Beyond ViT Tokens: Masked-Diffusion Pretrained Convolutional Pathology Foundation Model for Cell-Level Dense Prediction

Weiming Chen[*,1], **Xitong Ling**[*,1,3], **Zhenyang Cai**[2], **Xidong Wang**[2], **Jiawen Li**[1], **Tian Guan**[1], **Benyou Wang**[†,2] **Yonghong He**[†,1,3,4]

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University
[2]The Chinese University of Hong Kong, Shenzhen
[3]Research Institute of Tsinghua, Pearl River Delta
[4]Jinfeng Laboratory

Abstract

Cell-level dense prediction is central to computational pathology, but remains challenging due to fine-grained histological structures, strong domain shifts, and costly dense annotations. Existing ViT-based pathology foundation models rely on patch tokenization, which can disrupt spatial continuity and weaken local morphological details needed for cell-level prediction. To address this, we propose Masked-Diffusion Convolutional Foundation Models, termed **C**onvNeXt **M**asked-**D**iffusion (**CMD**), a self-supervised convolutional generative pretraining framework for dense pathology representation learning. CMD uses a fully convolutional ConvNeXt-UNet backbone, performs masked-diffusion pretraining in pixel space, and incorporates frozen pathology foundation model features through adaptive normalization. Experimental results demonstrate that CMD consistently outperforms existing ViT-based pathology foundation models and even surpasses state-of-the-art end-to-end segmentation methods while fine-tuning only a small number of task-specific parameters across multiple pathology dense prediction tasks. The advantage is particularly pronounced under limited annotation settings, where CMD exhibits stronger robustness and generalization ability. Our findings suggest that purely convolutional architectures can also serve as competitive pathology foundation models for cell-level dense prediction, achieving leading performance within the current ViT-dominated paradigm and providing a scalable, high-performance solution that better preserves histological structural priors for fine-grained pathology understanding. Code is available at `https://github.com/M4A1TasteGood/ConvNeXt-Masked-Diffusion`

1 Introduction

Cell-level dense prediction is central to computational pathology, enabling nuclear segmentation, inflammatory cell detection, and tissue microenvironment analysis. This task remains difficult because pathological structures are often tiny, morphologically diverse, and separated by ambiguous boundaries, while dense annotations are costly and domain shifts across stains, scanners, tissues, and institutions are substantial.

Recent pathology foundation models largely build on ViT architectures. Although effective for high-level recognition, ViT-style patch tokenization is not naturally suited to cell-level dense prediction:

[*]Contributed equally
[†]Corresponding authors

fixed-size patches may disrupt continuous histological structures and lose intra-patch morphology, texture, and boundary details that are critical for pixel- or instance-level prediction. This suggests that convolutional networks, with their locality and spatial continuity biases, are a more suitable architectural choice for fine-grained pathology representation learning.

However, convolutional networks still lack a scalable pretraining paradigm comparable to masked image modeling for ViTs. Inspired by recent generative models, which demonstrate a strong ability to learn high-fidelity image representations through reconstruction and synthesis, we ask whether generative modeling can serve as an effective self-supervised pretraining recipe for convolutional pathology foundation models.

To this end, we propose Masked-Diffusion Convolutional Foundation Models, termed **C**onvNeXt **M**asked-**D**iffusion (**CMD**), a generative self-supervised pretraining framework for cell-level dense prediction in pathology. CMD performs masked-diffusion pretraining in pixel space with a convolutional architecture, preserving local spatial continuity while learning morphology-aware representations. We instantiate CMD with a ConvNeXt-UNet backbone and condition the diffusion process on pathology foundation model features, combining semantic priors with fine-grained spatial reconstruction.

Across multiple cell-level pathology benchmarks, including multi-dataset training, few-shot adaptation, and scaling settings, CMD consistently outperforms ViT-based pathology foundation models and state-of-the-art end-to-end segmentation networks.

Our contributions are summarized as follows:

I. We propose **CMD**, a self-supervised Masked-Diffusion Convolutional Foundation Model for learning cell-level dense representations in pathology.

II. We systematically design the key components of the framework, including the masked-diffusion objective, ConvNeXt-UNet backbone, pixel-space modeling, and pathology foundation model conditioning.

III. We validate the generalizability of the learned representations across multiple datasets, few-shot scenarios, and scaling regimes, with visualizations showing localized, cell-aware dense features.

2 Related Work

Pathology Foundation Models. Pathology foundation models (PFMs) provide transferable representations for computational pathology. Most existing PFMs follow vision-only self-supervised pretraining with ViT backbones and objectives such as DINOv2 [1], iBOT [2], or masked image modeling [3], and are later paired with MIL-style aggregators [4, 5, 6] for WSI-level tasks. Recent advances further explore end-to-end adaptation paradigms that directly convert pretrained vision backbones into whole-slide survival prediction models, streamlining the pipeline from patch-level pretraining to slide-level clinical tasks [7]. Representative models include UNI/UNI2 [8], Virchow/Virchow2 [9, 10], PathOrchestra [11], Phikon [12, 13], Prov-GigaPath [14], Hibou [15], Kaiko [16], Digepath [17], StainNet [18], GPFM [19], Midnight-12k [20], GenBio-PathFM [21], and ALICE [22], which unifies vision, vision-language, and slide-level expert priors into a general-purpose pathology foundation model. Another line uses vision–language pretraining to align pathology images with reports or biomedical text, such as PLIP [23], CONCH [24, 25] and MUSK [26]. Beyond global image-text alignment, diagnostic text-guided representation learning has been proposed for hierarchical WSI classification, leveraging structured pathological descriptions to guide multi-level feature aggregation and refine fine-grained slide-level semantics [27]. While effective for global image- or slide-level semantics, ViT-based PFMs may lose fine local continuity due to patch tokenization, limiting their suitability for cell-level dense prediction.

Dense Prediction in Pathology. Dense prediction [28] supports pixel- or region-level analysis of nuclei, glands, tumor regions, immune cells, and tissue microenvironment components [29]. Existing methods often adapt natural-image segmentation architectures to pathology. TransUNet [30] combines CNN local features with Transformer global context, ViT-Adapter [31] adapts pretrained ViTs through multi-scale modules. These methods improve supervised dense prediction, but typically require end-to-end training with dense annotations.

Generative and Convolutional Pretraining. Generative pretraining learns representations by reconstructing or synthesizing image content. Masked image modeling is widely used for ViT pretraining [3], while diffusion models learn visual structure through denoising objectives. Masked diffusion [32] further views diffusion as time-conditioned reconstruction, suggesting that the corruption process can be designed for representation learning rather than image synthesis alone. In parallel, modern convolutional architectures such as ConvNeXt [33] and ConvNeXt V2 [34] provide strong locality bias and efficient multi-scale feature extraction, making convolutional pretraining an important alternative to token-based visual representation learning.

3 Method

We develop CMD by addressing five key design choices: the pretraining objective, backbone architecture, representation space, foundation-model conditioning, and downstream feature extraction. This section first summarizes the overall pipeline.

3.1 Overview: Frozen Generative Pretraining for Cell-Level Dense Prediction

CMD aims to learn a reusable generative pathology representation rather than another task-specific supervised segmentor. Given unlabeled pathology image patches, we pretrain a ConvNeXt masked-diffusion model in a self-supervised manner, following the masked diffusion formulation [32]. The model takes partially masked patches as input and learns to recover the clean image content through a diffusion denoising objective, encouraging it to capture local morphology, texture continuity, and cellular boundaries without dense annotations.

After pretraining, we discard the sampling process and use the pretrained diffusion network as a frozen dense feature extractor. Multi-scale feature maps are extracted from intermediate decoder blocks, providing both fine spatial details and contextual information for downstream cell-level dense prediction. For each task, only a lightweight prediction head is trained, while the pretrained generative representation remains fixed.

This protocol separates representation learning from task-specific supervision: expensive pretraining is performed once on unlabeled pathology data, and downstream adaptation requires only limited annotations. It also allows us to evaluate whether masked-diffusion pretraining learns generalizable pathology representations, rather than relying on end-to-end supervised tuning. The overall pipeline is shown in Figure 1.

3.2 Design Question I: Why ConvNeXt-UNet as the Diffusion Backbone?

ConvNeXt-UNet Backbone for Microscopic Locality and Multi-Scale Structure

The diffusion backbone determines how effectively masked-diffusion pretraining captures pathology morphology. For cell-level dense prediction, features must preserve nuclear contours, chromatin texture, thin boundaries, and multi-scale tissue context. We therefore compare DiT [35], Attention U-Net [36], and ConvNeXt-UNet under the same pretraining and evaluation protocol.

DiT offers scalable global modeling, but patch tokenization may weaken small structures and intra-patch boundaries. Attention U-Net is naturally suited to dense prediction, yet its conventional blocks may limit scalability. ConvNeXt-UNet combines U-Net-style multi-scale feature reuse with modern ConvNeXt blocks, providing locality bias, efficient channel mixing, and adaptive conditioning with timestep and pathology foundation model features.

As shown in Table 1, ConvNeXt-UNet achieves stronger boundary-sensitive dense representations across datasets, supporting its use as the CMD diffusion backbone.

3.3 Design Question II: Why Masked-Diffusion Instead of Standard DDPM?

Masked-Diffusion Objective for Morphology-Aware Self-Supervision

Inspired by masked diffusion [32], we treat diffusion pretraining as time-conditioned reconstruction rather than only generative sampling. From this view, the timestep controls corruption difficulty, and

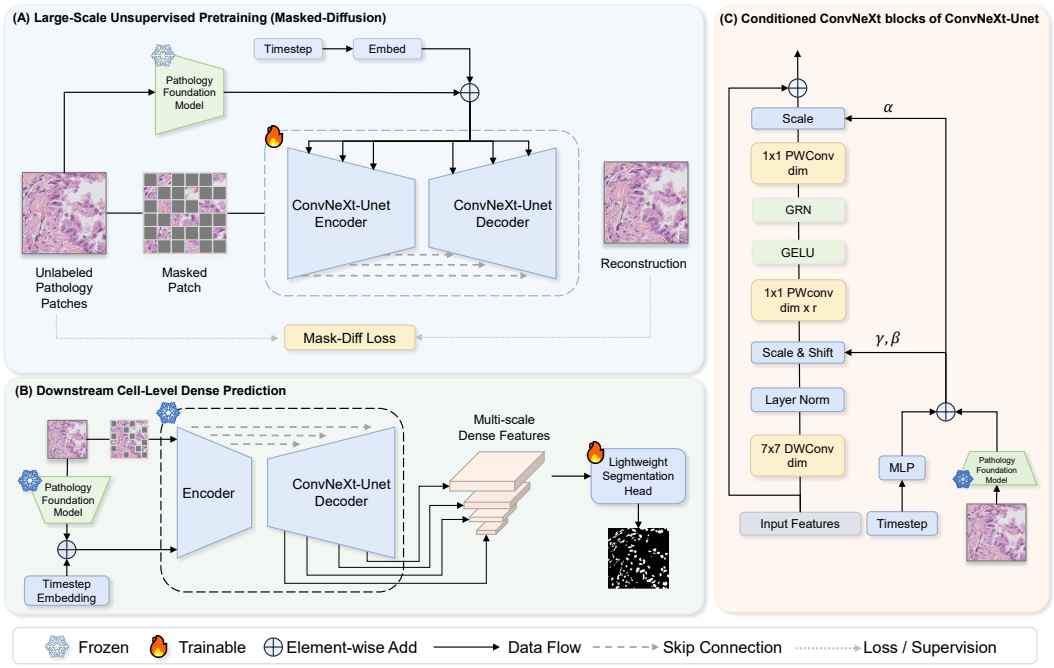

Figure 1: Overview of the proposed CMD framework. (A) Large-scale unlabeled pathology patches are used for self-supervised masked-diffusion pretraining, where timestep embeddings and pathology foundation model features condition a ConvNeXt U-Net to reconstruct masked patches. (B) After pretraining, the frozen generative backbone is reused as a dense visual encoder: multi-scale decoder features are extracted and fed into lightweight task-specific heads for cell-level dense prediction. (C) Each conditioned ConvNeXt block injects the fused timestep and pathology features via adaLN.

Table 1: Backbone comparison under the same diffusion setting. DiT, Attention U-Net, and ConvNeXt-UNet all use VAE representations and pathology foundation model conditioning, are pretrained on 55K unlabeled pathology images with comparable parameter sizes, and are evaluated with frozen pretrained features and a linear-probe segmentation head. BF1 denotes boundary F1 score, measuring boundary agreement between predicted and ground-truth masks. Values in brackets denote 95% confidence intervals estimated with 1000 bootstrap resamples. Detailed experimental settings are provided in Appendix B.4.

Backbone(linear-probe)	CPM-15 [37]		CPM-17 [37]		Janowczyk [38]		Kumar [39]		TNBC [40]	
	Dice	BF1	Dice	BF1	Dice	BF1	Dice	BF1	Dice	BF1
DiT	44.39 [38.74, 50.04]	19.65 [17.71, 21.58]	56.33 [51.37, 61.12]	23.24 [12.60, 32.62]	50.56 [49.95, 51.26]	5.15 [3.36, 7.34]	48.80 [46.73, 52.28]	11.12 [9.45, 12.65]	55.11 [53.19, 56.97]	21.12 [15.63, 25.75]
Attention U-Net	53.68 [49.98, 57.38]	19.31 [12.57, 26.04]	56.04 [53.71, 59.12]	23.19 [19.62, 26.25]	49.50 [49.24, 49.76]	1.67 [0.89, 2.59]	50.37 [47.53, 55.40]	20.80 [14.66, 28.45]	54.05 [51.91, 56.02]	16.39 [8.68, 22.57]
ConvNeXt-UNet	50.19 [49.39, 50.99]	33.66 [31.42, 35.89]	55.20 [52.82, 57.39]	35.04 [27.83, 41.36]	50.27 [49.86, 50.75]	6.45 [4.24, 8.92]	51.61 [51.27, 51.90]	34.38 [31.67, 37.95]	53.23 [51.86, 54.60]	25.09 [14.05, 32.77]

Gaussian noise in standard DDPM [41] can be replaced by a corruption process better aligned with representation learning.

For cell-level pathology, we use structure-oriented masking instead of Gaussian corruption. CMD learns to recover missing histological regions from visible context, encouraging representations that capture local morphology, tissue texture, nuclear boundaries, and spatial organization.

Given an unlabeled pathology patch x_0, we sample $t \in [1, T]$ and set $r_t = t/(T+1)$. Random non-overlapping patches are masked according to r_t to obtain x_t, and the ConvNeXt-UNet reconstructs the original image or latent representation conditioned on the timestep and pathology foundation

model feature:

$$x_t = \mathcal{M}(x_0, r_t), \quad r_t = \frac{t}{T+1}, \tag{1}$$

$$\hat{x}_0 = f_\theta(x_t, t, z_{\text{pfm}}), \tag{2}$$

$$\mathcal{L}_{\text{masked-diff}} = \mathbb{E}_{x_0,t}\left[\|x_0 - f_\theta(x_t, t, z_{\text{pfm}})\|_1\right], \tag{3}$$

where $\mathcal{M}(\cdot)$ denotes timestep-controlled patch masking, z_{pfm} is the frozen pathology foundation model feature, and f_θ is the trainable ConvNeXt-UNet. A detailed theoretical derivation is provided in Appendix A.

Thus, t changes from controlling Gaussian noise strength in DDPM to controlling structural occlusion in CMD. Timestep and pathology features are fused and injected into ConvNeXt blocks through adaptive Layer Normalization.

3.4 Design Question III: Pixel-Space or VAE Latent-Space Pretraining?

Pixel-Space Masked Diffusion for Preserving Cell-Level Details

Masked diffusion can operate in pixel space or compressed VAE [42] latent space. While latent-space diffusion is efficient, VAE compression may discard high-frequency cues such as nuclear boundaries, chromatin texture, and small inter-cell gaps, which are critical for pathology dense prediction.

We compare pixel-space pretraining, VAE latent-space pretraining, and a high-resolution VAE variant. As shown in Table 2, pixel-space masked diffusion yields stronger downstream dense features, suggesting that preserving native pixel-level morphology is more important than latent-space efficiency for cell-level prediction.

Table 2: Pixel-space versus VAE latent-space masked-diffusion pretraining. Pixel-space and standard VAE latent-space settings use 256×256 inputs, while the High-res VAE latent setting uses 512×512 inputs. Other experimental settings are the same as in Table 1.

Pretraining Space	CPM-15		CPM-17		Janowczyk		Kumar		TNBC	
	Dice	BF1	Dice	BF1	Dice	BF1	Dice	BF1	Dice	BF1
VAE latent-space	50.19 [49.39, 50.99]	33.66 [31.42, 35.89]	55.20 [52.82, 57.39]	35.04 [27.83, 41.36]	50.27 [49.86, 50.75]	6.45 [4.24, 8.92]	51.61 [51.27, 51.90]	34.38 [31.67, 37.95]	53.23 [51.86, 54.60]	25.09 [14.05, 32.77]
High-res VAE latent	47.96 [47.95, 47.98]	11.52 [9.54, 13.50]	63.80 [58.88, 68.49]	30.49 [24.61, 35.82]	51.87 [51.24, 52.63]	11.03 [9.00, 13.37]	53.09 [51.39, 54.40]	33.31 [30.64, 36.10]	58.24 [56.51, 59.81]	22.43 [19.24, 24.80]
Pixel-space	63.25 [59.34, 67.16]	56.03 [55.41, 56.64]	76.74 [70.29, 82.39]	73.45 [62.26, 83.70]	53.66 [52.71, 54.70]	18.24 [13.99, 22.46]	62.76 [49.07, 70.45]	62.25 [26.32, 83.66]	73.51 [68.28, 80.45]	75.63 [66.74, 86.52]

3.5 Design Question IV: Is Pathology Foundation Model Conditioning Essential?

Pathology Foundation Model Conditioning as Complementary Semantic Guidance

Pathology foundation models provide global tissue context and high-level morphological semantics. CMD uses frozen pathology foundation model features as conditional guidance, complementing local masked reconstruction with pathology-aware semantic priors.

A frozen foundation model extracts an image-level feature from the original patch, which is fused with the timestep embedding and injected into ConvNeXt-UNet blocks through adaptive Layer Normalization. The dense representation is still learned by the masked-diffusion backbone.

We compare ConvNeXt masked diffusion without conditioning and full CMD with conditioning, using UNI as the foundation model. As shown in Table 3, conditioning further improves performance, indicating that PFM features provide complementary semantic guidance rather than replacing masked-diffusion representation learning.

Table 3: Effect of pathology foundation model conditioning. Other experimental settings are the same as in Table 1.

Method	CPM-15		CPM-17		Janowczyk		Kumar		TNBC	
	Dice	BF1	Dice	BF1	Dice	BF1	Dice	BF1	Dice	BF1
CMD w/o PFM	55.17 [50.90, 59.45]	43.52 [40.68, 46.36]	71.29 [64.18, 78.05]	64.36 [50.59, 77.59]	52.21 [51.47, 53.08]	14.19 [10.53, 17.69]	60.21 [55.20, 66.46]	68.02 [61.82, 76.16]	60.24 [52.82, 67.65]	45.90 [32.53, 58.28]
CMD w/ PFM	63.25 [59.34, 67.16]	56.03 [55.41, 56.64]	76.74 [70.29, 82.39]	73.45 [62.26, 83.70]	53.66 [52.71, 54.70]	18.24 [13.99, 22.46]	62.76 [49.07, 70.45]	62.25 [26.32, 83.66]	73.51 [68.28, 80.45]	75.63 [66.74, 86.52]

4 Experiments

4.1 Experimental Setup

4.1.1 Pretraining data

We pretrain CMD on a large-scale unlabeled pathology corpus with approximately 1 million 512×512 image patches. All patches are unlabeled and used only for self-supervised masked-diffusion pretraining. Additional dataset details are provided in Appendix B.1.

4.1.2 Pretraining protocol

During pretraining, each 512×512 pathology patch is converted into a 256×256 input. We use a mixed resizing strategy: with 80% probability, we randomly crop a 256×256 region, and with 20% probability, we resize the whole patch to 256×256. This exposes the model to both fine cellular details and broader tissue layouts.

We use masked diffusion with $T = 1000$ timesteps. For a sampled timestep t, the masking ratio is $r_t = t/T$, and random non-overlapping patches are masked and reconstructed by the diffusion backbone. For pixel-space pretraining, the mask patch size is 8, yielding a 32×32 patch grid for 256×256 inputs. When pathology foundation model conditioning is enabled, we use a frozen H0-mini encoder and inject its image-level feature together with the timestep embedding into ConvNeXt blocks through adaptive Layer Normalization.

We choose the lightweight H0-mini encoder as the default conditioning backbone for all main cross-model comparisons in Section 4 to ensure fair parameter-scale alignment with existing ViT-based pathology foundation model baselines. The full CMD-L pipeline with H0-mini conditioning totals approximately 602M parameters, which is comparable to the parameter budget of most mainstream baselines. In contrast, using a larger foundation model such as UNI would raise the total parameter count to roughly 823M, substantially exceeding most competing methods and introducing a parameter-related performance advantage that would confound fair comparison. We use UNI as the conditioning model in Section 3's ablation study specifically to isolate and demonstrate the semantic guidance effect of foundation model conditioning, independent of cross-model fairness considerations.

To further verify the generality of our conditioning framework and show that performance can be further scaled with stronger semantic priors, we evaluate CMD-L with three additional state-of-the-art pathology foundation models (UNI, Virchow2, and H-optimus-1) as conditioning inputs. Full quantitative results and analysis are provided in Appendix E.

The model is optimized with AdamW using a learning rate of 3×10^{-5}, no weight decay, BF16 mixed precision, and an exponential moving average decay of 0.9999. Unless otherwise specified, we train for 80K optimization steps and use the EMA checkpoint for downstream evaluation.

4.1.3 Downstream task protocol

For downstream cell-level dense prediction, we freeze the pretrained CMD backbone and use it only as a dense feature extractor. Multi-scale features are extracted from selected decoder blocks and passed to a task-specific segmentation head. During downstream training, only the segmentation head is optimized, using a cosine learning-rate schedule and a combined cross-entropy plus Dice loss. Details of the downstream head designs are provided in Appendix B.3.

4.2 Comparisons With State-of-the-art Methods

We evaluate CMD on cell-level dense prediction under the frozen-backbone setting and compare it with both frozen pathology foundation models and end-to-end segmentation baselines. The frozen-backbone comparison isolates representation quality, while the end-to-end baselines provide strong task-specific references trained directly for segmentation.

For the frozen ViT-based pathology foundation models, patch tokens from the final Transformer layer, of shape (B, N, C) with $N = H_p \times W_p$, are deterministically reshaped into a dense feature map (B, C, H_p, W_p). A shared lightweight decoder—a 3×3 convolution followed by progressive bilinear upsampling—restores the input resolution, and a segmentation head produces the pixel-level prediction.

As shown in Table 4, CMD achieves stronger overall performance than ViT-based pathology foundation models and remains competitive with, or superior to, end-to-end segmentation models across CPM-15, CPM-17, and TNBC. This indicates that CMD is not only a stronger frozen pathology representation, but also provides dense features that can rival specialized segmentation architectures.

An important advantage of CMD is its reduced sensitivity to input resolution. CMD uses a unified 256×256 downstream input, whereas ViT-based foundation models and end-to-end baselines often require larger dataset-specific resolutions. Despite this smaller input size, CMD matches or outperforms high-resolution ViT-based models, suggesting that the convolutional masked-diffusion representation captures local morphology efficiently without relying on larger image fields.

Figure 2 further supports this observation. Frozen ViT-based foundation models often capture coarse foreground regions but miss small nuclei, merge adjacent cells, or produce noisy boundaries in crowded tissue areas. End-to-end baselines improve spatial layout but can still produce fragmented boundaries or incomplete cell separation. In contrast, CMD-L better preserves small-cell structures, boundary consistency, and separation between adjacent nuclei, especially in the highlighted challenging regions.

To rule out the confounding effect of segmentation head design and verify that our performance gains originate from the masked-diffusion pretrained representations rather than task-specific head architecture, we conduct an additional controlled comparison with a unified multi-scale segmentation head shared across all methods. In this setting, multi-scale features are extracted from hierarchical Transformer layers of each ViT baseline, aligned to match the resolution hierarchy of the CMD decoder, and fed into the identical convolutional decoder head. Full quantitative results and implementation details are provided in Appendix D. Under this fair unified-head setting, CMD consistently and significantly outperforms all ViT-based pathology foundation models across all three cell-level benchmarks, especially on boundary-sensitive BF1 metrics. This confirms that the superiority of CMD is primarily driven by our convolutional masked-diffusion pretraining paradigm, which learns morphology-aware dense representations better suited for fine-grained pathology prediction.

Overall, these results show that CMD is an effective frozen backbone for pathology dense prediction. Its convolutional architecture and masked-diffusion pretraining make it less dependent on high input resolution while maintaining strong morphology-aware segmentation performance.

4.3 State-of-the-Art Parameter-Efficient Few-Shot Adaptation

We further evaluate few-shot transfer for cell-level dense prediction on CPM-17 and TNBC under 1-shot, 5-shot, and 10-shot settings. This setting tests whether pretrained representations can be adapted with very limited annotations. We compare CMD-L with frozen pathology foundation model baselines, TransUNet with frozen UNI2-h features, and an end-to-end ViT-Adapter baseline. For CMD-L, the pretrained ConvNeXt masked-diffusion backbone remains frozen, and only a segmentation head with 7M parameters [44] is optimized.

As shown in Table 5, frozen pathology foundation models provide limited few-shot segmentation performance, indicating that global pathology representations are not sufficient for dense cell-level prediction. TransUNet improves slightly in some settings, but still struggles to recover fine-grained nuclei structures from frozen image-level features. ViT-Adapter achieves strong results when more labels are available, but requires end-to-end optimization of a much larger number of trainable parameters.

Table 4: Segmentation performance and parameter statistics on CPM-15, CPM-17, and TNBC. The table compares frozen pathology foundation model backbones, end-to-end segmentation baselines, and CMD. Dataset headers report the input resolutions used by different model families. Results are reported as mean Dice/BF1 with 95% confidence intervals. Best in **bold**; second best underlined.

Method	Frozen Params. (M)	Seg. Head Params. (M)	cpm15 ViT/16:384; ViT/14:392; CMD:256 E2E Model:384		cpm17 ViT/16:496; ViT/14:490; CMD:256 E2E Model:496		TNBC ViT/16:512; ViT/14:504; CMD:256 E2E Model:512	
			Dice	BF1	Dice	BF1	Dice	BF1
Pre-trained ViT-based Pathology Foundation Models								
PathOrchestra	303	7.06	0.737 [0.658, 0.822]	0.467 [0.363, 0.601]	0.861 [0.837, 0.882]	0.584 [0.544, 0.624]	0.819 [0.781, 0.855]	0.482 [0.426, 0.534]
CONCHv1.5	306	7.06	0.748 [0.670, 0.824]	0.457 [0.382, 0.550]	0.858 [0.829, 0.885]	0.581 [0.520, 0.643]	0.826 [0.793, 0.859]	0.497 [0.446, 0.549]
CONCH	90.4	5.88	0.722 [0.608, 0.825]	0.428 [0.311, 0.567]	0.867 [0.841, 0.889]	0.620 [0.572, 0.669]	0.816 [0.785, 0.847]	0.455 [0.423, 0.502]
Prov-Gigapath	1100	9.42	0.759 [0.680, 0.842]	0.503 [0.403, 0.631]	0.871 [0.851, 0.888]	0.628 [0.587, 0.661]	0.828 [0.795, 0.860]	0.509 [0.460, 0.560]
Hibou	304	7.06	0.794 [0.722, 0.874]	0.611 [0.520, 0.730]	0.875 [0.853, 0.896]	0.658 [0.613, 0.715]	0.841 [0.812, 0.866]	0.570 [0.531, 0.619]
H-Optimus-0	1100	9.42	0.781 [0.700, 0.868]	0.584 [0.467, 0.733]	0.881 [0.862, 0.899]	0.671 [0.631, 0.717]	0.843 [0.808, 0.869]	0.565 [0.499, 0.622]
H-Optimus-1	1100	9.42	0.773 [0.711, 0.846]	0.562 [0.487, 0.664]	0.879 [0.857, 0.896]	0.665 [0.628, 0.700]	0.833 [0.805, 0.859]	0.534 [0.479, 0.576]
Kaiko	304	7.06	0.758 [0.663, 0.855]	0.529 [0.402, 0.687]	0.882 [0.861, 0.899]	0.674 [0.632, 0.718]	0.838 [0.807, 0.867]	0.554 [0.504, 0.602]
Midnight-12k	1100	9.42	0.710 [0.650, 0.775]	0.403 [0.360, 0.460]	0.845 [0.821, 0.865]	0.527 [0.502, 0.558]	0.798 [0.761, 0.834]	0.413 [0.363, 0.460]
MUSK	675	7.06	0.651 [0.614, 0.693]	0.272 [0.243, 0.309]	0.737 [0.706, 0.766]	0.370 [0.334, 0.409]	0.693 [0.666, 0.720]	0.275 [0.242, 0.313]
Phikon	86.4	5.88	0.759 [0.686, 0.837]	0.517 [0.439, 0.618]	0.871 [0.844, 0.897]	0.630 [0.568, 0.698]	0.828 [0.800, 0.856]	0.491 [0.443, 0.555]
Phikon-v2	303	7.06	0.754 [0.678, 0.841]	0.520 [0.424, 0.648]	0.874 [0.854, 0.893]	0.647 [0.609, 0.688]	0.824 [0.800, 0.849]	0.469 [0.450, 0.494]
UNI	303	7.06	0.738 [0.662, 0.813]	0.451 [0.398, 0.517]	0.867 [0.838, 0.892]	0.606 [0.543, 0.660]	0.820 [0.790, 0.850]	0.467 [0.439, 0.500]
UNI2-h	681	9.42	0.767 [0.703, 0.840]	0.556 [0.483, 0.652]	0.873 [0.852, 0.890]	0.643 [0.605, 0.673]	0.833 [0.807, 0.855]	0.535 [0.490, 0.569]
Virchow	631	8.24	0.770 [0.701, 0.853]	0.577 [0.490, 0.695]	0.876 [0.859, 0.890]	0.659 [0.631, 0.693]	0.839 [0.817, 0.861]	0.542 [0.512, 0.568]
Virchow2	631	8.24	0.725 [0.643, 0.827]	0.449 [0.356, 0.577]	0.863 [0.832, 0.887]	0.608 [0.541, 0.666]	0.821 [0.801, 0.841]	0.469 [0.433, 0.507]
H0-mini	85.7	5.88	0.776 [0.704, 0.853]	0.563 [0.477, 0.677]	0.878 [0.859, 0.896]	0.675 [0.644, 0.715]	0.841 [0.813, 0.868]	0.557 [0.508, 0.601]
End-to-End Segmentation Models								
nnUNet	-	12.19	0.791 [0.765, 0.820]	0.563 [0.506, 0.607]	0.875 [0.859, 0.898]	0.657 [0.568, 0.730]	0.803 [0.778, 0.822]	0.455 [0.302, 0.533]
TransUNet	-	105.28	0.809 [0.771, 0.850]	0.649 [0.640, 0.657]	**0.897 [0.882, 0.914]**	0.769 [0.738, 0.806]	0.843 [0.822, 0.860]	0.615 [0.578, 0.654]
ViT-Adapter	-	133.52	0.876 [0.830, 0.924]	0.853 [0.817, 0.902]	0.872 [0.839, 0.898]	0.668 [0.571, 0.737]	0.854 [0.832, 0.876]	0.661 [0.620, 0.714]
SAM Pre-trained ViT-based Pathology Foundation Models								
VISTA [43]	168	Zero-shot	0.208 [0.149, 0.266]	0.020 [0.000, 0.048]	0.481 [0.366, 0.560]	0.011 [0.002, 0.022]	0.551 [0.501, 0.607]	0.049 [0.017, 0.079]
Pre-trained Purely Convolutional Pathology Foundation Models								
CMD-L(ours)	602	Linear Probe	0.676 [0.643, 0.709]	0.627 [0.508, 0.746]	0.841 [0.811, 0.871]	0.861 [0.806, 0.907]	0.807 [0.774, 0.841]	0.826 [0.802, 0.851]
CMD-L(ours) + SegHead: Fig. B2(A)	602	7.15	0.864 [0.845, 0.882]	0.911 [0.888, 0.933]	0.865 [0.827, 0.902]	**0.875 [0.807, 0.942]**	0.854 [0.824, 0.891]	**0.874 [0.829, 0.943]**
CMD-L(ours) + SegHead: Fig. B2(B)	602	21.21	**0.889 [0.859, 0.925]**	**0.947 [0.935, 0.961]**	0.880 [0.847, 0.908]	0.866 [0.800, 0.925]	**0.854 [0.835, 0.874]**	0.838 [0.817, 0.868]

In contrast, CMD-L achieves competitive or superior performance with only 7M trainable parameters. It performs especially well in the most label-scarce settings and maintains strong Dice and Precision as the number of shots increases. These results show that CMD-L learns morphology-aware dense features that are both label-efficient and parameter-efficient for few-shot cell-level segmentation.

4.4 Scaling Behavior in Pretraining Duration and Model Capacity

We study CMD scaling from two aspects: masked-diffusion pretraining duration and diffusion backbone capacity. Unlike the few-shot setting, this analysis trains the downstream segmentation head with the full training split and evaluates on the corresponding test split, allowing us to assess representation quality under the standard full-data protocol.

Table 6 shows that CMD generally benefits from longer pretraining. Performance improves or remains stable as the number of pretraining steps increases, indicating that masked-diffusion pretraining continues to produce transferable dense representations rather than overfitting to the reconstruction objective. Some dataset-specific fluctuations appear across intermediate checkpoints, but later checkpoints do not show systematic degradation.

We further compare CMD-B and CMD-L in Table 7. Increasing backbone capacity consistently improves Dice across CPM-17 and TNBC, while maintaining comparable Precision. This suggests that

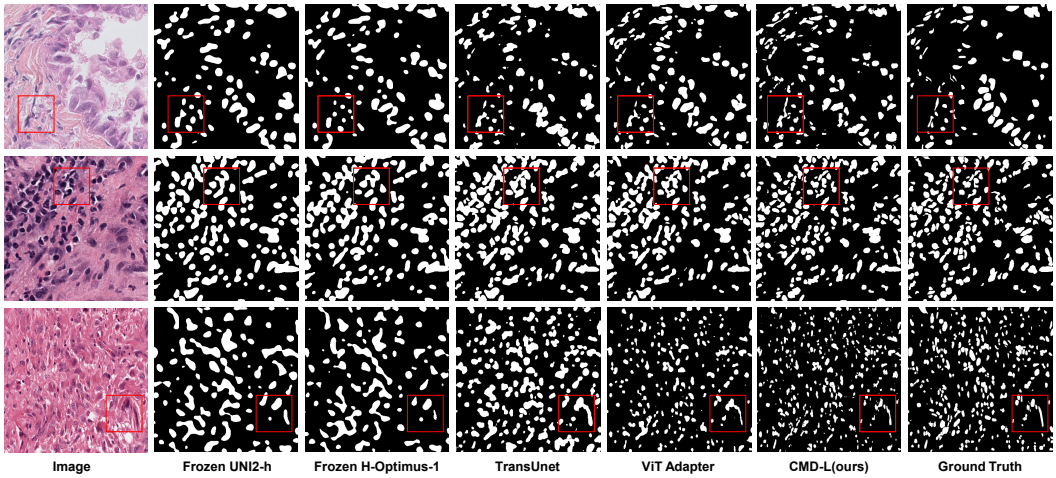

Figure 2: Qualitative comparison under the frozen-backbone dense prediction setting. CMD-L produces more complete and boundary-consistent cell masks than frozen ViT-based pathology foundation models and supervised dense prediction baselines, especially in crowded or ambiguous regions highlighted by red boxes. From top to bottom are the TNBC, CPM17, and CPM15 datasets.

Table 5: Few-shot segmentation performance and parameter statistics on CPM-17 and TNBC. The table compares frozen pathology foundation models, TransUNet, end-to-end ViT-Adapter, and CMD-L under 1-shot, 5-shot, and 10-shot settings. Results are reported as mean Dice/Precision with 95% confidence intervals.

Condition	Frozen UNI2-h		Frozen H-opt-1		TransUNet		ViT-Adapter		Frozen CMD-L(ours)	
	Dice	Precision	Dice	Precision	Dice	Precision	Dice	Precision	Dice	Precision
Trainable/Total Params. (M)	9/690		8/1144		23/704		364/364		7/609	
cpm17 - 1-Shot	0.463 [0.454, 0.472]	0.507 [0.490, 0.528]	0.468 [0.458, 0.479]	0.493 [0.485, 0.502]	0.479 [0.464, 0.492]	0.499 [0.485, 0.514]	0.714 [0.683, 0.749]	0.695 [0.661, 0.740]	0.769 [0.696, 0.832]	0.784 [0.718, 0.845]
cpm17 - 5-Shot	0.468 [0.457, 0.477]	0.501 [0.487, 0.521]	0.467 [0.456, 0.480]	0.497 [0.483, 0.512]	0.477 [0.463, 0.489]	0.499 [0.487, 0.511]	0.812 [0.694, 0.880]	0.852 [0.806, 0.904]	0.811 [0.735, 0.882]	0.893 [0.863, 0.918]
cpm17 - 10-Shot	0.464 [0.453, 0.474]	0.500 [0.487, 0.518]	0.472 [0.460, 0.485]	0.508 [0.489, 0.529]	0.476 [0.463, 0.488]	0.497 [0.485, 0.509]	0.819 [0.725, 0.881]	0.832 [0.780, 0.894]	0.820 [0.744, 0.887]	0.896 [0.875, 0.914]
TNBC - 1-Shot	0.497 [0.477, 0.519]	0.506 [0.473, 0.534]	0.486 [0.477, 0.497]	0.500 [0.484, 0.516]	0.478 [0.469, 0.489]	0.495 [0.482, 0.512]	0.498 [0.470, 0.529]	0.501 [0.464, 0.539]	0.714 [0.618, 0.817]	0.865 [0.817, 0.917]
TNBC - 5-Shot	0.493 [0.482, 0.506]	0.509 [0.492, 0.529]	0.490 [0.480, 0.500]	0.508 [0.488, 0.529]	0.477 [0.468, 0.488]	0.484 [0.470, 0.500]	0.800 [0.773, 0.830]	0.808 [0.783, 0.849]	0.792 [0.761, 0.851]	0.845 [0.800, 0.880]
TNBC - 10-Shot	0.497 [0.491, 0.504]	0.514 [0.499, 0.525]	0.492 [0.484, 0.503]	0.510 [0.495, 0.524]	0.476 [0.468, 0.488]	0.910 [0.879, 0.953]	0.828 [0.807, 0.852]	0.837 [0.796, 0.916]	0.819 [0.790, 0.870]	0.845 [0.807, 0.877]

larger ConvNeXt diffusion backbones improve region-level segmentation quality without increasing false positives. Architecture details of CMD-B and CMD-L are provided in Appendix B.2.

Overall, CMD shows favorable scaling behavior with both pretraining duration and model capacity. Although not a full scaling-law analysis, these results indicate that masked-diffusion convolutional pretraining can continue to benefit from more computation and larger backbones for cell-level dense prediction.

Table 6: Effect of CMD-L pretraining duration on full-data downstream segmentation. Results are Dice scores reported as mean with 95% confidence interval.

Dataset	Training Steps							
	10k	**20k**	**30k**	**40k**	**50k**	**60k**	**70k**	**80k**
CPM15	0.851 [0.806, 0.895]	0.858 [0.842, 0.874]	0.774 [0.751, 0.797]	0.859 [0.840, 0.879]	0.868 [0.845, 0.892]	0.866 [0.842, 0.889]	0.865 [0.841, 0.889]	0.864 [0.845, 0.882]
CPM17	0.873 [0.845, 0.903]	0.864 [0.829, 0.901]	0.827 [0.757, 0.899]	0.842 [0.787, 0.895]	0.851 [0.802, 0.899]	0.849 [0.796, 0.902]	0.863 [0.825, 0.902]	0.865 [0.827, 0.902]
TNBC	0.820 [0.779, 0.884]	0.823 [0.775, 0.889]	0.829 [0.790, 0.877]	0.839 [0.802, 0.892]	0.844 [0.810, 0.888]	0.845 [0.814, 0.885]	0.850 [0.817, 0.892]	0.854 [0.824, 0.891]

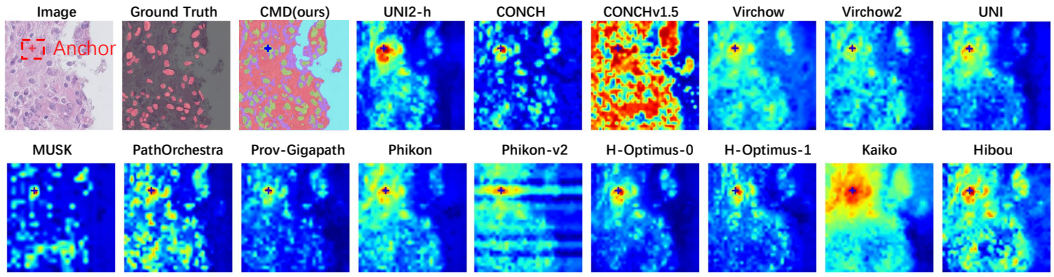

Figure 3: Visualization of cell-level dense representations. For ViT-based pathology foundation models, we select an anchor patch on a nucleus and visualize cosine similarity between the anchor token and all other patch tokens. These models mainly highlight coarse tissue regions and show limited sensitivity to individual cellular structures. For CMD, which does not use patch tokens, we extract dense convolutional features and visualize their K-means (K=4) clustering result. CMD produces a more cell-aware representation that better follows fine nuclei distribution and local morphology.

Table 7: Backbone capacity scaling of CMD at 70k pretraining steps on full-data downstream segmentation. Results are mean with 95% confidence interval.

Dataset	**CMD-B**		**CMD-L**	
	Dice	Precision	Dice	Precision
Params. (M)	217		602	
CPM15	0.852 [0.836, 0.869]	0.843 [0.837, 0.849]	0.865 [0.841, 0.889]	0.852 [0.835, 0.869]
CPM17	0.829 [0.755, 0.902]	0.900 [0.870, 0.928]	0.863 [0.825, 0.902]	0.899 [0.872, 0.924]
TNBC	0.836 [0.806, 0.875]	0.828 [0.797, 0.854]	0.850 [0.817, 0.892]	0.846 [0.812, 0.877]

4.5 Visualization of Cell-Level Dense Representations

We visualize learned representations to examine whether CMD captures fine cellular structures rather than only coarse tissue semantics. As shown in Figure 3, ViT-based models tend to highlight broad tissue regions or coarse contextual patterns, with activations often spreading beyond individual nuclei. In contrast, CMD produces more localized and cell-aware feature clusters that better align with nuclei and fine boundaries. This supports the quantitative results: convolutional masked-diffusion pretraining learns dense, morphology-sensitive representations that are better suited for cell-level prediction.

5 Conclusion and Future Work

In this article, we introduced Masked-Diffusion Convolutional Foundation Models, termed **C**onvNeXt **M**asked-**D**iffusion (**CMD**), for cell-level dense prediction in computational pathology. CMD uses a fully convolutional ConvNeXt-UNet backbone with pixel-space masked-diffusion pretraining to learn dense, morphology-aware representations, while incorporating pathology foundation model conditioning as semantic guidance.

Experiments across multiple pathology benchmarks show that CMD provides strong frozen dense features, outperforming ViT-based pathology foundation models and remaining competitive with state-of-the-art end-to-end segmentation baselines. CMD also demonstrates parameter-efficient few-shot adaptation, favorable scaling behavior, and localized cell-aware representations in visualization analyses.

Future work will explore larger corpora and backbones, multimodal or report-guided conditioning, and whole-slide workflows linking cell-level prediction to slide-level diagnosis and tumor microenvironment analysis.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant No. 82430062, the Shenzhen Engineering Research Centre under grant No. XMHT20230115004, the Beijing Municipal Health Commission Research Ward Excellence Clinical Research Program BR-WEP2024WO32240114 at Beijing Tsinghua Changgung Hospital, and the National High Level Hospital Clinical Research Funding BJ-2024-155 at Beijing Hospital.

References

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

[2] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021.

[3] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[4] Ming Y Lu, Drew FK Williamson, Tiffany Y Chen, Richard J Chen, Matteo Barbieri, and Faisal Mahmood. Data-efficient and weakly supervised computational pathology on whole-slide images. *Nature biomedical engineering*, 5(6):555–570, 2021.

[5] Xitong Ling, Minxi Ouyang, Yizhi Wang, Xinrui Chen, Renao Yan, Hongbo Chu, Junru Cheng, Tian Guan, Sufang Tian, Xiaoping Liu, et al. Agent aggregator with mask denoise mechanism for histopathology whole slide image analysis. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 2795–2803, 2024.

[6] Xiangde Luo, Jinxi Xiang, Yuanfeng Ji, and Ruijiang Li. nnmil: A generalizable multiple instance learning framework for computational pathology. *arXiv preprint arXiv:2511.14907*, 2025.

[7] Jiawen Li, Jiali Hu, Xitong Ling, Renao Yan, Yuxuan Chen, Tian Guan, and Yonghong He. Turning pre-trained vision transformers into end-to-end histopathology whole slide image models for survival prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21046–21056, June 2026.

[8] Richard J Chen, Tong Ding, Ming Y Lu, Drew FK Williamson, Guillaume Jaume, Andrew H Song, Bowen Chen, Andrew Zhang, Daniel Shao, Muhammad Shaban, et al. Towards a general-purpose foundation model for computational pathology. *Nature medicine*, 30(3):850–862, 2024.

[9] Eugene Vorontsov, Alican Bozkurt, Adam Casson, George Shaikovski, Michal Zelechowski, Kristen Severson, Eric Zimmermann, James Hall, Neil Tenenholtz, Nicolo Fusi, et al. A foundation model for clinical-grade computational pathology and rare cancers detection. *Nature medicine*, 30(10):2924–2935, 2024.

[10] Eric Zimmermann, Eugene Vorontsov, Julian Viret, Adam Casson, Michal Zelechowski, George Shaikovski, Neil Tenenholtz, James Hall, David Klimstra, Razik Yousfi, et al. Virchow2: Scaling self-supervised mixed magnification models in pathology. *arXiv preprint arXiv:2408.00738*, 2024.

[11] Fang Yan, Jianfeng Wu, Jiawen Li, Wei Wang, Yirong Chen, Linda Wei, Jiaxuan Lu, Wen Chen, Zizhao Gao, Jianan Li, et al. Pathorchestra: A comprehensive foundation model for computational pathology with over 100 diverse clinical-grade tasks. *npj Digital Medicine*, 8(1):695, 2025.

[12] Alexandre Filiot, Ridouane Ghermi, Antoine Olivier, Paul Jacob, Lucas Fidon, Alice Mac Kain, Charlie Saillard, and Jean-Baptiste Schiratti. Scaling self-supervised learning for histopathology with masked image modeling. *medRxiv*, 2023.

[13] Alexandre Filiot, Paul Jacob, Alice Mac Kain, and Charlie Saillard. Phikon-v2, a large and public feature extractor for biomarker prediction. *arXiv preprint arXiv:2409.09173*, 2024.

[14] Hanwen Xu, Naoto Usuyama, Jaspreet Bagga, Sheng Zhang, Rajesh Rao, Tristan Naumann, Cliff Wong, Zelalem Gero, Javier González, Yu Gu, et al. A whole-slide foundation model for digital pathology from real-world data. *Nature*, 630(8015):181–188, 2024.

[15] Dmitry Nechaev, Alexey Pchelnikov, and Ekaterina Ivanova. Hibou: A family of foundational vision transformers for pathology. *arXiv preprint arXiv:2406.05074*, 2024.

[16] Nanne Aben, Edwin D de Jong, Ioannis Gatopoulos, Nicolas Känzig, Mikhail Karasikov, Axel Lagré, Roman Moser, Joost van Doorn, Fei Tang, et al. Towards large-scale training of pathology foundation models. *arXiv preprint arXiv:2404.15217*, 2024.

[17] Lianghui Zhu, Xitong Ling, Minxi Ouyang, Xiaoping Liu, Tian Guan, Mingxi Fu, Zhiqiang Cheng, Fanglei Fu, Maomao Zeng, Liming Liu, et al. Subspecialty-specific foundation model for intelligent gastrointestinal pathology. *arXiv preprint arXiv:2505.21928*, 2025.

[18] Jiawen Li, Jiali Hu, Xitong Ling, Yongqiang Lv, Yuxuan Chen, Yizhi Wang, Tian Guan, Yifei Liu, and Yonghong He. Stainnet: A special staining self-supervised vision transformer for computational pathology. *arXiv preprint arXiv:2512.10326*, 2025.

[19] Jiabo Ma, Zhengrui Guo, Fengtao Zhou, Yihui Wang, Yingxue Xu, Jinbang Li, Fang Yan, Yu Cai, Zhengjie Zhu, Cheng Jin, et al. A generalizable pathology foundation model using a unified knowledge distillation pretraining framework. *Nature Biomedical Engineering*, pages 1–20, 2025.

[20] Mikhail Karasikov, Joost van Doorn, Nicolas Känzig, Melis Erdal Cesur, Hugo Mark Horlings, Robert Berke, Fei Tang, and Sebastian Otálora. Training state-of-the-art pathology foundation models with orders of magnitude less data. *arXiv preprint arXiv:2504.05186*, 2025.

[21] Saarthak Kapse, Mehmet Aygün, Elijah Cole, Emma Lundberg, Le Song, and Eric P Xing. Genbio-pathfm: A state-of-the-art foundation model for histopathology. *bioRxiv*, pages 2026–03, 2026.

[22] Jiawen Li, Tian Guan, Huijuan Shi, Xitong Ling, Mingxi Fu, Anjia Han, Chao He, and Yonghong He. Alice: Learning a general-purpose pathology foundation model from vision, vision-language, and slide-level experts, 2026.

[23] Zhi Huang, Federico Bianchi, Mert Yuksekgonul, Thomas J Montine, and James Zou. A visual–language foundation model for pathology image analysis using medical twitter. *Nature medicine*, 29(9):2307–2316, 2023.

[24] Ming Y Lu, Bowen Chen, Drew FK Williamson, Richard J Chen, Ivy Liang, Tong Ding, Guillaume Jaume, Igor Odintsov, Long Phi Le, Georg Gerber, et al. A visual-language foundation model for computational pathology. *Nature medicine*, 30(3):863–874, 2024.

[25] Tong Ding, Sophia J Wagner, Andrew H Song, Richard J Chen, Ming Y Lu, Andrew Zhang, Anurag J Vaidya, Guillaume Jaume, Muhammad Shaban, Ahrong Kim, et al. A multimodal whole-slide foundation model for pathology. *Nature medicine*, pages 1–13, 2025.

[26] Jinxi Xiang, Xiyue Wang, Xiaoming Zhang, Yinghua Xi, Feyisope Eweje, Yijiang Chen, Yuchen Li, Colin Bergstrom, Matthew Gopaulchan, Ted Kim, et al. A vision–language foundation model for precision oncology. *Nature*, 638(8051):769–778, 2025.

[27] Jiawen Li, Qiehe Sun, Renao Yan, Yizhi Wang, Yuqiu Fu, Yani Wei, Tian Guan, Huijuan Shi, Yonghong He, and Anjia Han. Diagnostic text-guided representation learning in hierarchical classification for pathological whole slide image. *Medical Image Analysis*, 109:103894, 2026.

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[29] Shangke Liu, Mohamed Amgad, Deeptej More, Muhammad A Rathore, Roberto Salgado, and Lee AD Cooper. A panoptic segmentation dataset and deep-learning approach for explainable scoring of tumor-infiltrating lymphocytes. *NPJ Breast Cancer*, 10(1):52, 2024.

[30] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306*, 2021.

[31] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022.

[32] Zixuan Pan, Jianxu Chen, and Yiyu Shi. Masked diffusion as self-supervised representation learner. *arXiv preprint arXiv:2308.05695*, 2023.

[33] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022.

[34] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16133–16142, 2023.

[35] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[36] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, et al. Attention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999*, 2018.

[37] Quoc Dang Vu, Simon Graham, Tahsin Kurc, Minh Nguyen Nhat To, Muhammad Shaban, Talha Qaiser, Navid Alemi Koohbanani, Syed Ali Khurram, Jayashree Kalpathy-Cramer, Tianhao Zhao, et al. Methods for segmentation and classification of digital microscopy tissue images. *Frontiers in bioengineering and biotechnology*, 7:53, 2019.

[38] Andrew Janowczyk and Anant Madabhushi. Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases. *Journal of pathology informatics*, 7(1):29, 2016.

[39] Simon Graham, David Epstein, and Nasir Rajpoot. Dense steerable filter cnns for exploiting rotational symmetry in histology images. *IEEE Transactions on Medical Imaging*, 39(12):4124–4136, 2020.

[40] Peter Naylor, Marick Laé, Fabien Reyal, and Thomas Walter. Segmentation of nuclei in histopathology images by deep regression of the distance map. *IEEE transactions on medical imaging*, 38(2):448–459, 2018.

[41] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

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

[43] Peixian Liang, Songhao Li, Shunsuke Koga, Yutong Li, Zahra Alipour, Yucheng Tang, Daguang Xu, and Zhi Huang. Vista-path: An interactive foundation model for pathology image segmentation and quantitative analysis in computational pathology. *arXiv preprint arXiv:2601.16451*, 2026.

[44] Ziyi Li, Qinye Zhou, Xiaoyun Zhang, Ya Zhang, Yanfeng Wang, and Weidi Xie. Open-vocabulary object segmentation with diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7667–7676, 2023.

[45] Dmitry Nechaev, Alexey Pchelnikov, and Ekaterina Ivanova. Histai: an open-source, large-scale whole slide image dataset for computational pathology. *arXiv preprint arXiv:2505.12120*, 2025.

A Theoretical Overview of ConvNeXt Masked-Diffusion Models

Masked-diffusion pretraining can be viewed as a self-supervised relaxation of denoising diffusion models for dense representation learning. In a standard DDPM, an image $x_0 \sim q(x_0)$ is corrupted by Gaussian noise at timestep t, and a time-conditioned network is trained to predict either the injected noise or the clean image. In contrast, masked diffusion replaces Gaussian corruption with timestep-controlled structural masking. This change removes the requirement that the forward process correspond to a valid generative diffusion chain, but preserves the key representation-learning property: the model must recover the original signal from corruptions of varying difficulty.

Given an unlabeled pathology image patch $x_0 \in \mathbb{R}^{H\times W\times C}$, we divide it into $N = HW/P^2$ non-overlapping patches with patch size P. A timestep $t \sim \mathcal{U}\{1,\dots,T\}$ determines the masking ratio

$$r_t = \frac{t}{T+1}. \tag{1}$$

Let $m_t \in \{0,1\}^{H\times W\times C}$ be a binary patch mask obtained by randomly masking $\lfloor r_t N \rfloor$ patches and broadcasting the patch-level mask to pixels. The corrupted input is

$$x_t = \mathcal{M}(x_0, t) = m_t \odot x_0, \tag{2}$$

where $\odot$ denotes element-wise multiplication. Larger timesteps therefore correspond to stronger structural occlusion rather than stronger Gaussian noise. The model is trained to reconstruct the clean image from the partially observed input:

$$\hat{x}_0 = f_\theta(x_t, t, z_{\mathrm{pfm}}), \tag{3}$$

where f_θ is a ConvNeXt-U-Net masked-diffusion backbone and z_{pfm} denotes an optional frozen pathology foundation model feature.

The ConvNeXt-U-Net architecture provides a locality-preserving alternative to token-based diffusion backbones. Its encoder-decoder structure extracts multi-scale dense features, while skip connections preserve high-resolution morphology. Timestep and pathology conditions are projected into a shared conditioning vector,

$$c = \phi_t(t) + \phi_z(z_{\mathrm{pfm}}), \tag{4}$$

and injected into ConvNeXt blocks through adaptive normalization, e.g.,

$$\mathrm{AdaLN}(u, c) = \gamma(c) \odot \mathrm{LN}(u) + \beta(c), \tag{5}$$

where $\gamma(c)$ and $\beta(c)$ are learned scale and shift parameters. This allows the same convolutional backbone to adapt its reconstruction behavior to both the corruption level and pathology-aware semantic context.

Following the masked diffusion formulation, the pretraining loss compares the reconstruction $\hat{x}_0$ with the original image x_0. A pixel reconstruction objective can be written as

$$\mathcal{L}_{\mathrm{rec}} = \mathbb{E}_{x_0,t}\left[\|x_0 - f_\theta(x_t, t, z_{\mathrm{pfm}})\|_1\right]. \tag{6}$$

To better align reconstruction pretraining with downstream dense prediction, one may also use a structural similarity loss:

$$\mathcal{L}_{\mathrm{SSIM}}(x_0, \hat{x}_0) = \frac{1 - \mathrm{SSIM}(x_0, \hat{x}_0)}{2}, \tag{7}$$

where

$$\mathrm{SSIM}(x, \hat{x}) = \frac{(2\mu_x\mu_{\hat{x}} + c_1)(2\sigma_{x\hat{x}} + c_2)}{(\mu_x^2 + \mu_{\hat{x}}^2 + c_1)(\sigma_x^2 + \sigma_{\hat{x}}^2 + c_2)}. \tag{8}$$

Here μ, σ^2, and $\sigma_{x\hat{x}}$ denote local means, variances, and covariance, while c_1 and c_2 stabilize the division. The overall masked-diffusion objective can therefore be expressed as

$$\mathcal{L}_{\mathrm{CMD}} = \mathbb{E}_{x_0,t}\left[\lambda_1 \|x_0 - \hat{x}_0\|_1 + \lambda_s \mathcal{L}_{\mathrm{SSIM}}(x_0, \hat{x}_0)\right], \tag{9}$$

with λ_1 and λ_s controlling the balance between pixel fidelity and structural consistency.

After pretraining, the generative reconstruction head is not used for sampling. Instead, the frozen ConvNeXt masked-diffusion model serves as a dense feature extractor. Multi-scale decoder activations are collected at a fixed timestep, resized to a common resolution, and passed to a lightweight segmentation head. Thus, the model transfers the structural representations learned from unlabeled masked reconstruction to cell-level dense prediction with limited annotation.

B Experimental Details

B.1 Datasets

We use unlabeled pathology patches sourced from the public HistAI collection [45] for masked-diffusion pretraining and evaluate the learned frozen representations on downstream cell-level dense prediction datasets. Figure B1 summarizes the organ-source distribution of the unlabeled pretraining corpus. We strictly confirm that the pretraining dataset contains no overlapping samples with downstream evaluation data to eliminate data leakage concerns.

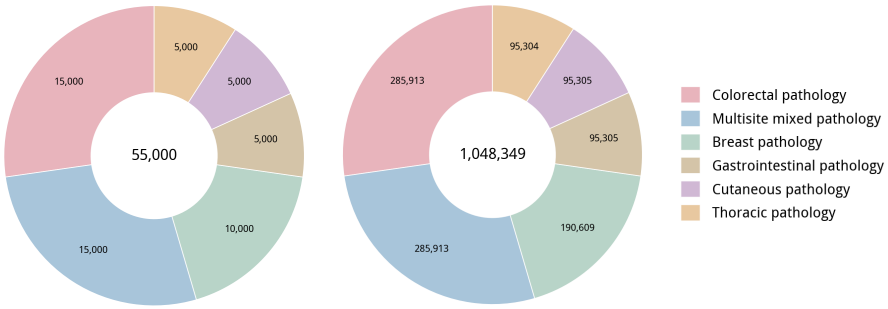

Figure B1: Organ-source distribution of unlabeled pathology images used for masked-diffusion pretraining. The left donut chart shows the 55,000-image method-development corpus, and the right donut chart shows the 1,048,349-image large-scale experimental corpus.

B.2 ConvNeXt Masked-Diffusion Architecture Details

Table B1 summarizes the architectural differences between CMD-B and CMD-L, together with their shared ConvNeXt masked-diffusion design. The two variants use the same U-shaped topology and differ mainly in model width. CMD-L increases the number of channels at each stage, providing higher representation capacity while preserving the same resolution hierarchy, block allocation, conditioning mechanism, and downstream feature extraction protocol.

B.3 Downstream Segmentation Head

After pretraining, the CMD diffusion backbone is frozen and used as a dense feature extractor. We evaluate two downstream segmentation heads, as illustrated in Fig. B2. The first is a lightweight segmentation head with 7.15M trainable parameters, corresponding to the setting reported in Table 5. The second is a stronger SOTA-level segmentation head with 21.21M trainable parameters, also reported in Table 5.

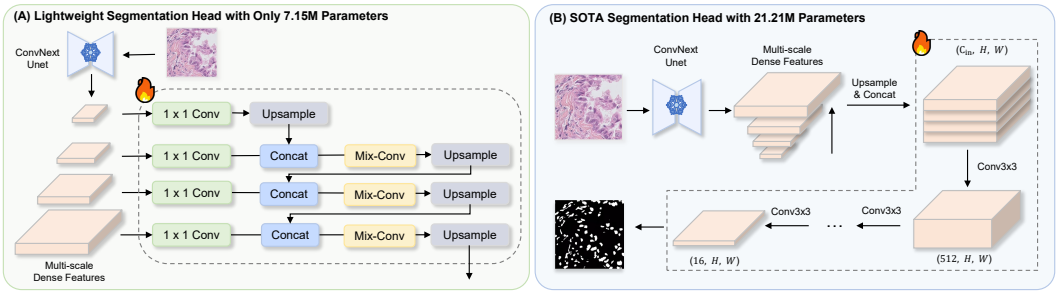

Figure B2: Two downstream segmentation heads built on frozen CMD dense features. (A) The lightweight segmentation head has 7.15M trainable parameters and fuses multi-scale decoder features in a top-down manner. (B) The SOTA-level segmentation head has 21.21M trainable parameters and uses a convolutional decoder after multi-scale feature concatenation.

For the lightweight head in Fig. B2(A), we follow the visual encoder design of Li et al. [44]. Multi-scale decoder features are extracted from selected frozen diffusion decoder stages. Each feature is

Table B1: Architecture details of CMD-B and CMD-L. Variant-specific settings are listed at the top, while shared architectural components are grouped below.

Component	**CMD-B**	**CMD-L**
Base channel width	256	512
Channel hierarchy	$256, 256, 256, 512, 512, 1024$	$512, 512, 512, 1024, 1024, 2048$
Trainable parameters	$\sim$130.75M	$\sim$516.71M
Shared ConvNeXt Masked-Diffusion Design		
Input/output	RGB pathology image input and RGB reconstruction target at 256×256 resolution.	
Backbone topology	ConvNeXt-style U-Net with five downsampling stages, a bottleneck, and five mirrored decoder stages.	
Resolution hierarchy	$256^2 \rightarrow 128^2 \rightarrow 64^2 \rightarrow 32^2 \rightarrow 16^2 \rightarrow 8^2$, followed by symmetric upsampling back to 256^2.	
Encoder depth	The five encoder stages contain $1, 2, 3, 2, 2$ ConvNeXt blocks, respectively.	
Bottleneck depth	The lowest-resolution 8×8 bottleneck contains 6 ConvNeXt blocks for global tissue-context aggregation.	
Decoder depth	The five decoder stages contain $2, 2, 3, 2, 1$ ConvNeXt blocks, respectively.	
Total ConvNeXt blocks	26 blocks in total.	
ConvNeXt block	7×7 depthwise convolution, LayerNorm, pointwise MLP, GELU, GRN, residual connection, and adaptive modulation.	
MLP expansion ratio	3.	
Down/up sampling	Pixel-unshuffle downsampling and pixel-shuffle upsampling.	
Skip connection	Encoder features are concatenated with decoder features at the same resolution and compressed by a 1×1 convolution.	
Conditioning	Diffusion timestep embedding and frozen pathology feature are projected to each stage dimension and fused by addition.	
Condition injection	Adaptive LayerNorm-Zero modulation is applied in every ConvNeXt block.	
Feature extraction	During downstream dense prediction, the reconstruction output is discarded and multi-scale decoder features are reused as frozen dense representations.	

first projected to a unified dimension $d = 256$ by a 1×1 convolution. Starting from the coarsest scale, features are progressively upsampled, concatenated with the next finer feature map, and blended by a Mix-Conv module. Each Mix-Conv contains two 3×3 convolutions with a residual connection and conditional batch normalization. The final fused feature is passed to a lightweight 1×1 convolutional segmentation head.

For the SOTA-level head in Fig. B2(B), dense features from selected decoder blocks are resized to the target segmentation resolution and concatenated along the channel dimension, producing a feature map of shape (C_{in}, H, W). In the FCDM-L setting, $C_{\text{in}} = 4096$ using decoder blocks $[1, 3, 5, 6, 8, 9]$ at diffusion step $t = 50$. The head first applies a 3×3 convolution with batch normalization and ReLU to reduce the channel dimension to 512, followed by four convolutional decoder blocks with output channels 256, 128, 64, and 16. Each block contains two 3×3 Conv-BN-ReLU layers and preserves spatial resolution. A final 3×3 convolution maps the 16-channel feature map to segmentation logits.

B.4 Hyperparameters and Implementation Details of Method-Section

We summarize the configuration used in our method-section. Table B2 details the architectural specifications of the compared backbones. To keep the comparison controlled, ConvNeXt-U-Net, Attention U-Net, and DiT are trained with the same masked-diffusion objective and the same frozen pathology foundation model conditioning, with unified pretraining and downstream training settings summarized in Table B3. The ConvNeXt-U-Net model is our default backbone, while Attention U-Net-B and DiT-B serve as architecture ablations.

Table B2: Backbone configurations for method-section masked-diffusion models.

Configuration	**ConvNeXt-U-Net**	**Attention U-Net**	**DiT**
Backbone type	ConvNeXt-style U-Net	Hybrid-style U-Net	Transformer
Default input space	Pixel space	Pixel / VAE latent	VAE latent
Input size	256×256	256×256	256×256
Parameters	$\sim$130.75M	$\sim$132.65M	$\sim$131M

Table B3: Shared masked-diffusion and downstream training settings of Method-Section.

Configuration	**Setting**
Pretraining data	55,000 unlabeled pathology images
Diffusion timesteps	$T = 1000$
Pixel-space mask	8×8 patches, producing a 32×32 mask grid
VAE-space mask	4×4 latent patches, producing an 8×8 mask grid
Optimizer	AdamW, weight decay 0
Pretraining schedule	50,000 steps, learning rate 3×10^{-5}
EMA	0.9999
Pathology conditioning	Frozen UNI feature, 1024 dimensions
VAE variant	SD-VAE with 4 latent channels and scaling factor 0.18215
Downstream protocol	Linear Probe
Downstream timestep	$t = 50$
Downstream training	150 epochs, AdamW, learning rate 10^{-3}, cosine schedule
Downstream loss	Cross-entropy plus Dice loss

C Comparison with Test-Time Training

We further compare CMD-L with Test-Time Training (TTT) on CPM-17 and TNBC. TTT relies on test-time optimization, while CMD-L uses a frozen pretrained diffusion backbone and performs standard feed-forward inference with downstream segmentation heads. As shown in Table C1, CMD-L achieves performance close to TTT without test-time adaptation, **highlighting the effectiveness of the learned frozen dense representation**.

Table C1: Comparison between Test-Time Training (TTT) and CMD-L on CPM-17 and TNBC. CMD-L is evaluated with two downstream segmentation heads from Fig. B2. Results are reported as mean Dice/Precision with 95% confidence intervals.

Method	**CPM-17**		**TNBC**	
	Dice	Precision	Dice	Precision
CMD-L(ours) + SegHead: Fig. B2(A)	0.865 [0.827, 0.902]	0.901 [0.876, 0.925]	0.854 [0.824, 0.891]	0.851 [0.822, 0.876]
CMD-L(ours) + SegHead: Fig. B2(B)	0.880 [0.847, 0.908]	0.883 [0.860, 0.920]	0.854 [0.835, 0.874]	0.830 [0.802, 0.858]
CMD-L(ours) + TTT + SegHead: Fig. B2(B)	0.885 [0.862, 0.908]	0.892 [0.866, 0.924]	0.861 [0.842, 0.880]	0.860 [0.823, 0.894]

D Unified Multi-Scale Segmentation Head for Fair Baseline Comparison

To address core concerns regarding whether the performance gains of CMD originate from its masked-diffusion pretraining paradigm or from task-specific segmentation head design, and to further ensure the fairness of baseline comparisons, we conduct a controlled ablation study where **all compared methods share an identical multi-scale segmentation head architecture**. This experiment isolates the quality of frozen pretrained representations by eliminating head-design confounders across all baselines.

Table D1: Cell-level dense prediction under the frozen-backbone setting with a unified multi-scale segmentation head. Each entry reports the mean metric followed by its 95% confidence interval. Best result in **bold**; second best underlined.

Method	CPM-15			CPM-17			TNBC		
	Dice	Pre	BF1	Dice	Pre	BF1	Dice	Pre	BF1
CMD-L (ours)	**0.889** [0.859, 0.924]	**0.885** [0.853, 0.925]	**0.947** [0.935, 0.961]	**0.880** [0.847, 0.908]	**0.883** [0.860, 0.919]	**0.866** [0.800, 0.925]	**0.854** [0.835, 0.874]	**0.830** [0.802, 0.858]	**0.838** [0.817, 0.867]
UNI-v1	0.725 [0.655, 0.794]	0.715 [0.666, 0.761]	0.435 [0.399, 0.482]	0.834 [0.796, 0.866]	0.817 [0.769, 0.853]	0.501 [0.430, 0.558]	0.792 [0.759, 0.825]	0.748 [0.711, 0.783]	0.378 [0.334, 0.414]
UNI-v2	0.758 [0.698, 0.820]	0.734 [0.683, 0.786]	0.502 [0.471, 0.544]	0.847 [0.821, 0.868]	0.842 [0.809, 0.868]	0.554 [0.517, 0.589]	0.809 [0.787, 0.833]	0.776 [0.752, 0.796]	0.450 [0.424, 0.485]
CONCH-v1	0.732 [0.664, 0.804]	0.710 [0.650, 0.774]	0.456 [0.403, 0.525]	0.836 [0.805, 0.865]	0.848 [0.809, 0.881]	0.521 [0.470, 0.576]	0.796 [0.756, 0.829]	0.780 [0.770, 0.791]	0.400 [0.364, 0.433]
CONCH-v1.5	0.729 [0.665, 0.800]	0.713 [0.650, 0.786]	0.458 [0.408, 0.526]	0.843 [0.814, 0.868]	0.827 [0.793, 0.853]	0.522 [0.480, 0.563]	0.802 [0.769, 0.837]	0.785 [0.734, 0.827]	0.439 [0.387, 0.494]
Virchow-v1	0.752 [0.692, 0.821]	0.752 [0.679, 0.842]	0.526 [0.489, 0.577]	0.849 [0.825, 0.868]	0.839 [0.811, 0.865]	0.556 [0.522, 0.589]	0.817 [0.794, 0.840]	0.788 [0.763, 0.815]	0.470 [0.449, 0.495]
Virchow-v2	0.759 [0.690, 0.839]	0.740 [0.669, 0.836]	0.513 [0.440, 0.613]	0.848 [0.814, 0.876]	0.838 [0.789, 0.880]	0.567 [0.499, 0.625]	0.815 [0.792, 0.837]	0.779 [0.751, 0.802]	0.467 [0.443, 0.489]
Phikon	0.731 [0.659, 0.804]	0.721 [0.661, 0.780]	0.452 [0.398, 0.521]	0.834 [0.801, 0.864]	0.824 [0.785, 0.853]	0.495 [0.444, 0.542]	0.804 [0.779, 0.829]	0.780 [0.758, 0.800]	0.431 [0.408, 0.456]
Phikon-v2	0.726 [0.666, 0.793]	0.719 [0.654, 0.797]	0.436 [0.386, 0.502]	0.835 [0.803, 0.863]	0.822 [0.783, 0.855]	0.504 [0.454, 0.551]	0.794 [0.766, 0.821]	0.769 [0.752, 0.782]	0.400 [0.377, 0.424]
GigaPath	0.731 [0.668, 0.794]	0.715 [0.666, 0.763]	0.440 [0.407, 0.483]	0.838 [0.807, 0.864]	0.831 [0.794, 0.861]	0.515 [0.468, 0.562]	0.802 [0.773, 0.831]	0.773 [0.742, 0.801]	0.427 [0.393, 0.451]
Hibou-L	0.767 [0.697, 0.846]	0.746 [0.677, 0.830]	0.540 [0.463, 0.643]	0.856 [0.829, 0.882]	0.861 [0.827, 0.893]	0.589 [0.540, 0.638]	0.820 [0.792, 0.846]	0.804 [0.784, 0.823]	0.486 [0.451, 0.530]
H-Optimus-0	0.762 [0.695, 0.833]	0.754 [0.693, 0.817]	0.521 [0.476, 0.579]	0.858 [0.826, 0.881]	0.842 [0.803, 0.873]	0.587 [0.534, 0.627]	0.820 [0.789, 0.848]	0.788 [0.758, 0.814]	0.477 [0.432, 0.516]
H-Optimus-1	0.756 [0.697, 0.823]	0.757 [0.686, 0.846]	0.515 [0.478, 0.564]	0.858 [0.831, 0.879]	0.851 [0.821, 0.876]	0.593 [0.558, 0.619]	0.815 [0.790, 0.839]	0.786 [0.770, 0.804]	0.469 [0.441, 0.503]
Kaiko-ViT-L/14	0.768 [0.700, 0.841]	0.760 [0.694, 0.831]	0.533 [0.466, 0.621]	0.855 [0.827, 0.878]	0.851 [0.822, 0.881]	0.587 [0.546, 0.632]	0.814 [0.789, 0.839]	0.791 [0.778, 0.805]	0.460 [0.423, 0.501]
Midnight-12k	0.742 [0.686, 0.803]	0.731 [0.677, 0.788]	0.473 [0.447, 0.509]	0.842 [0.814, 0.864]	0.825 [0.792, 0.845]	0.533 [0.488, 0.577]	0.804 [0.779, 0.829]	0.770 [0.746, 0.794]	0.428 [0.404, 0.459]
MUSK	0.686 [0.619, 0.752]	0.690 [0.637, 0.740]	0.318 [0.277, 0.370]	0.741 [0.712, 0.768]	0.730 [0.697, 0.758]	0.371 [0.345, 0.395]	0.713 [0.694, 0.737]	0.690 [0.661, 0.727]	0.289 [0.264, 0.312]
PathOrchestra	0.717 [0.654, 0.778]	0.701 [0.657, 0.742]	0.398 [0.374, 0.429]	0.826 [0.799, 0.850]	0.811 [0.781, 0.840]	0.459 [0.429, 0.487]	0.792 [0.753, 0.829]	0.774 [0.724, 0.812]	0.411 [0.360, 0.453]

The segmentation head used across all methods follows the exact architecture of the SOTA-level head illustrated in Fig. B2(B) in Appendix B.3. For ViT-based pathology foundation models, we select output tokens from Transformer layers that correspond to the hierarchical resolution levels of the CMD-L decoder. These multi-scale token maps are reshaped into 2D feature grids, bilinearly upsampled to a unified spatial resolution, and concatenated along the channel dimension. The concatenated feature volume is then reduced via a 3×3 convolution with batch normalization and ReLU activation, followed by the same convolutional decoder blocks and final prediction layer used for CMD-L. For CMD-L, we keep the multi-scale decoder feature extraction protocol unchanged from the main paper. Throughout all experiments, the full backbone of every method remains frozen, and only the shared segmentation head is trained under identical optimization configurations (optimizer, learning rate schedule, loss function, and training epochs).

Table D1 presents the results under this unified head setting. CMD-L consistently achieves significantly higher Dice, Precision and BF1 scores across all three cell-level segmentation benchmarks compared to every ViT-based pathology foundation model. The particularly large performance gap in boundary-sensitive BF1 metrics directly reflects the advantage of convolutional masked-diffusion pretraining in preserving fine morphological structures and spatial continuity.

These results confirm that the performance superiority of CMD is primarily driven by its pixel-space masked-diffusion convolutional pretraining strategy—which learns morphology-aware dense representations—rather than by differences in segmentation head design. This controlled comparison further validates that purely convolutional architectures with generative masked-diffusion pretraining provide stronger frozen representations for cell-level dense prediction tasks compared to ViT-based pathology foundation models.

E Ablation on Conditioning Pathology Foundation Models

In this section, we evaluate CMD-L with different state-of-the-art pathology foundation models as conditioning inputs to verify the generality of our conditioning framework and examine how stronger semantic priors affect downstream dense prediction performance. All model variants follow the exact

Table E1: Performance of CMD-L with different conditioning pathology foundation models on three cell-level segmentation benchmarks. Each entry reports the mean metric followed by its 95% confidence interval.

Conditioning PFM	CPM-15			CPM-17			TNBC		
	Dice	Precision	BF1	Dice	Precision	BF1	Dice	Precision	BF1
H0-mini (default)	0.889 [0.859, 0.924]	0.885 [0.853, 0.925]	0.947 [0.935, 0.961]	0.880 [0.847, 0.908]	0.883 [0.860, 0.919]	0.866 [0.800, 0.925]	0.854 [0.835, 0.874]	0.830 [0.802, 0.858]	0.838 [0.817, 0.867]
H-optimus-1	0.906 [0.877, 0.940]	0.915 [0.885, 0.950]	0.862 [0.849, 0.881]	0.896 [0.867, 0.921]	0.909 [0.886, 0.938]	0.789 [0.716, 0.846]	0.861 [0.837, 0.884]	0.865 [0.824, 0.899]	0.751 [0.707, 0.807]
UNI	0.902 [0.878, 0.929]	0.924 [0.890, 0.966]	0.844 [0.842, 0.846]	0.894 [0.867, 0.917]	0.900 [0.876, 0.929]	0.783 [0.717, 0.837]	0.868 [0.838, 0.898]	0.868 [0.822, 0.901]	0.764 [0.714, 0.805]
Virchow2	0.913 [0.888, 0.944]	0.906 [0.874, 0.947]	0.873 [0.859, 0.894]	0.888 [0.850, 0.920]	0.904 [0.880, 0.933]	0.772 [0.678, 0.848]	0.863 [0.843, 0.886]	0.861 [0.827, 0.902]	0.752 [0.713, 0.795]

Table F1: Cell-level segmentation performance on PanNuke under the frozen-backbone setting with a unified multi-scale segmentation head. Best result in **bold**; second best underlined.

Method	Dice	BF1
PathOrchestra	0.580	0.611
CONCH-v1.5	0.601	0.632
CONCH	0.595	0.621
GigaPath	0.609	0.654
Hibou-L	0.610	**0.711**
H-Optimus-0	<u>0.620</u>	0.700
H-Optimus-1	0.619	0.691
Kaiko-ViT-L/14	0.619	0.691
Midnight-12k	0.575	0.547
MUSK	0.513	0.333
Phikon	0.597	0.647
Phikon-v2	0.614	0.656
UNI-v1	0.598	0.630
UNI-v2	0.609	0.656
Virchow-v1	0.608	0.660
Virchow-v2	0.609	0.660
CMD-L (ours)	**0.711**	<u>0.705</u>

same pretraining and downstream evaluation protocol, with only the frozen conditioning encoder swapped. The full quantitative results on three cell-level benchmarks are presented in Table E1.

These results demonstrate that CMD-L consistently delivers stable, competitive performance across different mainstream pathology foundation models used for conditioning, verifying the generality and compatibility of our conditioning mechanism.

F Additional Evaluation on PanNuke Dataset

To further verify the cross-tissue generalization performance of the proposed CMD framework, we conduct additional evaluation on the PanNuke dataset. PanNuke is a large-scale multi-tissue pathology benchmark covering diverse human tissue types with high annotation heterogeneity, which provides a more comprehensive and challenging testbed for assessing the generalizability of dense representations across different tissue domains.

We follow the exact same evaluation protocol as the main cross-model comparisons: all backbone models remain frozen, and an identical unified multi-scale segmentation head is trained for every method under consistent optimization configurations to ensure fair comparison. The full segmentation results on PanNuke are reported in Table F1.

The results demonstrate that the performance advantage of CMD-L over existing ViT-based pathology foundation models becomes even more pronounced on this large-scale, multi-tissue and heterogeneous benchmark. This further validates that the dense morphological representations learned by our convolutional masked-diffusion pretraining possess stronger cross-tissue generalization capability for fine-grained pathology segmentation tasks.

Table G1: Ablation results of fixed masking ratios versus the proposed timestep-varying masking strategy across five cell-level segmentation benchmarks. Each entry reports the mean metric followed by its 95% confidence interval. Best result in **bold**; second best underlined.

Masking Strategy	CPM-15		CPM-17		Janowczyk		Kumar		TNBC	
	Dice	BF1	Dice	BF1	Dice	BF1	Dice	BF1	Dice	BF1
25% Fixed	57.31 [54.58, 60.04]	50.90 [46.00, 55.80]	73.39 [66.54, 79.80]	70.78 [55.09, 84.81]	52.45 [51.42, 53.80]	15.28 [9.99, 20.93]	57.51 [45.76, 66.30]	53.78 [17.26, 78.12]	69.77 [64.58, 74.78]	70.42 [60.88, 77.28]
50% Fixed	62.89 [61.51, 64.28]	**58.50** [57.56, 59.44]	71.45 [63.28, 79.18]	63.33 [45.93, 79.82]	52.12 [51.10, 53.26]	12.66 [8.34, 17.09]	59.24 [50.41, 65.33]	56.11 [34.03, 69.40]	68.75 [66.27, 71.94]	64.01 [62.48, 65.35]
75% Fixed	57.89 [50.45, 65.34]	51.96 [43.05, 60.88]	70.41 [62.51, 77.56]	67.14 [49.70, 82.69]	52.36 [51.46, 53.46]	16.03 [10.94, 21.43]	55.02 [50.47, 60.30]	55.16 [48.07, 63.53]	58.66 [56.03, 60.68]	48.05 [35.42, 60.55]
90% Fixed	56.80 [53.29, 60.31]	50.47 [40.56, 60.37]	75.16 [68.85, 80.69]	71.96 [59.80, 82.88]	52.41 [51.68, 53.14]	16.57 [12.73, 20.68]	61.05 [52.32, 67.42]	**63.68** [46.85, 73.16]	**74.18** [69.48, 82.13]	**75.87** [70.75, 84.81]
Timestep-varying	**63.25** [59.34, 67.16]	56.03 [55.41, 56.64]	**76.74** [70.29, 82.39]	**73.45** [62.26, 83.70]	**53.66** [52.71, 54.70]	**18.24** [13.99, 22.46]	**62.76** [49.07, 70.45]	62.25 [26.32, 83.66]	73.51 [68.28, 80.45]	75.63 [66.74, 86.52]

G Ablation on Masking Strategy: Fixed Ratios vs. Timestep-Varying Masking

To validate the core effectiveness of the timestep-controlled masked-diffusion paradigm, we conduct a full pretraining ablation study with four fixed masking ratios (25%, 50%, 75%, and 90%). All experimental settings—including pretraining data, backbone architecture, training steps, and downstream evaluation protocols—are kept strictly identical; only the dynamic timestep-varying masking is replaced by a constant fixed masking ratio throughout training. The full quantitative results across all five benchmarks are presented in Table G1.

Overall, the timestep-varying masking scheme achieves the strongest comprehensive performance across all five benchmarks, outperforming most single fixed-ratio settings on the majority of datasets. In contrast, fixed masking ratios exhibit substantial performance fluctuations across different datasets, with no single constant ratio delivering consistently superior results. This indicates that the optimal masking ratio is highly dataset-dependent, and there is no universally applicable fixed ratio for cross-dataset cell-level dense prediction.

More importantly, the timestep-varying strategy consistently reaches either the best or near-best performance on all five datasets without severe performance degradation on any subset, demonstrating the most robust cross-dataset generalization capability. This property of stable cross-scenario transfer without per-task hyperparameter tuning is particularly valuable for pathology foundation models, which are required to generalize across diverse organs, staining protocols, and scanner platforms in real clinical scenarios.

From a mechanistic perspective, we attribute the superiority of timestep-varying masking to its continuous difficulty curriculum that is inherently aligned with the diffusion formulation. The 1000 timesteps correspond to a smooth transition of masking ratios from nearly 0 to nearly 1, covering the full spectrum of reconstruction difficulties ranging from fine local texture inpainting and mid-sized region completion to global tissue structure reasoning. By randomly sampling reconstruction tasks of varying difficulties during training, the model is forced to simultaneously learn both fine-grained cellular morphology and global tissue organization patterns, yielding multi-grained and more robust representations that cannot be captured by discrete fixed-ratio masked autoencoder schemes.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims are scoped to CMD for cell-level dense prediction and are supported by the proposed masked-diffusion framework, ablations, downstream comparisons, few-shot results, scaling analysis, and visualizations.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the limitations and future extensions to broader dense pathology tasks, larger pretraining corpora, larger backbones, multimodal conditioning, and whole-slide workflows.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper is empirical and mechanistic; it does not present theorems, lemmas, or formal proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper provides sufficient implementation and evaluation details to support reproducibility of the main results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide an anonymized code link in the abstract, and the datasets are publicly available.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the pretraining data, input processing, masking strategy, optimizer, training schedule, downstream heads, losses, datasets, and metrics.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Main results report confidence intervals, and the paper states that bootstrap resampling is used for interval estimation.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: The current draft reports model sizes and training steps, but not GPU type, wall-clock time, memory usage, or total compute.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The study uses pathology image datasets for methodological research and does not involve deployment, identification of individuals, or direct clinical decision making.

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper is foundational and does not include a dedicated broader-impacts discussion; any societal effects are indirect and are not analyzed in the current draft.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The submission does not release a high-risk generative model, scraped dataset, or similarly dual-use asset that would require special safeguards.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the datasets, baseline methods, foundation models, and architectural components used in the experiments. We adhere to their licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release anonymized research code as a new asset, including documentation, environment files, configuration files, and scripts. The released code is accompanied by a license and usage instructions. No new dataset is introduced or redistributed.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The work does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The work does not involve human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not an important, original, or non-standard component of the research method or experiments in this paper.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.