Self-Supervised Anomaly Segmentation via Diffusion Models with Dynamic Transformer UNet

Supplementary Material

6. Discussion

In the realm of anomaly detection based on diffusion models, a full Markov chain is not necessarily required. However, when it comes to generating high-quality images, the full Markov chain becomes an essential component. Our observations reveal that as the number of diffusion steps increases, Tsimplex exhibits a decrease in sample quality, as depicted in Figure 6. A multi-section noise can be specifically designed to incorporate multi-frequency noise functions, aiming to improve sampling quality and subsequently enhance the SSIM which can be a versatile solution, offering potential avenues to enhance sample quality and tackle the asymmetry inherent in noise patterns. The presence of zig-zag patterns in the results can be attributed to the stochastic nature of Tsimplex noise as shown in Figure 11. To mitigate this, we adopted a strategy of multiple sampling and the averaging of reconstructions, resulting in a smoother Dice score graph, as illustrated in Figure 7 (DTU-Net V1). It is worth considering the integration of such strategies into the training of diffusion models in the future. Furthermore, our DTU-Net model, serving as the backbone of the diffusion model, is not limited to labeled data, unlike existing transformer-based models such as UViT [3] and DiT [30]. The effectiveness of image-level guidance indicates that we can develop a powerful anomaly detection model through guided function gradients. This strategy can also be employed for pretraining on unlabeled datasets, which can subsequently be enhanced using specific desirable functions. The methodology presented here exhibits robustness when dealing with unannotated data. In the future, we envision its potential to harness additional information in 3D images, particularly when coupled with the developed Tsimplex noise. This expansion into 3D data could offer exciting prospects for enhancing anomaly detection and image generation.

Qualitative Results: We also showcase the results of sampling from Tsimplex and Gaussian noise on the BrainMRI and leather (channels = 3) subset of the MVTec AD dataset, demonstrating excellent healthy reconstructions in Figure 1, 8.

Objective Function Stability: The self-supervised anomaly detection algorithm is designed to preserve the structure of input data by focusing on repairing anomalies while maintaining overall integrity. In our experiments, various loss functions were utilized, including L1-norm, L2-norm, L2 with L_{vlb}, SSIM loss [49], MS-SSIM loss [50], and several other combinations, aiming to determine the most effective approach. We compared these loss functions with Expected Calibration Error (ECE) [54] to assess uncertainty quality and segmentation performance. Due to the algorithm's stochastic nature, we evaluate uncertainty quality using the ECE to determine when the model's predictions are trustworthy. We presented results in Table 5 for the BrainMRI dataset using the ECE uncertainty measure, comparing absolute error (ae) and square error (se) anomaly maps between inputs and predicted outputs. Additionally, segmentation performances are reported in Table 5. Incorporating L_{vlb} and SSIM loss with the L2 measure notably reduces uncertainty in the direct difference between input and output, as shown in Table 5. However, this improvement in uncertainty comes at the expense of reduced segmentation performance. Notably, the L2 with l_{vlb} exhibits better Dice, IOU, and AUC metrics but is deemed less trustworthy.

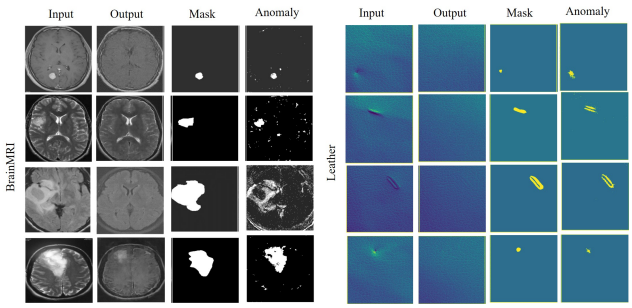


Figure 8. Qualitative visualization of anomaly segmentation in complex distributions without the need for annotations by DTU-Net with Tsimplex noise.

Nature of Tsimplex: Figure 11 showcases the histogram of the noise under various conditions within Tsimplex noise, including persistence (Figure 9(a)), octave value (Figure 9(b)), and frequency (μ) or amplitude (Figure 11(c)). We identified parameters where the histogram range approximately falls between -1 and 1 with a mean around 0. Increasing persistence expands the histogram range, as illustrated in Figure 9(a), which aligns with the formulation in Algorithm 1 for Tsimplex. We selected persistence values between 0.6 and 0.9 for a better histogram range, as demonstrated in Figure 9(d) with persistence $= 0.8$. Altering the number of octaves shifts the histogram's center away from 0. After experimentation and considering Algorithm 1's complexity, we chose octave values within the

Table 5. Performance comparison of DTU-Net's segmentation using various objective functions. Trained without averaging multiple sample outputs, employing square error for mask prediction. Comparison of ECE metrics in the BrainMRI dataset with DTU-Net, showcasing ECE uncertainty measures and highlighting absolute error (ae) and square error (se) anomaly maps. Best results in bold. Integrating multiple objectives improves model reliability and performance.

Loss function	**Dice**	**IOU**	**AUC**	**Precision**	**Recall**	ECE(ae)	ECE(se)
L1	0.3861 ± 0.2267	0.2657 ± 0.1912	0.6961 ± 0.1222	0.4436 ± 0.2281	0.4192 ± 0.2870	0.0342	0.0705
L2	0.3970 ± 0.2320	0.2757 ± 0.1958	0.6946 ± 0.1225	0.4415 ± 0.2235	0.4454 ± 0.2977	0.3011	0.0900
SSIM	0.3541 ± 0.2259	0.2401 ± 0.1838	0.6717 ± 0.1180	0.4083 ± 0.2175	0.3737 ± 0.2776	0.0435	0.0674
MS-SSIM	0.2239 ± 0.1471	0.1347 ± 0.1049	0.6682 ± 0.1170	0.5827 ± 0.202	0.1549 ± 0.1258	0.0293	0.0722
L2+L_{vlb}	$\mathbf{0.4210 \pm 0.2390}$	$\mathbf{0.2977 \pm 0.2081}$	$\mathbf{0.7084 \pm 0.1247}$	0.4634 ± 0.2350	$\mathbf{0.4621 \pm 0.2953}$	0.0332	0.0711
L2+L_{vlb}+SSIM	0.2772 ± 0.2080	0.1802 ± 0.1608	0.6810 ± 0.1395	0.5858 ± 0.2127	0.2124 ± 0.1987	**0.0164**	0.0716
L2+L_{vlb}+MS-SSIM	0.2648 ± 0.1947	0.1691 ± 0.1477	0.6791 ± 0.1303	$\mathbf{0.5891 \pm 0.1943}$	0.1980 ± 0.1804	0.0828	**0.0621**

range of $6 - 10$ to cover a range of values, manage time costs, and maintain a center closer to 0. Next, we examined the effect of frequency (μ) on the histogram (Figure 9(c)). Our analysis revealed that frequencies ranging from 2^0 to 2^5 ($\mu \in [0, 5]$) produced more symmetric noise. However, despite this, the histogram ranges remained nearly between -1 and $+1$. Consequently, combining noises generated at different μ values might yield superior results. Figure 9(d) relies on octave $= 10$, persistence $= 0.8$, and frequency $= 128$, producing a more favorable histogram across all samples.

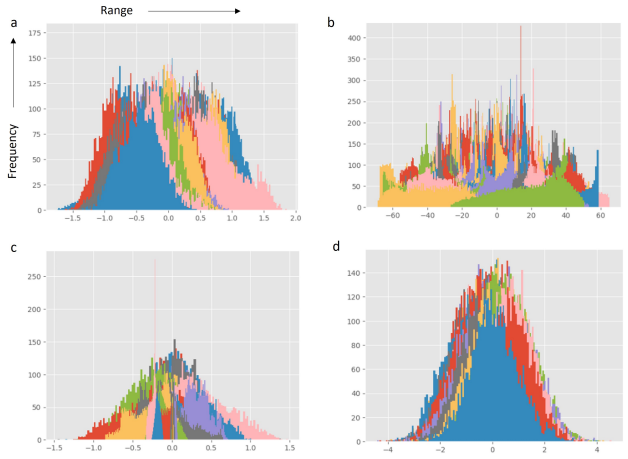


Figure 9. Histogram of noise under various conditions within Tsimplex noise. (a) Shows the impact of persistence on the histogram range; increasing persistence expands the range, in line with Algorithm 1 for Tsimplex. (b) Demonstrates the effect of altering the number of octaves on the histogram's center. (c) Explores the effect of frequency μ on kurtosis and histogram symmetry. (d) Displays the combined effects of octave, persistence, and frequency settings on the histogram, highlighting a more favorable histogram with specific settings.

7. Implementation details

We employ the hyperparameters detailed in Table 6. The model is implemented using PyTorch and trained on a single GPU, specifically the NVIDIA RTX A4000, which boasts 16GB of GDDR6 VRAM. Additionally, we conduct an area under the curve (AUC) for further comparison, as depicted in Figure 10. We train only in healthy images with the goal of repairing the anomaly. We compute the $(\{anomaly - repaired\}\text{image})^2$ followed by binarization for testing the method on segmentation tasks using a variety of segmentation measures.

Table 6. Hyperparameters

Parameter	**Value**
Image Settings	
Img_size	$(224, 224)$
Batch_Size	32
Epochs	3000
Time_step	1000
Model Configuration	
channels	1 or 3
beta_schedule	cos
loss-type	$l2 - norm$
learning rate	1e-4
patch_size	16
embed_dim	384
depth	6
num_heads	6
mlp_ratio	4
num_class	null or 2
EMA rate	0.9999
Tsimplex Parameters	
octave	6
frequency	64
persistence	0.9

7.1. Simplex Noise generation

Tsimplex is based on simplex as shown in the Algorithm 1. We use OpenSimplex[5] for generating simplex noise and the steps are shown in Algorithm 2 in simplified. It is gen-

[5] https://github.com/lmas/opensimplex

erated by organizing a grid of pseudo-random gradient vectors in a multi-dimensional space. When evaluating the noise value at a specific point within this space, the algorithm identifies the closest grid points around that location and retrieves their associated gradient vectors. By calculating dot products between these gradients and vectors from the nearby grid points to the target position, it determines their contributions to the overall noise value. Applying interpolation techniques to blend these contributions results in a smoothed, continuous noise value at the given point. This process is repeated across multiple octaves, varying frequencies, and amplitudes to create more intricate and detailed patterns. The culmination of these steps yields a coherent and natural-looking noise output used extensively in computer graphics for generating realistic textures, simulating terrain, and other applications requiring organic randomness.

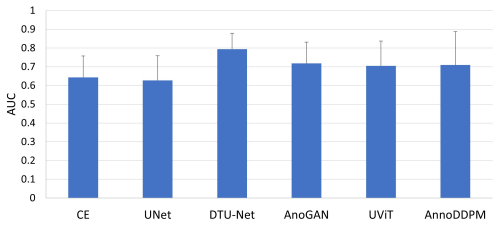


Figure 10. Comparision with the AUC matric on BrainMRI dataset for CE [29], AnnoGAN [41], AnnoDDPM [53], DDPM [18] with gaussion noise, and some backbones, including UNet [10], UViT [3], and prposed DTU-Net with Tsimplex.

Algorithm 2 Simplex Noise

Input: Input point $\mathbf{P} = (x, y, z)$
Output: Noise value $N(\mathbf{P})$
Initialize $N(\mathbf{P}) \leftarrow 0$
Step 1: Grid Setup
Define a regular grid of points by dividing the space into a grid of cells: $\mathbf{G} = \{\mathbf{G}_1, \mathbf{G}_2, \ldots, \mathbf{G}_n\}$.
Step 2: Grid Positioning
Determine the position of $\mathbf{P}$ within the grid by finding the closest grid points to $\mathbf{P}$.
$\mathbf{V} = \{\mathbf{V}_1, \mathbf{V}_2, \ldots, \mathbf{V}_n\}$, where $\mathbf{V}_i$ is the position of the i-th closest grid point to $\mathbf{P}$.
Step 3: Gradient Vectors
Calculate the pseudo-random gradient vector $\mathbf{G}_i$ for each grid point: $\mathbf{G}_i = (g_{ix}, g_{iy}, g_{iz})$.
Step 4: Dot Products
Calculate the dot products between $\mathbf{G}_i$ and vectors from the closest grid points to $\mathbf{P}$: $D_i = \mathbf{G}_i \cdot (\mathbf{P} - \mathbf{V}_i)$.
Step 5: Interpolation
Interpolate the dot products using a smooth function to obtain $N(\mathbf{P})$:
$N(\mathbf{P}) = \sum_{i=1}^{n} F(D_i)$, where $F(D_i)$ is the interpolation function.
Step 6: Octaves
Repeat Steps 1-5 for multiple octaves with different frequencies and amplitudes, accumulating the results in $N(\mathbf{P})$.
return $N(\mathbf{P})$

8. Nature of Tsimplex noise

8.1. Symmetry

We've noticed a slight decline in sample quality in Tsimplex models, especially when exposed to higher levels of noise (denoted by a further 't' value). This decrease is likely due to the asymmetry present in the Tsimplex noise function. To investigate this, we conducted a study on the symmetry of the Tsimplex to assess how its parameters influence this asymmetry. To measure the symmetry of the noise, we employed the statistical Kurtosis score [51].

Kurtosis is a statistical metric that quantifies the extent of outliers or the distribution's 'tailedness' in comparison to a normal distribution. Mathematically, the formula for sample kurtosis can be expressed as:

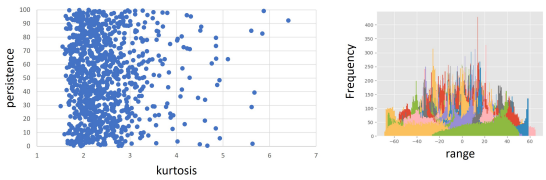
$$\text{Kurtosis} = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{x_i - \bar{x}}{s} \right)^4$$

Here, x_i represents individual observations, $\bar{x}$ stands for the sample mean, s denotes the sample standard deviation, and n is the number of observations. This formula calculates the fourth standardized moment of the data distribution. A kurtosis value of 3 indicates a distribution similar to a normal one (mesokurtic). Values greater than 3 imply heavier tails or more outliers (leptokurtic), while values less than 3 suggest lighter tails or fewer outliers (platykurtic).

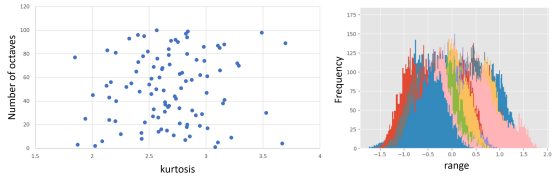
Figure 11 showcases the Kurtosis score and histogram of the noise under various conditions within Tsimplex noise, including persistence (Figure 11a), octave value (Figure 11b), and frequency (μ) or amplitude (Figure 11c). We identified parameters where the histogram range approximately falls between -1 and 1 with a mean around 0. Increasing persistence expands the histogram range, as illustrated in Figure 11a, which aligns with the formulation in Algorithm 1 for Tsimplex.

We selected persistence values between 0.6 and 0.9 for a better histogram range, as demonstrated in Figure 11d with persistence $= 0.8$. Altering the number of octaves shifts the histogram's center away from 0. After experimentation and considering Algorithm 1's complexity, we chose octave values within the range of $6 - 10$ to cover a range of values, manage time costs, and maintain a center closer to 0, as depicted in Figure 11d.

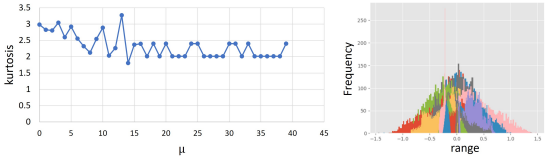
Next, we examined the effect of frequency (μ) on kurtosis and the histogram (Figure 11c). Our analysis revealed that frequencies ranging from 2^0 to 2^5 produced more symmetric noise. However, despite this, the histogram ranges remained nearly between -1 and $+1$. Consequently, we believe that combining noises generated at different μ values might yield superior results. Figure 11d relies on octave $= 10$, persistence $= 0.8$, and frequency $= 128$, producing a more favorable histogram, with kurtosis averaging around 3 across all samples from $0 - 100$.



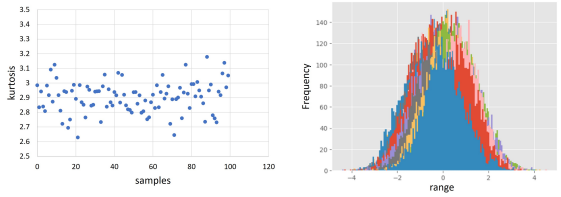
(a) Illustrates the impact of persistence on histogram range; increasing persistence expands the range, aligned with Algorithm 1 for Tsimplex.



(b) Demonstrates the effect of altering the number of octaves on the histogram's center.



(c) Explores the effect of frequency μ on kurtosis and histogram symmetry. $\mu = 5$ means frequency $= 2^5$



(d) Displays the combined effects of octave, persistence, and frequency settings on the histogram, highlighting a more favorable histogram with specific settings (octave =10, persistence =0.8, frequency =128)

Figure 11. Kurtosis score and histogram of noise under various conditions within Tsimplex noise.

8.2. Stochasticity

Because of the random patterns in Tsimplex noise, the algorithm behaves stochastically which can also seen in Figure 11d where the kurtosis score is around 3 but not the same for some set of parameters. This stochasticity is due to the random patterns in the Tsimplex shown in Figure 12.

To solve this problem, we explore sampling forward diffusion more than once and averaging the corresponding reconstructions for better reconstruction.

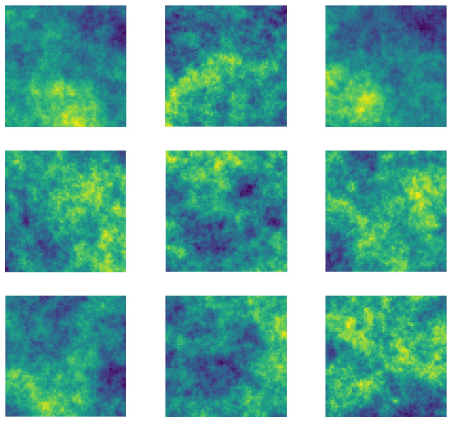


Figure 12. The visual representation showcases the intricate structure of Tsimplex noise with parameters set as octave=10, persistence=0.8, and frequency=128, revealing the random patterns inherent in the noise.

8.3. Further Noise function

We have endeavored to enhance the sample quality by introducing two new noise sources: Tsimplex Gauss (Tsg) and Komal-Tsimplex (KTs) noise. Gaussian noise, known for its high-quality samples, is integrated into Tsg, which is formulated as follows:

$$\text{Tsg}(x) = \alpha \cdot \text{GenNoise}(S, t) + (1 - \alpha) \cdot \text{gauss}(x), \quad (6)$$

Here, $\alpha = \left(1 - \frac{t}{\chi}\right)$, with t representing the current diffusion step and χ denoting the total diffusion steps. For KTs noise, we introduce the multi-frequency ($F_{i,j}$) for simplex noise, where the position of a pixel (i, j), and described by the following equation:

$$F_{i,j} = \delta \times e^{-\left(\frac{(i-\frac{n+1}{2})^2}{2\sigma^2} + \frac{(j-\frac{m+1}{2})^2}{2\sigma^2}\right)} \quad (7)$$

In this equation, δ represents the desired average frequency value, n and m are the dimensions of the grid, and σ controls the spread of the Gaussian distribution. A larger value of σ results in a Komal (smoother) distribution. In Figure 13, we illustrate the impact of diffusion steps on SSIM (Structural Similarity Index Measure), highlighting that Komal-Tsimplex yields superior sample outputs compared to other methods. Our experiments indicate that while Tsipplex possesses section-wise frequency noise, it struggles to rectify anomalies, suggesting the need for future investigations in this specific direction.

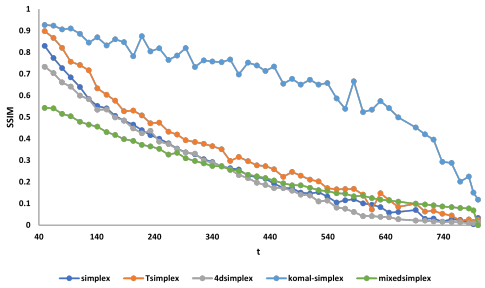


Figure 13. Effect of diffusion steps on SSIM using DTU-Net as the backbone with a variety of noise functions. Komal-Simplex, tailored for section-based noise schemes, demonstrates superior SSIM quality compared to others. Further optimization could enhance Dice and AUC scores.

9. Guiding Function

The self-supervised anomaly detection algorithm is designed to preserve the structure of input data by focusing on repairing anomalies while maintaining overall integrity. In our experiments, various loss functions were utilized, including l1-norm, l2-norm, l2 combined with total variation (TV) loss [8], l2 with l_{vlb}, SSIM loss [49], MS-SSIM loss [50], and several other combinations, aiming to determine the most effective approach. We compared these loss functions with Expected Calibration Error (ECE) to assess uncertainty quality and segmentation performance. Due to the algorithm's stochastic nature, we evaluate uncertainty quality using the Expected Calibration Error (ECE) to determine when the model's predictions are trustworthy. ECE is defined as:

$$\mathrm{ECE} = \sum_{i=1}^{N} \frac{|B_i|}{N} \cdot |\mathrm{acc}(B_i) - \mathrm{conf}(B_i)|$$

Here, N represents the total number of bins, and B_i denotes the i-th bin. The accuracy ($\mathrm{acc}(B_i)$) of a bin signifies the alignment between model predictions and actual outcomes, while the confidence ($\mathrm{conf}(B_i)$) indicates the certainty level associated with those predictions. ECE evaluates the disparity between predicted confidence and actual accuracy across multiple bins to gauge the model's calibration performance.

Table 7. Comparison of ECE metrics in the BrainMRI dataset with DTU-Net. The table exhibits ECE uncertainty measures, highlighting absolute error (ae) and square error (se) anomaly maps between input and predicted outputs. The best objective function, denoted in bold, demonstrates lower ECE values, indicating higher statistical reliability.

Loss function	ECE (ae)	ECE (se)
L1	0.03425	0.07057
L2	0.30111	0.09001
L2+TV	0.08288	**0.06214**
L2 + l_{vlb}	0.0332	0.07114
SSIM	0.04359	0.06747
MS-SSIM	0.02932	0.07223
L2+l_{vlb}+SSIM	**0.01645**	0.07165

We presented results in Table 7 for the BrainMRI dataset using the ECE uncertainty measure, comparing absolute error (ae) and square error (se) anomaly maps between inputs and predicted outputs. Additionally, segmentation performances are reported in Table 8. Incorporating l_{vlb} and SSIM loss with the L2 measure notably reduces uncertainty in the direct difference between input and output, as shown in Table 7. However, this improvement in uncertainty comes at the expense of reduced segmentation performance. Notably, the L2 with l_{vlb} exhibits better Dice, IOU, and AUC metrics but is deemed less trustworthy.

Table 8. Performance comparison of DTU-Net's segmentation capabilities across various objective functions. DTU-Net is trained using different loss functions without averaging multiple sample outputs, employing square error as a predictor for the mask. Best-performing results are highlighted in bold. Integrating multiple objectives enhances model reliability and overall performance.

Loss function	**Dice**	**IOU**	**AUC**	**Precision**	**Recall**
L1	0.3861 ± 0.2267	0.2657 ± 0.1912	0.6961 ± 0.1222	0.4436 ± 0.2281	0.4192 ± 0.2870
L2	0.3970 ± 0.2320	0.2757 ± 0.1958	0.6946 ± 0.1225	0.4415 ± 0.2235	0.4454 ± 0.2977
SSIM	0.3541 ± 0.2259	0.2401 ± 0.1838	0.6717 ± 0.1180	0.4083 ± 0.2175	0.3737 ± 0.2776
MS-SSIM	0.2239 ± 0.1471	0.1347 ± 0.1049	0.6682 ± 0.1170	0.5827 ± 0.202	0.1549 ± 0.1258
L2+l_{vlb}	$\mathbf{0.4210 \pm 0.2390}$	$\mathbf{0.2977 \pm 0.2081}$	$\mathbf{0.7084 \pm 0.1247}$	0.4634 ± 0.2350	$\mathbf{0.4621 \pm 0.2953}$
L2+l_{vlb}+SSIM	0.2772 ± 0.2080	0.1802 ± 0.1608	0.6810 ± 0.1395	0.5858 ± 0.2127	0.2124 ± 0.1987
L2+l_{vlb}+MS-SSIM	0.2648 ± 0.1947	0.1691 ± 0.1477	0.6791 ± 0.1303	$\mathbf{0.5891 \pm 0.1943}$	0.1980 ± 0.1804