/99.5
leather	**100**/99.4	97.1/-	97.7/**99.7**	**100/99.5**	-/99.2	**100**/99.3	**100/99.5**	**100**/98.6	99.6/99.3	97.8/97.9	98.3/99.0
tile	98.7/89.1	95.5/-	**98.7/98.0**	**100**/96.3	-/94.1	98.7/95.6	94.6/90.5	**99.6/99.2**	98.6/94.4	98.7/95.7	100/94.8
wood	93.0/85.8	95.7/-	**99.6/96.7**	**100/97.0**	-/94.9	99.2/95.0	99.1/95.5	99.1/**96.4**	98.8/95.1	98.9/96.9	96.2/95.5
bottle	99.9/98.4	99.6/-	**100/99.0**	**100**/97.7	-/98.3	**100**/98.6	98.2/97.6	99.2/**99.1**	**100**/97.9	**100**/98.1	99.7/95.7
cable	81.9/84.2	99.1/-	**100**/97.6	**100/98.4**	-/96.7	99.5/**98.4**	81.2/90.0	91.8/94.7	96.8/97.5	95.7/95.1	98.0/97.9
capsule	88.4/92.8	96.2/-	**99.3/99.0**	**100/99.1**	-/98.5	98.1/98.8	98.2/97.4	98.5/94.3	96.1/98.6	91.5/97.4	93.2/97.7
hazelnut	83.3/96.1	98.5/-	96.8/98.9	**100**/99.1	-/98.2	**100**/98.7	98.3/97.3	**100/99.7**	99.9/99.2	98.4/**99.4**	97.6/99.0
metal Nut	88.5/92.5	99.5/-	91.9/**98.6**	**100**/98.5	-/97.2	**100**/98.4	99.9/93.1	98.7/**99.5**	97.2/97.9	98.9/98.0	99.1/94.7
pill	83.8/95.7	98.3/-	**99.9/99.0**	**99.4/99.2**	-/95.7	96.7/97.1	94.9/95.7	98.9/97.6	95.3/96.0	88.3/96.4	93.7/94.4
screw	84.5/98.8	**100**/-	**99.7/98.9**	97.8/99.4	-/98.5	98.1/99.4	88.7/96.7	93.9/97.6	99.6/**99.6**	98.3/99.8	99.1/99.8
toothbrush	**100/98.9**	98.7/-	95.2/**99.0**	94.4/**98.9**	-/98.8	**100**/98.7	99.4/98.1	**100**/98.1	99.8/98.3	98.3/97.8	98.9/97.9
transistor	90.9/87.7	98.3/-	99.1/**98.0**	**99.8/97.3**	-/**97.5**	**100**/96.3	96.1/93.0	93.1/90.9	95.4/95.2	95.4/94.2	96.4/94.7
zipper	98.1/97.8	99.0/-	98.5/99.1	99.5/98.7	-/98.5	98.8/98.8	**99.9/99.3**	**100**/98.8	99.8/**99.3**	**99.9/99.3**	**99.9**/99.2
Average	91.7/94.2	98.3/-	98.3/**98.6**	**99.4/98.5**	97.9/97.5	**99.1**/98.1	96.1/96.0	98.0/97.3	98.2/**97.8**	96.4/97.4	97.5/97.2

Table 2: Anomaly detection (**left**) and localization (**right**) performance on BTAD dataset.

Class	Panda*	PaDiM*	FastFlow*	VT-ADL	OURS
1	96.4/96.4	99.4/97.2	-/95	-/99	99.2/97.7
2	81.0/94.1	79.5/95.2	-/96	-/94	81.1/95.2
3	69.8/98.0	99.4/98.7	-/99	-/77	99.1/98.3
Mean	82.4/96.2	92.7/97.0	-/97	-/79	93.1/97.1

Table 4: Quantitative results of Score-AD for ablation studies on MVTec AD dataset.

Case	Reconstruction Loss	Score-AD (w/o T scales)	Score-AD (w/ T scales)
AUROC	84.85 /89.34	91.30 /96.24	98.24/97.78

Table 3: Quantitative results of AUROC for Anomaly Detection on MNIST dataset.

Method	AUROC
ARAE	97.5
OCSVM	96.0
AnoGAN	91.4
DSVDD	94.8
OCGAN	97.5
LSA	97.5
U-Std*	99.35
MKDAD*	98.71
Score-AD	95.44

MNIST. We take area under the receiver operating characteristic curve (AUROC) the evaluation metric for both anomaly detection and localization. In the inference stage, after we get the anomaly map, leverage it to evaluate the AUROC metric for location task and the maximum value of each anomaly map to evaluate the AUROC metric for classification task. Other experiment details are presented in Appendix B.

5.3 State-of-the-Art Comparison

Results on MVTec AD are shown in Table 1, where **Score-AD** based on VE SDE and Flow ODE sampling method achieves the SOTA **98.2 image-level AUC** and **97.8 pixel-level AUC**, Score-AD based on VP SDE and reverse SDE sampling method achieves 96.4 image-level AUC and 97.4 pixel-level AUC, and Score-AD based on sub-VP SDE and the Flow ODE sampling method achieves 97.5 image-level AUC and 97.2 pixel-level AUC. We compare our results with the SOTA unsupervised AD methods on MVTec AD dataset. Specifically, Score-AD outperforms the AE-based method, RIAD (Zavrtanik et al., 2021b), and is only $0.1\%\downarrow$ than the GAN combined with pseduo-defects method, OCR-GAN (Liang et al., 2022). Although our approach still lags behind CFLOW (Gudovskiy et al., 2021) and FastFlow (Yu1 et al., 2022), works combined Flow with pre-trained models, as well as some others based on pre-trained models, likely PatchCore (Roth et al., 2022), Score-AD does not resort to pre-trained models that contain rich semantic information and have some comparability in some classes. For the methods of creating pseudo-defects to transform unsupervised learning into supervised learning, Score-AD also outperform CutPaste (Li et al., 2021), while DRÆM (Zavrtanik et al., 2021a), which uses additional data to create defects and specifically designed reconstruction model and anomaly segmentation model for AD, achieved SOTA results, Score-AD also outperforms it by $0.2\%\uparrow$ in the detection task and $0.5\%\uparrow$ in the localization task.

Results on BeanTech AD are shown in Table 2, where Score-AD based on VP SDE and the probabiliyu flow ODE achieves **93.4 image-level AUC** and **97.1 pixel-level AUC**. Ours outperforms the reconstruction-based method depended on transformation model, VT-ADL (Mishra et al., 2021). And compared with the methods relied on pre-trained models, including Panda (Reiss et al., 2021a), PaDiM (Defard et al., 2021) and FastFlow (Yu1 et al., 2022), we achieve a new SOTA results.

Results on MNIST are displayed in Table 3. Score-AD achieves **95.44 image-level AUC**. Compared with the methods relied on pre-trained models, U-Std (Bergmann et al., 2020) and MKDAD (Salehi et al., 2021b), we are about $3.91\% \downarrow$ below the best result. More fairly, compared with unsupervised methods, including ARAE (Salehi et al., 2021a), OCSVM (Chen et al., 2001), AnoGAN (Li et al., 2018), DSVDD (Ruff et al., 2018), OCGAN (Perera et al., 2019) and LSA (Abati et al., 2019). We come in $2.1\% \downarrow$ below the top score. Notably, our approach is sensitive to feature maps that are fixedly selected and cannot be adaptively adjusted according to different image. Thus difficulties are encountered in experimentally tuning a large dataset like MNIST. This can be improved by easily extending a professional and generalized classification or segmentation network with self-supervised (e.g., pseduo-defects) or semi-supervised (i.e., several available abnormal samples with image or pixel labels) methods. Since the purpose of this paper is to explore unsupervised solutions that are applicable to the score model and do not depend on other models, we leave this work for the future.

5.4 Ablation study and analysis

We do comparative experiments to confirm the efficacy of submodules methods. Specifically, we run a continuous version experiment (Score-AD without T scales) iterating from the maximum moment in $\{t\}$ to near the smallest moment ϵ, and then take difference of the $s_w(\cdot, \epsilon)$ and its feature map to simulate score difference, $||\nabla_{\mathbf{x}'} \log p_t(\mathbf{x}'(\epsilon)) - \nabla_{\mathbf{x}''} \log p_t(\mathbf{x}''(\epsilon))||^2$, and also do a experiment based on reconstruction loss between $\mathbf{x}'(0)$ and $\mathbf{x}''(0)$ for AD, $||\mathbf{x}'(0) - \mathbf{x}''(0)||^2$. From the Table 4, compared the case with reconstruction loss to Score-AD without T scales, it can be easily proved that the score as a metric is more useful for AD. Plus, compared the case with T scales to the case without T scales, T scales technique can enhance performance. This is because it iterates score model just few steps, some normal pixels have little opportunity accessing other high probability density regions, alleviating the problems discussed above in Section 3.2 further, and it also leverages the different semantic information at different moments.

5.5 Computational complexity

We propose T scales technique to speed up the inference process. Take an example of real case in MVTec AD dataset, after we train a score model that needs $S = 2000$ iteration steps to generate images, and we set inital timesteps as $t = 250/2000$. Therefore, the inference-time efficiency of Score-AD without T scales is $O(t * S - 1 + 2) = O(251)$. However, with T scales, we develop a set containing different timesteps $\{t\}$. For example, we can take a t from 250 to 50 every 50 steps, $\{t\} = \{250, 200, 150, 100, 50\}/2000$ containing $T = 5$ different timesteps. Therefore, the minimal inference-time efficiency is $O(T * (r + 2)) = O(T * 3) = O(15)$, which are more efficient than AnoDDPM (Wyatt et al., 2022) whose inference-time efficiency is $O(t * S) = O(250)$. Our calculated inference-time efficiency is based on the number of times the neural network is run. What's more, T scales technique also makes the sequential iteration split into parallel cases, and each t case needs to iterate just few r steps. Therefore, it can also run in parallel to speed up further. Moreover, the community is also exploring some ways to accelerate the score model and diffusion model.

6 Conclusion

We propose to use score-based generative model for unsupervised anomaly detection. Our research indicates that employing a metric for AD that is linked to the probability density of normal data, e.g., score value, can efficiently handle the challenge of reconstructed normal images that differ from the original normal images in pixel space. In addition, we propose to use T scales to solve the problem of slow speed due to the need of iterating multiple steps of Markov chain, and since we only need to iterate few steps at each t moment, it does not deviate the normal pixels too much from the original data, but in turn improves the accuracy. Without using additional data, algorithms and models, we achieve a competitive performance on several datasets.

REFERENCES

Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Latent space autoregression for novelty detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Asian conference on computer vision*, pp. 622–637. Springer, 2018.

Samet Akçay, Amir Atapour-Abarghouei, and Toby P Breckon. Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2019.

Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. 2022.

Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad — a comprehensive real-world dataset for unsupervised anomaly detection. *computer vision and pattern recognition*, 2019.

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)*, June 2020.

Yunqiang Chen, Xiang Sean Zhou, and T.S. Huang. One-class svm for learning in image retrieval. In *Proceedings 2001 International Conference on Image Processing (Cat. No.01CH37205)*, volume 1, pp. 34–37 vol.1, 2001. doi: 10.1109/ICIP.2001.958946.

Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. In *International Conference on Pattern Recognition*, pp. 475–489. Springer, 2021.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1705–1714, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Denis A. Gudovskiy, Shun Ishizaka, and Kazuki Kozuka. Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. *arXiv: Computer Vision and Pattern Recognition*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *neural information processing systems*, 2020a.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020b. URL https://proceedings.neurips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Jinlei Hou, Yingying Zhang, Qiaoyong Zhong, Di Xie, Shiliang Pu, and Hong Zhou. Divide-and-assemble: Learning block-wise memory for unsupervised anomaly detection. *arXiv: Computer Vision and Pattern Recognition*, 2021.

Chaoqin Huang, Fei Ye, Jinkun Cao, Maosen Li, Ya Zhang, and Cewu Lu. Attribute restoration framework for anomaly detection. *IEEE Transactions on Multimedia*, 2019.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Yann LeCun, Corinna Cortes, and CJ Curges. Mnist handwritten digit database. 2010.

Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. *arXiv: Computer Vision and Pattern Recognition*, 2021.

Dan Li, Dacheng Chen, Jonathan Goh, and See-kiong Ng. Anomaly Detection with Generative Adversarial Networks for Multivariate Time Series. *arXiv e-prints*, art. arXiv:1809.04758, September 2018.

Yufei Liang, Jiangning Zhang, Shiwei Zhao, Runze Wu, Yong Liu, and Shuwen Pan. Omni-frequency channel-selection representations for unsupervised anomaly detection. 2022.

Dimitra Maoutsa, Sebastian Reich, and Manfred Opper. Interacting particle solutions of fokker-planck equations through gradient-log-density estimation. *Entropy*, 2020.

Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Image synthesis and editing with stochastic differential equations. *arXiv: Computer Vision and Pattern Recognition*, 2021.

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, 2021. doi: 10.1109/ISIE45552.2021.9576231.

Cheng Ouyang, Carlo Biffi, Chen Chen, Turkay Kart, Huaqi Qiu, and Daniel Rueckert. Self-supervision with superpixels: Training few-shot medical image segmentation without annotation. In *European Conference on Computer Vision*, pp. 762–780. Springer, 2020.

Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. Ocgan: One-class novelty detection using gans with constrained latent representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

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, June 2021a.

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*, pp. 2806–2814, 2021b.

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, June 2022.

Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. Same same but differnet: Semi-supervised defect detection with normalizing flows. *arXiv: Computer Vision and Pattern Recognition*, 2020.

Marco Rudolph, Tom Wehrbein, Bodo Rosenhahn, and Bastian Wandt. Fully convolutional cross-scale-flows for image-based defect detection. 2022.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4393–4402. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/ruff18a.html`.

Mohammadreza Salehi, Atrin Arya, Barbod Pajoum, Mohammad Otoofi, Amirreza Shaeiri, Mohammad Hossein Rohban, and Hamid R. Rabiee. Arae: Adversarially robust training of autoencoders improves novelty detection. *Neural Networks*, 144:726–736, 2021a. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2021.09.014. URL `https://www.sciencedirect.com/science/article/pii/S0893608021003646`.

Mohammadreza Salehi, Niousha Sadjadi, Soroosh Baselizadeh, Mohammad H. Rohban, and Hamid R. Rabiee. Multiresolution knowledge distillation for anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14902–14912, June 2021b.

Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pp. 146–157. Springer, 2017.

Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, 54:30–44, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=PxTIG12RRHS`.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7537–7547. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf`.

Hao Tang, Xingwei Liu, Shanlin Sun, Xiangyi Yan, and Xiaohui Xie. Recurrent mask refinement for few-shot medical image segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3918–3928, 2021.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 2011.

Guodong Wang, Shumin Han, Errui Ding, and Di Huang. Student-teacher feature pyramid matching for anomaly detection. *arXiv preprint arXiv:2103.04257*, 2021.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Julian Wyatt, Adam Leach, Sebastian M. Schmon, and Chris G. Willcocks. Anoddpm: Anomaly detection with denoising diffusion probabilistic models using simplex noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 650–656, June 2022.

Shinji Yamada and Kazuhiro Hotta. Reconstruction student with attention for student-teacher pyramid matching. *arXiv preprint arXiv:2111.15376*, 2021.

Jie Yang, Yong Shi, and Zhiquan Qi. Dfr: Deep feature reconstruction for unsupervised anomaly segmentation. *arXiv: Computer Vision and Pattern Recognition*, 2020.

Jongmin Yoon, Sung Ju Hwang, and Juho Lee. Adversarial purification with score-based generative models. *arXiv: Learning*, 2021.

Jiawei Yu1, Ye Zheng, Xiang Wang, Wei Li, Yushuang Wu, Rui Zhao, and Liwei Wu1. Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows. 2022.

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, October 2021a.

Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition*, 112:107706, 2021b. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2020.107706. URL `https://www.sciencedirect.com/science/article/pii/S0031320320305094`.

A Supplementary formula

A.1 Proof for the true trajectory

First, we define the true trajectory from $\mathbf{x}(t)$ to $\mathbf{x}(0)$ in the sense that after obtaining $\mathbf{x}(t)$ by injecting noise to $\mathbf{x}(0)$, the path is iterated back to the original $\mathbf{x}(0)$ through reverse stochastic process. Below, we give the proof of the formulation about the true trajectory from $\mathbf{x}(t)$ to $\mathbf{x}(0)$.

First, the objective of generative model is to generate samples that satisfy the distribution of the given training data. Recalling Section 3.1, we train a neural network to fit the score function $\nabla_{\mathbf{x}}\log p_t(\mathbf{x}(t))$ of given dataset $\mathbf{X}_N$, which guarantees the score-based generative model eventually generate $\mathbf{x}(0) \sim p_0(\mathbf{x})$ through reverse process, where $p_0(\mathbf{x}) \approx p_{data}(\mathbf{x})$ by definition. Here,

$$\begin{aligned}\frac{\partial \log p_t(\mathbf{x}(t))}{\partial \mathbf{x}(t)} &= \frac{1}{p_t(\mathbf{x}(t))}\frac{\partial p_t(\mathbf{x}(t))}{\partial \mathbf{x}(t)} \\ &= \frac{1}{p_t(\mathbf{x}(t))}\frac{\partial}{\partial \mathbf{x}(t)}\int p_0(\mathbf{x}(0))p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))\mathrm{d}\mathbf{x}(0) \\ &= \frac{1}{p_t(\mathbf{x}(t))}\int p_0(\mathbf{x}(0))\frac{\partial p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))}{\partial \mathbf{x}(t)}\mathrm{d}\mathbf{x}(0) \\ &= \frac{1}{p_t(\mathbf{x}(t))}\int p_0(\mathbf{x}(0))p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))\frac{\partial \log p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))}{\partial \mathbf{x}(t)}\mathrm{d}\mathbf{x}(0) \\ &= \int \frac{p_0(\mathbf{x}(0))p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))}{p_t(\mathbf{x}(t))}\frac{\partial \log p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))}{\partial \mathbf{x}(t)}\mathrm{d}\mathbf{x}(0). \end{aligned} \tag{6}$$

However, regarding the true trajectory from $\mathbf{x}(t)$ to $\mathbf{x}(0)$, we can consider that $p_{data}(\mathbf{x})$ degenerates to a one-point distribution with mean $\mathbf{x}(0)$ and variance 0, denoted as $p''_{data}(\mathbf{x})$:

$$p''_{data}(\mathbf{x}) = \begin{cases} 1, & \mathbf{x} = \mathbf{x}(0) \\ 0, & \text{Others} \end{cases} \tag{7}$$

Therefore, $\nabla_{\mathbf{x}}\log p''_t(\mathbf{x}(t)) = \nabla_{\mathbf{x}}\log p''_{0t}(\mathbf{x}(t)|\mathbf{x}(0))$. As discussed in Section 3.1, if drift coefficient $f(t)$ of SDE is linear, the transition density is Gaussian $p_{0t}(\mathbf{x}(t)|\mathbf{x}(0)) = \mathcal{N}(\mathbf{x}(t); \mu(t)\mathbf{x}(0), \sigma(t)^2\mathrm{I})$. Thus, $\nabla_{\mathbf{x}}\log p''_t(\mathbf{x}(t)) = \nabla_{\mathbf{x}}\log p''_{0t}(\mathbf{x}(t)|\mathbf{x}(0)) = -\frac{\mathbf{x}(t)-\mu(t)\mathbf{x}(0)}{\sigma(t)^2} = -\frac{\mathbf{z}(t)}{\sigma(t)}$, where $\mathbf{x}(t) = \mu(t)\mathbf{x}(0) + \sigma(t)\mathbf{z}(t)$, $\mathbf{z}(t) \sim \mathcal{N}(0, \mathrm{I})$, denoted as self-score $s_e(\cdot, t)$ in our paper. Therefore, we can obtain Eq.(8) and Eq.(9) by plugging $\nabla_{\mathbf{x}}\log p''_t(\mathbf{x}(t)) = -\frac{\mathbf{z}(t)}{\sigma(t)}$ into Eq.(2) and Eq.(5) respectively.

$$\mathrm{d}\mathbf{x}(t) = (f(t)\mathbf{x}(t) - g(t)^2(-\frac{\mathbf{z}(t)}{\sigma(t)}))\mathrm{d}\bar{t} + g(t)\mathrm{d}\bar{\mathbf{w}}(t). \tag{8}$$

$$\mathrm{d}\mathbf{x}(t) = (f(t)\mathbf{x}(t) - \frac{1}{2}g(t)^2(-\frac{\mathbf{z}(t)}{\sigma(t)}))\mathrm{d}t. \tag{9}$$

The path obtained by iterating Eq.(8) is represented as the true trajectory from $\mathbf{x}(t)$ to $\mathbf{x}(0)$ with the Reverse SDE in Eq.(2), and the path from Eq.(9) as the true trajectory with the probability flow ODE in Eq.(5). In addition, when training the score-based model, whole-score is actually evaluated through self-score $-\frac{\mathbf{z}(t)}{\sigma(t)}$ of each sample in the training dataset. And we can provide more insights of the principle of **score-AD** by analyzing the difference of whole-score and self-score in calculation ways with Eq.(6).

A.2 Algorithm

From Eq.(8) and Eq.(9), we can approach the original $\mathbf{x}(0)$ from $\mathbf{x}(t)$ if we know $\mathbf{z}(t)$ relative to $\mathbf{x}(0)$ at each moment. It should be noted that the $\mathbf{z}(t_i)$ changes constantly over steps, and by deforming

Eq.(4), $\mathbf{z}(t) = \frac{\mathbf{x}(t)-\mu(t)\mathbf{x}(0)}{\sigma(t)}$, we can know it will be updated along with $\mathbf{x}''(t_i)$. At the current step, as the $\mathbf{z}(t_i)$ is known in advance, we can bring self-score computed by $\mathbf{z}(t_i)$ to Eq.(8) or Eq.(9) to obtain $\mathbf{x}''(t_{i+1})$. Through above deformation of Eq.(4) and $\mathbf{x}''(t_{i+1})$, we can therefore obtain $\mathbf{z}(t_{i+1})$. Repeating this process, we can obtain the complete trajectory of $\mathbf{x}(t)$ to $\mathbf{x}''(t_r)$, or ultimately to $\mathbf{x}''(0) \approx \mathbf{x}(0)$. Therefore, after r steps, we can feed the sets $\{x'(t_r)\}$ and $\{x''(t_r)\}$ (each of which has a capacity of T) into the score-based model. We conclude the algorithm about **Score-AD**. Algorithm 1 and 2 denotes the reverse diffusion process with the probability flow ODE, and Reverse SDE separately. $\mathbf{x}(0) \in \mathbf{X}_{N+A}$ is a test image, $\{t\}$ is a set of different initial time with capacity of T, and r is the number of iteration steps.

Algorithm 1 Score-AD with the flow ODE

```
Require: $\mathbf{x}(0)$;$\{t\}$; $r$;
for $t \in \{t\}$ do
    $\mathbf{x}(t) = \mu(t)\mathbf{x}(0)$
    $\mathbf{z}(t) \sim \mathcal{N}(0, I)$
    $\mathbf{x}(t_0) = \mathbf{x}(t) + \sigma(t)\mathbf{z}(t)$
    for $i = 0$ to $r - 1$ do
        $\Delta t = t_i - t_{i+1}$
        $\mathbf{x}'(t_{i+1}) = \mathbf{x}'(t_i) - f(t_i)\mathbf{x}'(t_i)\Delta t$
        $\mathbf{x}'(t_{i+1}) = \mathbf{x}'(t_{i+1}) + \frac{1}{2}g(t_i)^2 s_\theta(\mathbf{x}', t_i)\Delta t$
        $\mathbf{x}''(t_{i+1}) = \mathbf{x}''(t_i) - f(t_i)\mathbf{x}''(t_i)\Delta t$
        $\mathbf{x}''(t_{i+1}) = \mathbf{x}''(t_{i+1}) + \frac{1}{2}g(t_i)^2(-\frac{\mathbf{z}(t_i)}{\sigma(t_i)})\Delta t$
        $\mathbf{z}(t_{i+1}) = \frac{\mathbf{x}''(t_{i+1}) - \mu(t_{i+1})\mathbf{x}(0)}{\sigma(t_{i+1})}$
    end for
    Input $\mathbf{x}'(t_r)$ and $\mathbf{x}''(t_r)$ to the score model
end for
Add or multiply feature maps
return Anomaly Map
```

Algorithm 2 Score-AD with Reverse SDE

```
Require: $\mathbf{x}(0)$;$\{t\}$; $r$;
for $t \in \{t\}$ do
    $\mathbf{x}(t) = \mu(t)\mathbf{x}(0)$
    $\mathbf{z}(t) \sim \mathcal{N}(0, I)$
    $\mathbf{x}(t_0) = \mathbf{x}(t) + \sigma(t)\mathbf{z}(t)$
    for $i = 0$ to $r - 1$ do
        $\Delta t = t_i - t_{i+1}$
        $\mathbf{n}(t_i) \sim \mathcal{N}(0, I)$
        $\mathbf{x}'(t_{i+1}) = \mathbf{x}'(t_i) - f(t_i)\mathbf{x}'(t_i)\Delta t$
        $\mathbf{x}'(t_{i+1}) = \mathbf{x}'(t_{i+1}) + g(t_i)^2 s_\theta(\mathbf{x}', t_i)\Delta t$
        $\mathbf{x}'(t_{i+1}) = \mathbf{x}'(t_{i+1}) + g(t_i)\sqrt{\Delta t}\mathbf{n}(t_i)$
        $\mathbf{x}''(t_{i+1}) = \mathbf{x}''(t_i) - f(t_i)\mathbf{x}''(t_i)\Delta t$
        $\mathbf{x}''(t_{i+1}) = \mathbf{x}''(t_{i+1}) + g(t_i)^2(-\frac{\mathbf{z}(t_i)}{\sigma(t_i)})\Delta t$
        $\mathbf{x}''(t_{i+1}) = \mathbf{x}''(t_{i+1}) + g(t_i)\sqrt{\Delta t}\mathbf{n}(t_i)$
        $\mathbf{z}(t_{i+1}) = \frac{\mathbf{x}''(t_{i+1}) - \mu(t_{i+1})\mathbf{x}(0)}{\sigma(t_{i+1})}$
    end for
    Input $\mathbf{x}'(t_r)$ and $\mathbf{x}''(t_r)$ to the score model
end for
Add or multiply feature maps
return Anomaly Map
```

A.3 Details VE, VP and sub-VP SDEs

We follow the definitions of VE, VP and sub-VP SDEs as in Song et al. (2021):

$$\begin{cases} \mathrm{d}\mathbf{x}(t) = \sqrt{\frac{\mathrm{d}[\sigma^2(t)]}{\mathrm{d}t}}\mathrm{d}\mathbf{w}(t), & \text{(VE SDE)} \\ \mathrm{d}\mathbf{x}(t) = -\frac{1}{2}\beta(t)\mathbf{x}(t)\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{w}(t), & \text{(VP SDE)} \\ \mathrm{d}\mathbf{x}(t) = -\frac{1}{2}\beta(t)\mathbf{x}(t)\mathrm{d}t + \sqrt{\beta(t)(1 - e^{-2\int_0^t \beta(s)\mathrm{d}s})}\mathrm{d}\mathbf{w}(t). & \text{(sub-VP SDE)} \end{cases} \tag{10}$$

VE SDE refers to Variance Exploding (VE) SDE, because VE SDE always gives a process with exploding variance when t increases. The Variance Preserving (VP) SDE yields a process with a fixed variance of one when the initial distribution has unit variance. The variance of the stochastic process induced by sub-VP SDE is always bounded by the VP SDE at every intermediate time step. See Song et al. (2021) for more information.

Because VE, VP and sub-VP SDEs all have linear drift coefficients $f(t)$, their corresponding transition densities $p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))$ are all Gaussian:

$$p_{0t}(\mathbf{x}(t)|\mathbf{x}(0)) = \begin{cases} \mathcal{N}(\mathbf{x}(t); \mathbf{x}(0), [\sigma^2(t) - \sigma^2(0)]\mathrm{I}) & \text{(VE SDE)} \\ \mathcal{N}(\mathbf{x}(t); \mathbf{x}(0)e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, \mathbf{I} - \mathbf{I}e^{-\int_0^t \beta(s)\mathrm{d}s}) & \text{(VP SDE)} \\ \mathcal{N}(\mathbf{x}(t); \mathbf{x}(0)e^{-\frac{1}{2}\int_0^t \beta(s)\mathrm{d}s}, [1 - e^{-\int_0^t \beta(s)\mathrm{d}s}]^2\mathrm{I}) & \text{(sub-VP SDE)} \end{cases} . \tag{11}$$

In particular, there are discretizations of SDEs. For VE SDE, set

$$\sigma(t) = \begin{cases} \sigma_{\min}(\frac{\sigma_{\max}}{\sigma_{\min}})^t, & t \in (0, 1] \\ 0, & t = 0. \end{cases} \tag{12}$$

For both VP SDE and sub-VP SDE, they are set as:

$$\beta(t) = \beta_{\min} + t(\beta_{\max} - \beta_{\min}). \tag{13}$$

B Implementation Details

Below, we add additional implementation details for each experiment.

MNIST Figure. For Fig. 1, to demonstrate our method and show more insights, we train a score model on subset with category “1" on MNIST dataset, while select an image with category “7" in testing. The score model is based on VE SDE, which adopts U-net architecture and code can be found in colab tutorial of `https://github.com/yang-song/score_sde_pytorch`. We choose $\sigma(t) = (25)^t$, set diffusion timesteps as 1000 and initial moment $t = 0.2$ to get the final reconstructed image.

MNIST Experiment. We choose VP SDE. Specially, $\beta_{min} = 0.1$, $\beta_{max} = 20$, and set diffusion timesteps as 1000. Based on the previous work, we adopt the positional embeddings, the layers in Ho et al. (2020b) to condition the score model on continuous time variables. As for architecture of score-based model, we take DDPM++ structure introduced in Song et al. (2021): **1)** rescales skip connections; **3)** employs BigGAN-type residual blocks; **4)** uses 2 residual blocks per resolution; and **5)** uses "residual" for input. Please see Song et al. (2021) and yang-song/score_sde_pytorch to get more information.

Exploratory experiment. For Fig. 3, based on the instantiation scheme of VE SDE, we choose $\sigma_{\min} = 0.1$ and $\sigma_{\max} = 20$. Specially, we select three data points $(-6.0, 5.0), (5.17, 5.2), (-4.2, -4.3)$. Based on our assumption and normal data distribution, $(-6.0, 5.0)$ is anomaly data. Consistent with the results in the Fig. 3, the difference of whole-score value between the “reconstructed-original noisy” data pairs is much larger than normal data.

MvTec AD and BeanTech AD dataset. The MVTec AD dataset is available at `https://www.mvtec.com/company/research/datasets/mvtec-ad/` and the BTAD dataset is available at `https://github.com/pankajmishra000/VT-ADL`. For VE SDE, we choose $\sigma_{\min} = 0.01$ and $\sigma_{\max} = 348$. For VP SDE and sub-VP SDE, we select $\beta_{\min} = 0.1$ and $\beta_{\max} = 20$. Based on the previous work, we use random Fourier feature embeddings layers introduced in Tancik et al. (2020) to condition the score model on continuous time variables for VE SDE and the scale parameter of Fourier feature embeddings is fixed to 16. For VP and sub-VP SDE, we adopt the positional embeddings. As for architecture of score-based model, we take NCSN++ structure for all SDEs : **1)** uses FIR upsampling/downsampling; **2)** rescales skip connections; **3)** employs BigGAN-type residual blocks; **4)** uses 2 residual blocks per resolution; and **5)** uses "residual" for input and no progressive growing architecture for output. The code about the score-based model can be found at yang-song/score_sde_pytorch. The results of Table 2 are based on VPSDE and Flow ODE sampling method. For the results of Table 1, to choose a set of different initial moment $\{t\}$, we adjust the maximum and minimum t in $\{t\}$, and then take time stamp every 50 steps interval. The selected feature maps or $\{t\}$ set work well and are effective in most cases but not every image. Therefore, an adaptive feature maps selection strategy would be helpful and could be our future extension.