WEAVE: ADAPTIVE PATCHIFICATION FOR GIGAPIXEL WHOLE-SLIDE IMAGES

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformer-based architectures for computer vision, such as Vision Transformers, require patchification, which is a process in which the input image is split into patches and mapped to tokens. In contrast to uniform patchification, adaptive patchification allocates more tokens to information-rich image regions. While adaptive patchification is effective for natural images, extending this approach to high-resolution images, such as gigapixel whole-slide images (WSIs) in histopathology, introduces new challenges. Uniform patchification can produce prohibitively long token sequences, making slide-level aggregation expensive, particularly for quadratic-cost architectures such as Transformers. Adaptive patchification can reduce this cost by allocating tokens according to image content, but highly redundant and long-tailed WSI feature statistics destabilize existing scoring mechanisms. We propose **W**SI **E**nergy-based **A**daptive **V**ision **E**ncoder (**WEAVE**), an adaptive patchification method for gigapixel WSIs. We introduce a Hopfield energy-based scoring function that provides stable importance estimates under WSI feature statistics and a lightweight mLSTM encoder for subquadratic token aggregation. Together, these components enable content-adaptive tokenization at gigapixel scale and achieve competitive or improved performance across five WSI classification benchmarks and much higher efficiency.

1 INTRODUCTION

Whole-slide images (WSIs) usually exceed $50{,}000 \times 50{,}000$ pixels, making them orders of magnitude larger than typical natural images. Common WSI pipelines divide slides into uniform patches and process the resulting patch representations with multiple instance learning (MIL) (Ilse et al., 2018; Lu et al., 2021). However, on a gigapixel scale, uniform patchification can produce a very large number of patches, making pairwise quadratic cost slide-level token aggregation increasingly expensive (Chen et al., 2022; Li et al., 2023). This uniform allocation is also poorly matched to the non-homogeneous distribution of diagnostic relevance within WSIs. Large parts of a slide can contain background, while even within tissue regions, the amount of diagnostically relevant information can vary substantially. Assigning the same resolution to all regions therefore allocates the token budget independently of their diagnostic relevance.

Adaptive patchification offers a natural way to address this inefficiency by allocating finer spatial resolution to informative regions and coarser resolution elsewhere. This raises a central question: how should a given token budget be allocated across a gigapixel image to retain diagnostically relevant information? The goal is therefore not simply to minimize the number of extracted tokens, but to allocate them according to image content. Existing approaches demonstrate this principle for natural images through learned importance prediction and differentiable partitioning (Zhang et al., 2024; Aasan et al., 2026; Yin et al., 2025). However, we find that directly applying Dynamic Adaptive Region Tokenizer's (DART's) (Yin et al., 2025) scoring mechanism to WSIs is unstable due to the highly redundant and long-tailed feature statistics of WSIs. In our experiments, gradients in the original DART formulation occasionally become NaN. Even when gradient clipping prevents numerical failure, the resulting score distributions can become highly concentrated or nearly uniform.

Addressing how to allocate tokens is only one part of the WSI processing problem, as the tokens must still be aggregated into a slide-level representation. Even after adaptive patchification, the number of tokens can remain substantial at gigapixel scale. Recent WSI methods have explored

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

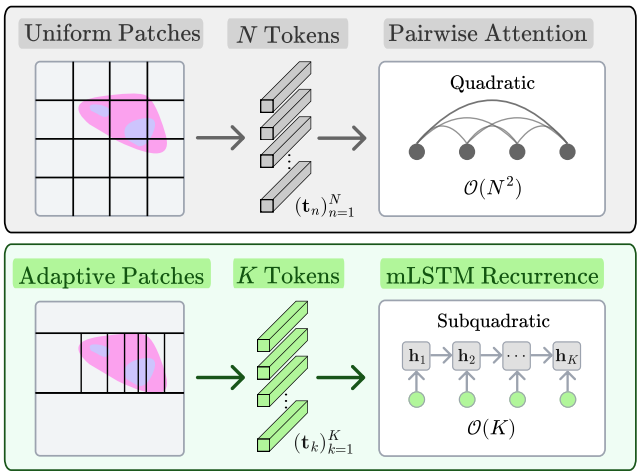


Figure 1: Uniform patchification assigns the same token resolution across the WSI, producing a large token sequence that makes pairwise slide-level aggregation expensive (top). **WEAVE** instead allocates tokens according to image content, concentrating finer resolution in high-importance regions, and aggregates the resulting tokens with a subquadratic mLSTM encoder (bottom).

subquadratic sequence models to reduce the cost of slide-level aggregation (Fang et al., 2024; Zhang et al., 2025), but these approaches typically operate on uniformly sampled or pre-extracted patch tokens. Therefore, we use a lightweight mLSTM encoder based on xLSTM (Beck et al., 2024) to aggregate the adaptively constructed tokens with subquadratic computational cost, providing an efficient alternative to quadratic pairwise aggregation.

Our method. We propose **WEAVE** (**W**SI **E**nergy-based **A**daptive **V**ision **E**ncoder), which combines content-aware adaptive patchification with efficient slide-level aggregation. WEAVE replaces DART's importance scoring mechanism with a prototype-based Hopfield energy score (Ramsauer et al., 2020), while retaining its differentiable quantile partitioning to construct content-dependent rectangular patches. Local WSI features are used for spatial importance scoring and token construction, while these tokens are then aggregated by an mLSTM encoder. Figure 1 illustrates the resulting contrast: rather than assigning a uniform token resolution and subsequently performing expensive pairwise aggregation, WEAVE allocates tokens according to image content and processes them with a subquadratic sequence model. We evaluate the resulting pipeline on five WSI classification benchmarks.

2 Related Work

Adaptive tokenization for vision. Vision models commonly represent images using a regular grid of fixed-size patches, as popularized by the Vision Transformer (ViT) (Dosovitskiy et al., 2020). Several methods reduce tokens after feature extraction, including token pruning (Rao et al., 2021), token learning (Ryoo et al., 2021), and token merging (Bolya et al., 2023). In contrast, adaptive patchification changes how tokens are extracted, allowing tokens to represent spatial regions of different sizes. Methods such as (Zhang et al., 2024) learn content-dependent spatial partitions, while ∂HT (Aasan et al., 2026) formulates patchification as differentiable hierarchical partitioning and DART (Yin et al., 2025) uses learned importance scores and differentiable quantiles to construct adaptively sized patches. Other work explores variable patch scales (Beyer et al., 2023; Havtorn et al., 2023). These methods have primarily been developed and evaluated in natural images, leaving adaptive patchification in gigapixel pathology largely unexplored.

Efficient WSI modeling. Whole-slide image (WSI) analysis commonly decomposes slides into patch-level representations followed by slide-level aggregation using multiple instance learning (MIL). Attention-based approaches such as ABMIL (Ilse et al., 2018), CLAM (Lu et al., 2021), DSMIL (Li et al., 2021), and DTFD-MIL (Zhang et al., 2022) exemplify this paradigm. Transformer-based and hierarchical approaches, such as TransMIL (Shao et al., 2021), HIPT (Chen et al., 2022), LongMIL (Li et al., 2023), and HMIL (Jin et al., 2024) further address the challenge of

modeling large collections of patch representations. Pathology foundation models such as RetCCL (Wang et al., 2023), UNI (Chen et al., 2024), CONCH (Lu et al., 2024), and Virchow (Vorontsov et al., 2023) provide general-purpose patch-level representations, but are generally applied to predefined patch geometries and do not adaptively allocate tokens spatially.

Hopfield networks. Modern Hopfield networks provide an energy-based formulation of associative retrieval (Krotov and Hopfield, 2016; Demircigil et al., 2017; Ramsauer et al., 2020), and have also been applied to multiple instance learning through Hopfield-based pooling (Widrich et al., 2020). In contrast, WEAVE uses the Hopfield energy as a spatial importance scoring function for adaptive patchification, rather than as a pooling mechanism.

Efficient sequence models. For subsequent aggregation of the resulting token sequence, state-space models provide subquadratic alternatives to self-attention Vaswani et al. (2017). S4 (Gu et al., 2021) introduced structured state-space sequence models and was subsequently applied to WSI classification in S4-MIL (Fillioux et al., 2023). Mamba (Gu and Dao, 2023) introduced input-dependent state transitions and has since been adapted to WSI and visual sequence modeling (Yang et al., 2024; Zhang et al., 2025). Related recurrent architectures include xLSTM (Beck et al., 2024), which introduces matrix LSTM (mLSTM) and has been adapted to visual token sequences in Vision-LSTM (Alkin et al., 2025). We apply an mLSTM-based encoder in WSIs for slide-level token aggregation, providing subquadratic processing of the token sequence.

3 Background: DART

We briefly summarize the Dynamic Adaptive Region Tokenizer (DART) (Yin et al., 2025) in the notation used throughout this paper, as WEAVE retains DART's differentiable quantile partitioning, but replaces both its scoring mechanism and its patch tokenization.

Feature map and scoring. Given an input image $\mathcal{I} \in \mathbb{R}^{H\times W\times 3}$, we use a network f to obtain a spatial feature map $\mathcal{H} = f(\mathcal{I}) \in \mathbb{R}^{H'\times W'\times C}$. Let $\boldsymbol{h}_d \in \mathbb{R}^C$ denote the feature at location d, with $D = H'W'$ locations, where $d = 1, \dots, D$ indexes the features. A learned MLP $g_{\boldsymbol{\omega}}$ produces a scalar logit $a_d^{\text{DART}} = g_{\boldsymbol{\omega}}(\boldsymbol{h}_d)$. DART jointly standardizes these logits across batch and spatial dimensions, applies a sigmoid, and adds a positive floor:

$$s_d^{\text{DART}} = \text{sigmoid}\left(\frac{a_d^{\text{DART}} - \mu_a}{\sigma_a + \varepsilon_{\text{DART}}}\right) + \delta, \qquad \varepsilon_{\text{DART}}, \delta > 0, \tag{1}$$

where μ_a and σ_a are the mean and standard deviation of $\{a_d^{\text{DART}}\}_{d=1}^D$. The scores are then normalized within each image to obtain the spatial distribution $\{p_d^{\text{DART}}\}_{d=1}^D$.

Differentiable quantile partitioning. DART places patch boundaries at equal score-mass quantiles. For a one-dimensional marginal with normalized bin probabilities $p_1, \dots, p_B$, define the cumulative mass at the left boundary of bin i as

$$C_i = \sum_{j=1}^{i-1} p_j, \qquad C_1 = 0, \quad C_{B+1} = 1. \tag{2}$$

Using a piecewise-linear CDF, the inverse at quantile q and $i(q) \in \{1, \dots, B\}$ is

$$Q(q) = (i(q) - 1) + \frac{q - C_{i(q)}}{p_{i(q)}}, \qquad C_{i(q)} < q \le C_{i(q)+1}. \tag{3}$$

DART constructs patches in two stages. First, Q is evaluated at $q = m/M$, $m = 1, \dots, M-1$, on the row-marginal distribution, partitioning the image into M row bands of approximately equal probability mass. Given N patches per row on average, the total budget is $K = MN$. The budget is then distributed across row bands: band m receives N_m column intervals, whose boundaries are obtained from Q applied to the flattened row-band distribution. Thus, bands with higher resampled density receive more columns and hence more patches, with $\sum_{m=1}^M N_m = K$.

Patch encoding. These rectangular patches are then converted into tokens. In the original DART formulation, each region is bilinearly resized to a common spatial resolution, linearly projected, and combined with a positional embedding sampled from a learned spatial position map.

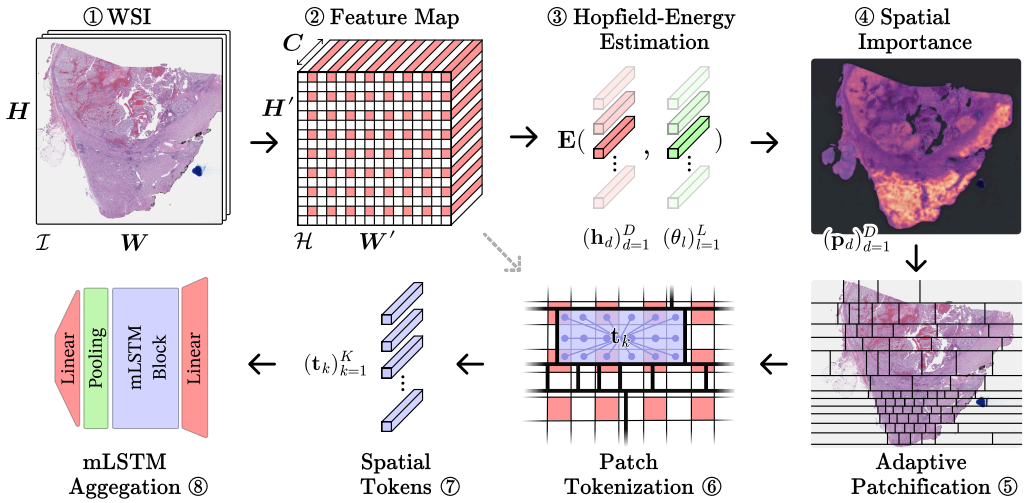


Figure 2: WEAVE overview. A frozen RetCCL encoder maps the WSI to a spatial feature map (1–2). Hopfield energy compares these features with learned prototypes to estimate spatial importance (3–4). Differentiable quantile partitioning assigns smaller patches to important regions and larger patches elsewhere (5), while overlap-weighted aggregation converts them into K patch tokens (6–7). The projected tokens are processed by the mLSTM encoder, and mean–max pooled into a slide-level feature vector, which is classified by a task-specific head (8). See Appendix C.2 for details on feature extraction and spatial subsampling.

4 WEAVE: WSI Energy-based Adaptive Vision Encoder

Figure 2 illustrates WEAVE, which we describe in the following stages: feature extraction, energy-based importance scoring, adaptive patch tokenization, and mLSTM token aggregation. Given a gigapixel WSI, WEAVE first constructs a spatial feature map using a frozen pretrained WSI patch encoder. A Hopfield energy-based scorer assigns an importance value to each feature map location, which is converted into a spatial probability distribution. We then use differentiable quantile partitioning to retrieve adaptive patch geometries according to this spatial distribution and construct one feature token per patch. Finally, the resulting tokens are aggregated into a slide-level representation using a lightweight mLSTM encoder. All trainable components downstream of the frozen feature encoder are jointly optimized end-to-end.

4.1 Feature Extraction

Applying a learnable vision encoder directly to gigapixel WSIs is computationally very expensive. Given an input WSI $\mathcal{I} \in \mathbb{R}^{H \times W \times 3}$, we instead use a frozen feature encoder f and precompute a spatial feature map

$$\mathcal{H} = f(\mathcal{I}) \in \mathbb{R}^{H' \times W' \times C}, \tag{4}$$

where H' and W' are the spatial dimensions of $\mathcal{H}$, and C is the number of feature channels.

Each location in $\mathcal{H}$ corresponds to a spatial region of $\mathcal{I}$. We flatten the spatial dimensions and denote the resulting $D = H'W'$ feature vectors by $\boldsymbol{h}_d \in \mathbb{R}^C$, $d = 1, \ldots, D$. The precomputed feature maps are reused throughout the training and serve as the basis for both importance scoring and adaptive patch tokenization. While the feature encoder could in principle be optimized jointly with the scoring, tokenization, and aggregation modules, we keep it frozen due to the additional memory requirements of gradient computation through the RetCCL encoder.

4.2 Energy-based importance scoring

We find DART's scorer to become unstable on high-resolution WSI feature maps. In our experiments, gradients through the original DART formulation occasionally become NaN during training.

Gradient clipping prevents numerical failure, but in practice does not reliably resolve the resulting highly concentrated or nearly uniform score distributions. We attribute this behavior to the combination of highly redundant feature locations and global score standardization in DART, which can amplify small differences between otherwise similar locations and, in turn, the associated gradients. This effect is further compounded by the inverse-CDF used for quantile partitioning, whose gradient can become large for bins with low probability mass (Appendix A.1). Moreover, the sigmoid and positive score floor prevent low-relevance locations from being effectively suppressed, allowing the large number of background locations to retain substantial probability mass.

To avoid these instabilities, we replace DART's scorer with a Hopfield energy-based scoring function (Ramsauer et al., 2020) based on affinity to a set of learned prototypes. Each location is scored against the same set of L prototypes, without statistics computed across the D locations. This makes the score less dependent on the scale and redundancy of gigapixel WSI feature maps.

Introducing L learnable, ℓ_2-normalized prototypes $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_L \in \mathbb{R}^C$, we define the Hopfield energy at each spatial location as

$$E(\boldsymbol{h}_d) = -\frac{1}{\beta} \log \sum_{l=1}^{L} \exp\big(\beta \langle \boldsymbol{\theta}_l, \boldsymbol{h}_d \rangle\big) + \frac{1}{\beta} \log L + \underbrace{\tfrac{1}{2}\|\boldsymbol{h}_d\|^2 + \tfrac{1}{2} \max_l \|\boldsymbol{\theta}_l\|^2}_{=1}, \tag{5}$$

where $\beta > 0$ is a learnable inverse temperature. The constant term, which reduces to 1 under our ℓ_2-normalization of features and prototypes, is retained to match the standard Modern Hopfield energy formulation of Ramsauer et al. (2020), with the gradient given in Appendix A.2.

Since lower energy indicates stronger affinity to the learned prototypes, we convert it to a higher-is-better importance score and normalize it into a spatial probability distribution $\boldsymbol{p} = (p_1, \ldots, p_D)$ required for quantile splitting:

$$\tilde{s}_d = 1 - E(\boldsymbol{h}_d), \qquad s_d = \mathrm{softplus}_\tau(\tilde{s}_d)^\gamma + \varepsilon, \qquad p_d = \frac{s_d}{\sum_{j=1}^{D} s_j}, \tag{6}$$

where $\tau > 0$ controls the sharpness of the softplus transition, $\gamma > 0$ controls the contrast among the resulting scores, and $\varepsilon > 0$ provides a numerical floor. The softplus maps importance scores to positive values while smoothly attenuating the contribution of low-relevance locations before normalization. The learned prototypes are initialized near a tissue–background direction estimated from training slides, details provided in Appendix A.3.

4.3 Adaptive patch tokenization

Given the spatial distribution $\boldsymbol{p}$ from our scorer, we retain DART's differentiable quantile partitioning to determine patch geometries, but construct patch tokens directly from feature aggregation rather than DART's resize-and-project encoding. We describe the patch budget, the resulting partitioning, and this token construction as follows.

Patch budget. The number of patches K scales with the feature map and, consequently, image resolution, and is determined by a downsampling factor r:

$$M = \lceil H'/r \rceil, \qquad N = \lceil W'/r \rceil, \qquad K = MN. \tag{7}$$

Patch geometry. Each patch P_k occupies a rectangular region $[\rho_k^-, \rho_k^+) \times [\kappa_k^-, \kappa_k^+)$ in the spatial coordinate system of $\mathcal{H}$. The row boundaries $\rho_k^\pm$ are obtained by evaluating the quantile function Q (Eq. (3)) on the row-marginal of $\boldsymbol{p}$, producing M row bands. The column boundaries $\kappa_k^\pm$ are obtained by evaluating Q on the flattened row-band distribution, allocating N_m column bands to row m, with $\sum_{m=1}^{M} N_m = K$, following DART's nested partitioning scheme (Section 3). The patches are indexed $k = 1, \ldots, K$ in row-major order over the M row bands and, within each band, over its N_m column bands.

Crucially, N_m varies between rows rather than being fixed at the nominal value N as row bands with higher resampled density are allocated proportionally more column bands, so denser rows are split into more patches while sparser rows are split into fewer.

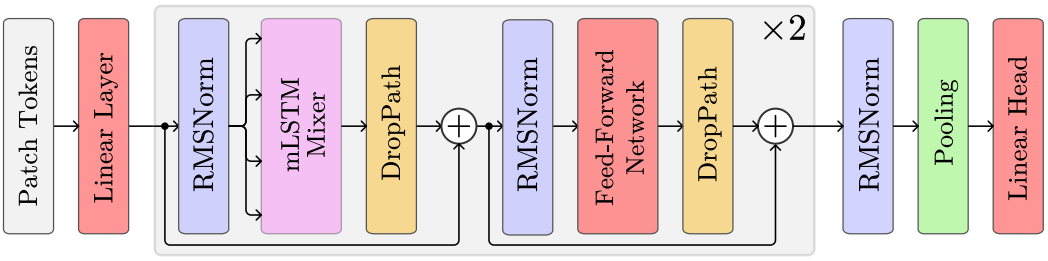


Figure 3: mLSTM-based token aggregation. Patch tokens are first linearly projected and processed by two pre-normalized residual mLSTM blocks. A final RMSNorm is followed by element-wise mean-max pooling, concatenation, and a task-specific classification head.

Patch tokens. Rather than DART's approach of resizing each region to a common resolution before projection, we construct each patch token by weighted averaging over the feature map locations it covers. Each location d, in the row $r_d \in \{0, \dots, H' - 1\}$ and column $c_d \in \{0, \dots, W' - 1\}$, occupies the unit cell $[r_d, r_d{+}1) \times [c_d, c_d{+}1)$. The aggregation weight for location d and patch P_k is:

$$w_{kd} = \left[\min(r_d{+}1, \rho_k^+) - \max(r_d, \rho_k^-)\right]_+ \cdot \left[\min(c_d{+}1, \kappa_k^+) - \max(c_d, \kappa_k^-)\right]_+, \tag{8}$$

where $[\cdot]_+ = \max(0, \cdot)$ and $w_{kd} = 0$ for cells or locations that do not overlap outside P_k. The token sequence $\boldsymbol{t}$ is then obtained by geometric aggregation over $\mathcal{H}$:

$$\boldsymbol{t}_k = \frac{\sum_{d=1}^{D} w_{kd} \boldsymbol{h}_d}{\sum_{d=1}^{D} w_{kd}}, \qquad \boldsymbol{t} = (\boldsymbol{t}_1, \dots, \boldsymbol{t}_K). \tag{9}$$

The token sequence follows a fixed raster order without positional encodings. We discuss this choice and the potential for overfitting to positional or geometric cues in Appendix B.

4.4 mLSTM token aggregation

The patch tokens $\boldsymbol{t}$ are aggregated into a slide-level representation using a lightweight mLSTM encoder based on xLSTM (Beck et al., 2024). We first project each token independently to a lower-dimensional feature space,

$$\boldsymbol{t}' = \phi_{\text{proj}}(\boldsymbol{t}) \in \mathbb{R}^{K \times C'}, \tag{10}$$

where ϕ_{proj} is a shared linear projection. The projected tokens are then processed by two pre-normalized residual blocks with the same architecture. Each block consists of an mLSTM mixer and a two-layer feed-forward network, with DropPath regularization (Touvron et al., 2021) applied to both residual branches, as illustrated in Figure 3. For a block with input $\boldsymbol{x}$, the intermediate representation $\boldsymbol{z}$ after the mLSTM branch and the block output $\boldsymbol{z}_{\text{out}}$ are

$$\boldsymbol{z} = \boldsymbol{x} + \text{DropPath}(\text{mLSTM}(\text{RMSNorm}(\boldsymbol{x}))), \tag{11}$$

$$\boldsymbol{z}_{\text{out}} = \boldsymbol{z} + \text{DropPath}(\text{MLP}(\text{RMSNorm}(\boldsymbol{z}))). \tag{12}$$

The output of the first block is used as the input of the second block. After the two blocks, we apply a final RMSNorm, followed by mean and element-wise maximum pooling over the token dimension. The pooled representations are concatenated and passed to a task-specific linear classification head.

Unlike pairwise self-attention, the mLSTM mixer does not form a $K \times K$ interaction matrix. Its chunkwise formulation processes the sequence in chunks while maintaining a fixed-size matrix-valued recurrent state, resulting in linear scaling with the number of tokens K (Beck et al., 2026).

5 Experiments

Evaluation. We evaluate WEAVE on five WSI classification datasets: BRACS (Brancati et al., 2022), DHMC (Zhu et al., 2021), PANDA (Bulten et al., 2022), TCGA-NSCLC, and TCGA-BRCA

Table 1: WSI classification results across five benchmark datasets. Values for WEAVE are mean ± standard deviation over five runs; baseline values are from Zhang et al. (2025). Higher is better; the best value per column is shown in bold, the second best underlined.

Method	BRACS			DHMC			PANDA			TCGA-NSCLC			TCGA-BRCA		
	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC	Acc	F1	AUC
ABMIL	0.7057	0.6015	**0.8939**	0.8684	0.7774	0.9695	0.4883	0.4269	0.7797	0.8758	0.8756	0.9572	0.9292	0.8893	0.9747
DSMIL	0.6759	0.5618	0.8618	0.8711	0.7934	0.9583	0.4633	0.3847	0.7660	0.8782	0.8780	0.9567	0.9375	0.8961	0.9770
CLAM	0.7103	0.6014	0.9016	0.8711	0.7909	**0.9727**	0.4802	0.4224	0.7820	0.8804	0.8803	0.9536	0.9333	0.8960	0.9753
DTFD-MIL	0.7012	0.6131	0.8787	0.8711	0.7704	0.9521	0.4704	0.3853	0.7665	0.8736	0.8732	0.9559	0.9271	0.8809	0.9633
TransMIL	0.6919	0.6063	0.8759	0.8067	0.7136	0.9466	0.4636	0.3970	0.7728	0.8850	0.8845	**0.9626**	0.9375	0.9028	0.9763
S4-MIL	0.6621	0.5904	0.8457	0.8644	0.7847	0.9284	0.5047	0.4486	0.7986	0.8851	0.8849	0.9571	0.9458	0.9154	0.9770
MambaMIL	0.7379	0.6832	0.8883	0.8550	0.7789	0.9661	0.4679	0.4216	0.7781	0.8758	0.8756	0.9582	0.9333	0.8939	0.9657
SRMambaMIL	0.7379	0.6789	0.8915	0.8590	0.7735	0.9639	0.4711	0.4209	0.7776	0.8850	0.8849	0.9592	0.9313	0.8900	0.9657
2DMambaMIL	**0.7517**	**0.7045**	0.8964	**0.8926**	0.8027	0.9468	0.5075	0.4562	0.8184	0.8851	0.8850	0.9618	0.9458	0.9156	**0.9782**
WEAVE (Ours)	0.7126 ±0.0141	0.6837 ±0.0209	0.8617 ±0.0157	**0.8926** ±0.0307	**0.8653** ±0.0292	0.9609 ±0.0107	**0.5477** ±0.0066	**0.4821** ±0.0082	**0.8239** ±0.0039	**0.8889** ±0.0176	**0.8886** ±0.0176	0.9613 ±0.0098	**0.9544** ±0.0061	**0.9293** ±0.0086	0.9693 ±0.0035

Table 2: Ablation of adaptive patchification and slide-level aggregation on DHMC and TCGA-NSCLC. Uniform + Mean-Max uses uniform patchification with mean-max pooling. Adaptive (DART) + Mean-Max uses the original DART scorer, while Adaptive (Energy-based) + Mean-Max uses our proposed energy-based scorer. Values are mean ± standard deviation over three runs.

Method	DHMC			TCGA-NSCLC		
	Acc	F1	AUC	Acc	F1	AUC
Uniform + Mean-Max	0.718 ±0.101	0.655 ±0.102	0.921 ±0.050	0.851 ±0.035	0.850 ±0.035	0.959 ±0.006
Adaptive (DART) + Mean-Max[1]	–	–	–	–	–	–
Adaptive (Energy-based) + Mean-Max	0.821 ±0.014	0.792 ±0.038	0.953 ±0.014	0.866 ±0.029	0.865 ±0.029	0.967 ±0.003
WEAVE (Ours)	0.875 ±0.016	0.841 ±0.023	0.977 ±0.011	0.897 ±0.011	0.896 ±0.011	0.962 ±0.003

(The Cancer Genome Atlas Research Network, 2013), using the data splits released by Zhang et al. (2025). We report accuracy (Acc), macro-F1 (F1), and one-vs-one multiclass ROC-AUC (AUC). We compare WEAVE with established MIL methods, including ABMIL (Ilse et al., 2018), DSMIL (Li et al., 2021), CLAM (Lu et al., 2021), DTFD-MIL (Zhang et al., 2022), TransMIL (Shao et al., 2021), S4-MIL (Fillioux et al., 2023), MambaMIL and SRMambaMIL (Yang et al., 2024), and 2DMambaMIL (Zhang et al., 2025). Dataset and split details are provided in Appendix C.1.

Implementation. We extract spatial feature maps using a frozen RetCCL encoder (Wang et al., 2023). WSIs are processed at $10\times$ magnification, and we extract 256×256 pixel patches. The resulting features are arranged as a dense spatial feature map, which serves as input to the energy-based scoring and adaptive patchification stages of WEAVE. Details on feature-map construction are provided in Appendix C.2. In contrast, 2DMambaMIL (Zhang et al., 2025) uses UNI features extracted from 512×512 pixel patches at $20\times$ magnification. These patches cover a similar physical field of view but sample the WSI at twice the spatial resolution. Since WEAVE requires a denser feature map for adaptive patchification, we use the substantially smaller 21M-parameter RetCCL encoder rather than the 307M-parameter UNI encoder (Chen et al., 2024). Evaluations generally report stronger downstream performance for UNI than RetCCL across several pathology tasks (Patkar et al., 2024; Wang et al., 2026; Kömen et al., 2026).

All trainable components of WEAVE are optimized end-to-end with a task-specific classification head, while the RetCCL encoder remains frozen. We use AdamW (Loshchilov and Hutter, 2017), cross-entropy loss, and no class weighting. Details on the training configuration and hardware are provided in Appendices C.3 and C.4.

Results. Table 1 compares WEAVE with nine existing WSI classification methods on five benchmark datasets. WEAVE achieves the best or tied-best accuracy and the highest F1 score on four of the five datasets (DHMC, PANDA, TCGA-NSCLC, TCGA-BRCA). AUC is more mixed where WEAVE leads only on PANDA and trails the best baseline by a small margin on DHMC, TCGA-NSCLC, and TCGA-BRCA, and by a larger margin on BRACS. Overall, WEAVE shows competitive or improved classification performance, despite using a smaller RetCCL encoder and lower-magnification features than the baselines against which it is compared.

[1]The original DART scorer produced NaN gradients during training. Gradient clipping and varying optimization hyperparameters did not yield stable training; see Sec. 4.2.

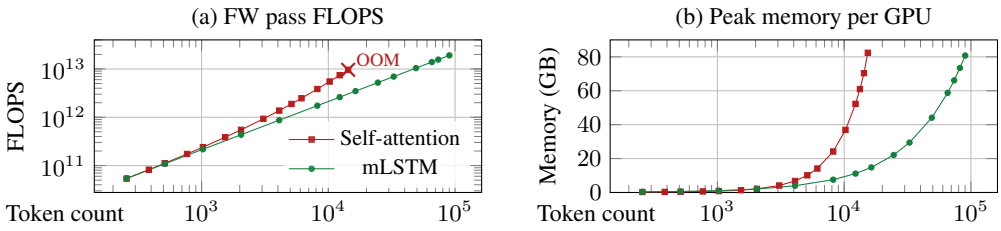


Figure 4: Computational scaling of mLSTM-based aggregation and self-attention on DHMC with the number of patch tokens. We report forward-pass FLOPS and peak GPU memory per training step. Self-attention exhibits substantially steeper scaling and exhausts GPU memory at larger token counts, whereas mLSTM shows approximately linear scaling over the evaluated range.

Ablation study. We isolate adaptive patchification and slide-level aggregation by comparing uniform patchification, DART, and energy-based patchification with mean-max pooling and the full WEAVE model. All variants use the same training configuration and token budget per dataset. The mean token count per WSI is approximately 700 on DHMC and 850 on TCGA-NSCLC. The results are reported in Table 2. Comparing uniform and energy-based patchification with the same mean-max aggregator isolates the effect of adaptive patchification, while comparing energy-based patchification with mean-max pooling to WEAVE isolates the effect of mLSTM-based token aggregation. Energy-based adaptive patchification outperforms uniform patchification, and replacing mean-max pooling with mLSTM-based aggregation yields a further improvement.

Aggregation scaling. To assess the computational scaling of mLSTM-based token aggregation, we compare it with an otherwise identical architecture using standard self-attention instead of mLSTM. We evaluate both models on DHMC while varying the number of tokens and keeping the remaining configuration fixed. Figure 4 reports FLOPS per forward pass and peak GPU memory during training. Where both aggregators are feasible, classification performance remains comparable (Appendix C.5). Self-attention incurs higher computational and memory costs as token count increases and eventually exceeds available GPU memory.

Qualitative analysis of adaptive patchification. Figure 5 visualizes the importance scoring and adaptive patchification on representative PANDA and DHMC WSI cutouts. The energy-based importance scores exhibit spatial variation throughout the tissue, with higher scores concentrated in selected regions. The resulting partitions adapt to these variations by assigning smaller patches to high-scoring regions and larger patches to low-scoring regions. Thus, rather than distributing the token budget uniformly across the slide, WEAVE allocates more spatial resolution to regions that receive higher importance scores while using fewer tokens elsewhere.

6 Discussion

The main benchmark results demonstrate that WEAVE achieves competitive classification performance as an end-to-end pipeline despite using a comparatively lightweight feature-extraction setup. In particular, WEAVE uses the substantially smaller RetCCL encoder at lower magnification, whereas the reported baseline results use UNI at higher magnification. Thus, the results show that the proposed pipeline remains competitive under a more conservative feature-extraction configuration, although the different encoders prevent the main benchmark from isolating the contribution of adaptive patchification and aggregation alone.

Our controlled ablations indicate that adaptive patchification and mLSTM-based aggregation provide complementary benefits. Adaptive patchification controls *where* information is extracted into tokens by allocating the token budget to regions based on predicted relevance, while mLSTM aggregation controls *how* the resulting tokens are combined into a slide-level representation. Together, these components address two distinct efficiency challenges in gigapixel WSI processing: allocating a limited token budget across a large spatial domain and aggregating those tokens without forming quadratic pairwise interactions. Their separate improvements at a matched token budget support treating these as distinct components of the pipeline. The aggregation scaling experiments further show that mLSTM-based aggregation maintains comparable classification performance while avoid-

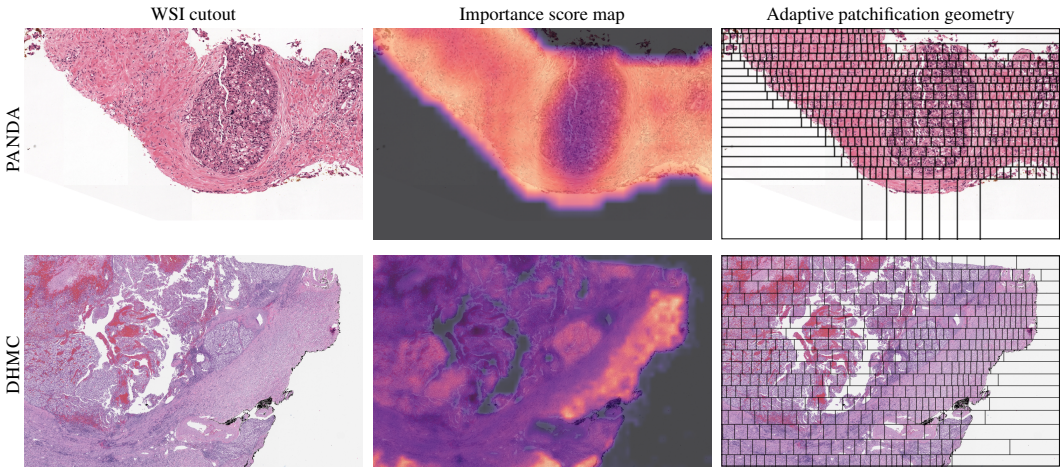


Figure 5: WEAVE scoring and patchification on representative cutouts from PANDA and DHMC WSIs. The energy-based importance score maps use the magma colormap, with brighter colors indicating higher importance scores. Quantile partitioning on the score map determines the adaptive patch geometry, assigning finer or coarser patches to regions depending on the scores.

ing the quadratic computational and memory growth associated with quadratic self-attention as the token budget increases.

The comparison with DART's scorer also highlights a potential instability when applying adaptive patchification to gigapixel pathology data. Importance scoring that relies on statistics computed across batch and spatial dimensions can become numerically unstable for the highly redundant feature statistics encountered in WSIs. In our experiments, this resulted in NaN gradients during training with DART's original scorer. Our Hopfield energy-based formulation instead scores each location through its affinity to a set of learned prototypes, avoiding dependence on such global statistics. This comparison is specific to the WSI setting evaluated here and does not establish that our proposed energy-based scoring is universally preferable to DART's.

Limitations. Our evaluation leaves several dimensions unexplored. WEAVE uses frozen RetCCL features, and alternative encoders, magnifications, or token budgets may affect performance. We focus on five WSI classification benchmarks and do not evaluate tasks such as slide-level regression or segmentation. Future work should investigate how positional and geometric information can be incorporated into adaptive patchification without causing shortcut learning based on spatial cues.

The main benchmark baselines are taken from Zhang et al. (2025) rather than reproduced in a common feature-extraction and training pipeline. This limits the extent to which the reported results can be interpreted as controlled comparisons between different approaches. Only on BRACS does WEAVE drop below the strongest reported baseline. The reason for this difference remains unclear and requires further controlled investigation.

7 Conclusion

We presented WEAVE, an encoding framework for gigapixel whole-slide images that combines Hopfield energy-based importance scoring, adaptive patchification, and efficient mLSTM-based token aggregation. Across five WSI classification benchmarks, WEAVE achieves competitive or improved classification performance while allocating a given token budget non-uniformly across the slide. By addressing both spatial token allocation and efficient slide-level aggregation, WEAVE provides a unified approach that constrains the computational cost of processing large WSI inputs. These results support the value of jointly adapting *where* tokens are allocated and *how* they are aggregated for efficient gigapixel WSI modeling.

AI USE STATEMENT

In this work, we used generative AI tools as a coding assistant and for drafting and editing the paper. For code, AI tools were used to assist with implementation and code-related tasks. For the writing of the paper itself, AI tools were used to assist with drafting, improving readability, and correcting grammar and language. We did not use generative AI tools to develop or ideate the scientific methods, propose or refine hypotheses, design or provide feedback on experiments or experimental setups, or interpret the experimental results. All AI-assisted work was reviewed by the authors. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This work uses publicly available whole-slide image datasets for research purposes. We do not collect new data from human subjects and no additional personally identifiable information is accessed or processed as part of this study. We use the datasets in accordance with their respective terms of use and licenses. The models are evaluated for research purposes and are not intended for clinical diagnosis or decision-making.

REPRODUCIBILITY STATEMENT

The implementation details and model specifications are provided in Section 4, while experimental settings, dataset splits, preprocessing procedures, and training configurations are described in Section 5 and Appendix C. To facilitate reproducibility, the source code and experiment configurations will be publicly released upon acceptance.

REFERENCES

Aasan, M., Hjelkrem Tan, M., Catalano, N., Choi, C., and Ramírez Rivera, A. (2026). Differentiable hierarchical visual tokenization. *Advances in Neural Information Processing Systems*, 38:87605–87634.

Alkin, B., Beck, M., Pöppel, K., Hochreiter, S., and Brandstetter, J. (2025). Vision-lstm: xlstm as generic vision backbone. In *International Conference on Learning Representations*, volume 2025, pages 23804–23826.

Beck, M., Pöppel, K., Lippe, P., and Hochreiter, S. (2026). Tiled flash linear attention: More efficient linear rnn and xlstm kernels. *Advances in Neural Information Processing Systems*, 38:75093–75148.

Beck, M., Pöppel, K., Spanring, M., Auer, A., Prudnikova, O., Kopp, M., Klambauer, G., Brandstetter, J., and Hochreiter, S. (2024). xlstm: Extended long short-term memory. In *Thirty-eighth Conference on Neural Information Processing Systems*.

Beyer, L., Izmailov, P., Kolesnikov, A., Caron, M., Kornblith, S., Zhai, X., Minderer, M., Tschannen, M., Alabdulmohsin, I., and Pavetic, F. (2023). Flexivit: One model for all patch sizes. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14496–14506. IEEE.

Bolya, D., Fu, C.-Y., Dai, X., Zhang, P., Feichtenhofer, C., and Hoffman, J. (2023). Token merging: Your ViT but faster. In *International Conference on Learning Representations*.

Brancati, N., Anniciello, A. M., Pati, P., Riccio, D., Scognamiglio, G., Jaume, G., De Pietro, G., Di Bonito, M., Foncubierta, A., Botti, G., Gabrani, M., Feroce, F., and Frucci, M. (2022). Bracs: A dataset for breast carcinoma subtyping in h&e histology images. *Database*, 2022:baac093.

Bulten, W., Kartasalo, K., Chen, P.-H. C., Ström, P., Pinckaers, H., Nagpal, K., Cai, Y., Steiner, D. F., Van Boven, H., Vink, R., et al. (2022). Artificial intelligence for diagnosis and gleason grading of prostate cancer: the panda challenge. *Nature medicine*, 28(1):154–163.

Chen, R. J., Chen, C., Li, Y., Chen, T. Y., Trister, A. D., Krishnan, R. G., and Mahmood, F. (2022). Scaling vision transformers to gigapixel images via hierarchical self-supervised learning. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 16123–16134. IEEE.

Chen, R. J., Ding, T., Lu, M. Y., Williamson, D. F., Jaume, G., Chen, B., Zhang, A., Shao, D., Song, A. H., Shaban, M., et al. (2024). Towards a general-purpose foundation model for computational pathology. *Nature Medicine*.

Demircigil, M., Heusel, J., Löwe, M., Upgang, S., and Vermet, F. (2017). On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Fang, Z., Wang, Y., Zhang, Y., Wang, Z., Zhang, J., Ji, X., and Zhang, Y. (2024). Mammil: Multiple instance learning for whole slide images with state space models. In *2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 3200–3205. IEEE.

Fillioux, L., Boyd, J., Vakalopoulou, M., Cournède, P.-H., and Christodoulidis, S. (2023). Structured state space models for multiple instance learning in digital pathology. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 594–604. Springer.

Gu, A. and Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.

Gu, A., Goel, K., and Ré, C. (2021). Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*.

Havtorn, J. D., Royer, A., Blankevoort, T., and Bejnordi, B. E. (2023). Msvit: Dynamic mixed-scale tokenization for vision transformers. In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 838–848. IEEE.

Ilse, M., Tomczak, J., and Welling, M. (2018). Attention-based deep multiple instance learning. In *International conference on machine learning*, pages 2127–2136. Pmlr.

Jin, C., Luo, L., Lin, H., Hou, J., and Chen, H. (2024). Hmil: Hierarchical multi-instance learning for fine-grained whole slide image classification. *IEEE Transactions on Medical Imaging*, 44(4):1796–1808.

Kömen, J., de Jong, E. D., Hense, J., Marienwald, H., Dippel, J., Naumann, P., Marcus, E., Ruff, L., Alber, M., Teuwen, J., et al. (2026). Towards robust foundation models for digital pathology. *Nature Communications*, 17(1):5218.

Krotov, D. and Hopfield, J. J. (2016). Dense associative memory for pattern recognition. *Advances in neural information processing systems*, 29.

Li, B., Li, Y., and Eliceiri, K. W. (2021). Dual-stream multiple instance learning network for whole slide image classification with self-supervised contrastive learning. In *2021 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 14313–14323. IEEE.

Li, H., Zhang, Y., Zhu, C., Cai, J., Zheng, S., and Yang, L. (2023). Long-mil: scaling long contextual multiple instance learning for histopathology whole slide image analysis. *arXiv preprint arXiv:2311.12885*.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lu, M. Y., Chen, B., Williamson, D. F., Chen, R. J., Liang, I., Ding, T., Jaume, G., Odintsov, I., Le, L. P., Gerber, G., et al. (2024). A visual-language foundation model for computational pathology. *Nature Medicine*.

Lu, M. Y., Williamson, D. F., Chen, T. Y., Chen, R. J., Barbieri, M., and Mahmood, F. (2021). Data-efficient and weakly supervised computational pathology on whole-slide images. *Nature biomedical engineering*, 5(6):555–570.

Patkar, S., Chen, A., Basnet, A., Bixby, A., Rajendran, R., Chernet, R., Faso, S., Kumar, P. A., Desai, D., El-Zammar, O., et al. (2024). Predicting the tumor microenvironment composition and immunotherapy response in non-small cell lung cancer from digital histopathology images. *NPJ Precision Oncology*, 8(1):280.

Ramsauer, H., Schäfl, B., Lehner, J., Seidl, P., Widrich, M., Adler, T., Gruber, L., Holzleitner, M., Pavlović, M., Sandve, G. K., et al. (2020). Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*.

Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., and Hsieh, C.-J. (2021). Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34:13937–13949.

Ryoo, M. S., Piergiovanni, A., Arnab, A., Dehghani, M., and Angelova, A. (2021). Tokenlearner: What can 8 learned tokens do for images and videos? *arXiv preprint arXiv:2106.11297*.

Shao, Z., Bian, H., Chen, Y., Wang, Y., Zhang, J., Ji, X., et al. (2021). Transmil: Transformer based correlated multiple instance learning for whole slide image classification. *Advances in neural information processing systems*, 34:2136–2147.

The Cancer Genome Atlas Research Network (2013). The cancer genome atlas pan-cancer analysis project. *Nature Genetics*, 45:1113–1120.

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., and Jégou, H. (2021). Going deeper with image transformers. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pages 32–42. IEEE.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Vorontsov, E., Bozkurt, A., Casson, A., Shaikovski, G., Zelechowski, M., Liu, S., Severson, K., Zimmermann, E., Hall, J., Tenenholtz, N., et al. (2023). Virchow: A million-slide digital pathology foundation model. *arXiv preprint arXiv:2309.07778*.

Wang, C.-W., Chu, T.-C., Wu, T.-K., Chung, Y.-P., Lin, S.-S., and Chao, T.-K. (2026). Deep learning for malignancy and tumor origin prediction using cytology or histopathology whole slide images. *npj Digital Medicine*, 9(1):175.

Wang, X., Du, Y., Yang, S., Zhang, J., Wang, M., Zhang, J., Yang, W., Huang, J., and Han, X. (2023). Retccl: Clustering-guided contrastive learning for whole-slide image retrieval. *Medical image analysis*, 83:102645.

Widrich, M., Schäfl, B., Pavlović, M., Ramsauer, H., Gruber, L., Holzleitner, M., Brandstetter, J., Sandve, G. K., Greiff, V., Hochreiter, S., et al. (2020). Modern hopfield networks and attention for immune repertoire classification. *Advances in neural information processing systems*, 33:18832–18845.

Yang, S., Wang, Y., and Chen, H. (2024). Mambamil: Enhancing long sequence modeling with sequence reordering in computational pathology. In *International conference on medical image computing and computer-assisted intervention*, pages 296–306. Springer.

Yin, S., Yin, K., Liu, Y., Chen, W., and Lin, L. (2025). Dart: Differentiable dynamic adaptive region tokenizer for vision foundation models. *arXiv preprint arXiv:2506.10390*.

Zhang, E., Lyngaas, I., Chen, P., Wang, X., Igarashi, J., Huo, Y., Munetomo, M., and Wahib, M. (2024). Adaptive patching for high-resolution image segmentation with transformers. In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Zhang, H., Meng, Y., Zhao, Y., Qiao, Y., Yang, X., Coupland, S. E., and Zheng, Y. (2022). Dtfd-mil: Double-tier feature distillation multiple instance learning for histopathology whole slide image classification. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18780–18790. IEEE.

Zhang, J., Nguyen, A. T., Han, X., Trinh, V. Q.-H., Qin, H., Samaras, D., and Hosseini, M. S. (2025). 2dmamba: Efficient state space model for image representation with applications on giga-pixel whole slide image classification. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3583–3592. IEEE.

Zhu, M., Ren, B., Richards, R., Suriawinata, M., Tomita, N., and Hassanpour, S. (2021). Development and evaluation of a deep neural network for histologic classification of renal cell carcinoma on biopsy and surgical resection slides. *Scientific reports*, 11(1):1–9.

Table 3: Notation used throughout WEAVE.

Symbol	Dimension	Description
Input and RetCCL features		
$\mathcal{I}$	$\mathbb{R}^{H\times W\times 3}$	Input WSI or image
H, W	$\in \mathbb{N}$	Spatial dimensions of the input
$\mathcal{H}$	$\mathbb{R}^{H'\times W'\times C}$	Spatial feature map
H', W'	$\in \mathbb{N}$	Spatial dimensions of $\mathcal{H}$
C	$\in \mathbb{N}$	RetCCL feature dimension
D	$\in \mathbb{N}$	Number of candidate locations ($H' \times W'$)
d	$\in \{1, \dots, D\}$	Candidate-location index
$\boldsymbol{h}_d$	$\mathbb{R}^C$	Feature vector at location d
Energy-based prototype scoring		
L	$\in \mathbb{N}$	Number of learnable prototypes
$\boldsymbol{\theta}_l$	$\in \mathbb{R}^C$	l-th ℓ_2-normalized prototype
β	$\in \mathbb{R}_{>0}$	Inverse temperature for prototype scoring
$E(\boldsymbol{h}_d)$	$\in \mathbb{R}$	Hopfield energy at location d
τ	$\in \mathbb{R}_{>0}$	Softplus sharpness
γ	$\in \mathbb{R}_{>0}$	Exponent controlling score contrast
ε	$\in \mathbb{R}_{>0}$	Positive numerical score floor
$\tilde{s}_d$	$\in [-1, 1]$	Higher-is-better, energy-derived score at location d
s_d	$\in \mathbb{R}_{>0}$	Softplus-transformed, exponentiated score at location d
p_d	$\in [0, 1]$	Sum-normalized score at location d
$\boldsymbol{p}$	$\in [0, 1]^D$	Sum-normalized spatial score distribution
Scoring and adaptive partitioning		
$g_{\boldsymbol{\omega}}$	$\mathbb{R}^C \to \mathbb{R}$	DART MLP scoring head
a_d^{DART}	$\in \mathbb{R}$	DART score logit at location d
μ_a, σ_a	$\in \mathbb{R}$	Mean and standard deviation of the DART logits
s_d^{DART}	$\in \mathbb{R}_{>0}$	DART score at location d
p_d^{DART}	$\in [0, 1]$	DART sum-normalized spatial distribution
p_i	$\in [0, 1]$	Probability mass of bin $i \in \{1, \dots, B\}$ in a one-dimensional marginal
C_i	$\in [0, 1]$	Cumulative mass at the left boundary of bin i
B	$\in \mathbb{N}$	Number of bins in a one-dimensional marginal
Q	$[0, 1] \to \mathbb{R}$	Piecewise-linear inverse CDF used for quantile partitioning
q	$\in [0, 1]$	Quantile level defining a patch boundary
$i(q)$	$\in \{1, \dots, B\}$	Bin containing quantile q
r	$\in \mathbb{N}$	Patch budget downsampling factor
K	$\in \mathbb{N}$	Total number of adaptive patches, $K = MN = \sum_{m=1}^{M} N_m$
M	$\in \mathbb{N}$	Number of row bands
N	$\in \mathbb{N}$	Nominal column bands per row band
N_m	$\in \mathbb{N}$	Number of patches allocated to row band m
P_k	$\subseteq \mathbb{R}^2$	k-th rectangular patch in feature map coordinates
Adaptive patch tokenization		
ρ_k^-, ρ_k^+	$\in \mathbb{R}$	Row boundaries of patch P_k
κ_k^-, κ_k^+	$\in \mathbb{R}$	Column boundaries of patch P_k within its row band
w_{kd}	$\in \mathbb{R}_{\geq 0}$	Aggregation weight for location d in patch P_k
$\boldsymbol{t}$	$\mathbb{R}^{K\times C}$	Patch-token sequence obtained by spatially weighted aggregation
Prototype initialization		
$\boldsymbol{\mu}_{\mathcal{T}}, \boldsymbol{\mu}_{\mathcal{B}}$	$\mathbb{R}^C$	Mean tissue and background features
$\bar{\boldsymbol{\mu}}_{\mathcal{T}}, \bar{\boldsymbol{\mu}}_{\mathcal{B}}$	$\mathbb{R}^C$	ℓ_2-normalized tissue and background means
$\boldsymbol{\theta}_{\text{init}}$	$\mathbb{R}^C$	Tissue–background initialization direction
$\mathcal{T}, \mathcal{B}$	$\subseteq \{1, \dots, D\}$	Tissue and background feature-location sets
b_d	$\in [0, 1]$	Patch-level background score
Token encoding and sequence modeling		
C'	$\in \mathbb{N}$	mLSTM feature dimension
ϕ_{proj}	$\mathbb{R}^C \to \mathbb{R}^{C'}$	Shared linear token projection
$\boldsymbol{t}'$	$\mathbb{R}^{K\times C'}$	Projected token sequence input to the mLSTM
$\boldsymbol{z}$	$\mathbb{R}^{K\times C'}$	Intermediate sequence representation after the mLSTM branch
$\boldsymbol{z}_{\text{out}}$	$\mathbb{R}^{K\times C'}$	Output token sequence from the mLSTM

A METHOD DETAILS

A.1 SENSITIVITY OF THE DIFFERENTIABLE QUANTILE PARTITION.

The differentiable quantile partition used by DART (Sec. 3) is obtained by inverting a piecewise-linear cumulative distribution. For a fixed bin $i(q)$, the inverse is differentiable with respect to the histogram values, with

$$\frac{\partial Q(q)}{\partial p_j} = \begin{cases} -\dfrac{1}{p_{i(q)}}, & j < i(q), \\ -\dfrac{q - C_{i(q)}}{p_{i(q)}^2}, & j = i(q), \\ 0, & j > i(q). \end{cases} \tag{13}$$

Thus, the inverse-CDF derivative contains the inverse powers of the probability mass $p_{i(q)}$ of the bin containing the quantile. In particular, the derivative with respect to the probability of the active bin contains a factor $1/p_{i(q)}^2$. As $p_{i(q)}$ becomes small, small changes in the probability distribution can therefore produce large changes in the corresponding quantile location.

The derivative is defined for a fixed active bin and is not defined when q lies exactly on a bin boundary, where the active bin changes. Away from such boundaries, the quantile operation remains differentiable and can be optimized by backpropagation. Thus, the quantile partition itself is differentiable and is retained unchanged in WEAVE.

In our high-resolution WSI setting, however, we find that DART's MLP-based scoring can produce unfavorable spatial probability distributions. Its global standardization makes scores dependent on statistics across all spatial locations, while the sigmoid and positive floor prevent low-scoring locations from being effectively suppressed. With many redundant and background locations, small differences between otherwise similar locations can therefore affect the normalized distribution. Combined with the inverse-CDF sensitivity above, we observed unstable gradients through the DART scoring and quantile-partitioning pipeline, including NaN gradients during training.

A.2 GRADIENT OF THE ENERGY-BASED SCORING FUNCTION.

The gradient of the scorer $\tilde{s}_d$ in Equation (6) with respect to the normalized feature vector $\boldsymbol{h}_d$ is

$$\nabla_{\boldsymbol{h}_d} \tilde{s}_d = \sum_{l=1}^{L} \mathrm{softmax}_l\big(\beta\langle\boldsymbol{\theta}_1, \boldsymbol{h}_d\rangle, \ldots, \beta\langle\boldsymbol{\theta}_L, \boldsymbol{h}_d\rangle\big)\boldsymbol{\theta}_l. \tag{14}$$

where $\mathrm{softmax}_l(\cdot)$ denotes the l-th component of the softmax. Thus, the gradient is a softmax-weighted combination of prototype directions, with prototypes having a higher affinity to $\boldsymbol{h}_d$ receiving a greater weight. Similarly, the gradient with respect to each normalized prototype $\boldsymbol{\theta}_l$ is proportional to $\boldsymbol{h}_d$ and weighted by the same softmax coefficient. Consequently, locations with higher affinity to a prototype exert a stronger influence on its update.

A.3 PROTOTYPE INITIALIZATION

We initialize the prototypes using tissue–background contrast estimated from 32 randomly sampled training slides. Each sampled slide provides feature vectors $\boldsymbol{h}_d$ at the spatial locations of the feature map.

We first identify tissue and background locations using a simple color-based heuristic.[2] For each pixel, we compute its mean brightness and channel range and assign a high background score to pixels that are sufficiently bright and have low color variation. Averaging these pixel-level scores over each feature map location yields a patch-level background score b_d. We threshold b_d at η, set to the median background score across all sampled feature map locations, to obtain the tissue and background location sets

$$\mathcal{T} = \{d : b_d < \eta\}, \qquad \mathcal{B} = \{d : b_d \geq \eta\}. \tag{15}$$

[2]Similar in purpose to the tissue segmentation used by CLAM (Lu et al., 2021), though based on brightness and color-range thresholds rather than HSV saturation and contours.

We then compute the corresponding mean feature vectors

$$\boldsymbol{\mu}_{\mathcal{T}} = \frac{1}{|\mathcal{T}|} \sum_{d \in \mathcal{T}} \boldsymbol{h}_d, \qquad \boldsymbol{\mu}_{\mathcal{B}} = \frac{1}{|\mathcal{B}|} \sum_{d \in \mathcal{B}} \boldsymbol{h}_d. \tag{16}$$

Normalizing these means and taking their difference gives the tissue–background initialization direction:

$$\bar{\boldsymbol{\mu}}_{\mathcal{T}} = \frac{\boldsymbol{\mu}_{\mathcal{T}}}{\|\boldsymbol{\mu}_{\mathcal{T}}\|_2}, \qquad \bar{\boldsymbol{\mu}}_{\mathcal{B}} = \frac{\boldsymbol{\mu}_{\mathcal{B}}}{\|\boldsymbol{\mu}_{\mathcal{B}}\|_2}, \qquad \boldsymbol{\theta}_{\text{init}} = \frac{\bar{\boldsymbol{\mu}}_{\mathcal{T}} - \bar{\boldsymbol{\mu}}_{\mathcal{B}}}{\|\bar{\boldsymbol{\mu}}_{\mathcal{T}} - \bar{\boldsymbol{\mu}}_{\mathcal{B}}\|_2}. \tag{17}$$

All prototypes are initialized near $\boldsymbol{\theta}_{\text{init}}$ with small random perturbations orthogonal to this direction, followed by ℓ_2-normalization. This yields distinct initial prototype directions while providing a shared tissue–background prior. The heuristic is used only for initialization; the prototypes and temperature β are subsequently optimized end-to-end during training.

B Potential Positional and Geometric Shortcuts

WEAVE does not provide the token aggregator with explicit positional or geometric information. Each token is a pure overlap-weighted mean of feature-map cells and therefore does not encode its spatial extent, boundary coordinates, or area. The token sequence is ordered in a fixed raster order without additional positional embeddings.

During development, we experimented with positional information and observed overfitting to the training data when it was provided to the model. We hypothesize that, in the adaptive patchification setting, positional or geometric information can provide a learning shortcut, allowing the model to exploit where tokens are located or how the adaptive partition is structured rather than relying primarily on visual features.

A related concern arises when full-background tokens are removed or masked. The extent of the retained tokens may contain cues correlated with the class label, regardless of the visual features within the tissue. In the adaptive setting, this can also influence the learned partition and, consequently, the locations and arrangement of the resulting tokens. This may make such cues more accessible to the token aggregator.

These observations motivate further study of positional and geometric information in adaptive tokenization. Future work could evaluate spatial representations that preserve useful information while controlling access to class-correlated metadata, together with experiments that disentangle visual content from spatial organization. Such studies could clarify how spatial information can be incorporated without introducing unintended shortcuts.

C Experimental Details

C.1 Datasets and Splits

Dataset statistics and class configurations are summarized in Table 4. The mean number of patch tokens per slide produced by WEAVE in the main experiments is reported in Table 5.

Table 4: Dataset tasks, statistics and numbers of WSI slides in the training, validation, and test splits.

Dataset	Task	Classes	Slides (Train/Val./Test)
BRACS	Breast cancer classification	3	395/65/87
DHMC	Kidney cancer subtype classification	5	391/23/149
PANDA	Prostate cancer grading (ISUP-scale)	6	8491/1061/1062
TCGA-NSCLC	Lung cancer subtype classification	2	785/86/87
TCGA-BRCA	Breast cancer subtype classification	2	841/96/96

Table 5: Patch-budget settings and resulting mean number of patch tokens per slide in the main experiments. The downsampling factor r is selected per dataset based on GPU memory constraints and validation performance.

Dataset	Downsampling factor r	Mean tokens / slide
BRACS	4	13,651.8
DHMC	4	11,476.9
PANDA	2	10,468.2
TCGA-BRCA	5	7,930.5
TCGA-NSCLC	8	2,830.3

C.2 Feature extraction

We use a frozen RetCCL encoder (Wang et al., 2023) to construct spatial feature representations. Each WSI is rescaled to $1.0\,\mu$m per pixel and sampled with overlapping 256×256 pixel patches. RetCCL encodes each patch independently, and the resulting embeddings are arranged according to their spatial locations. Since the encoder remains frozen throughout the training, we precompute the features once for each slide.

To reduce the computational cost of importance scoring, we subsample the precomputed feature map using a coarser stride s_{score}, reducing the number of locations processed by the prototype scorer and quantile partitioning. A finer stride s_{patch} is used for patch tokenization to retain greater spatial detail when constructing patch tokens.

Both feature maps share the same WSI coordinate frame. Consequently, the quantile boundaries computed from the coarse feature map can be transferred directly to the fine feature map when computing the aggregation weights w_{kd}. We choose s_{score} so that the coarse feature map remains sufficiently resolved for the token budget determined through the downsampling factor r.

C.3 Training configuration

The shared model and optimization hyperparameters are summarized in Table 6. Hyperparameters were selected by manual tuning on the validation splits and then fixed for the reported experiments. Training is performed with early stopping based on validation performance. The batch size is selected per dataset to fit the available GPU memory.

C.4 Hardware

All experiments are conducted on a single compute node equipped with eight NVIDIA H100 GPUs with 80 GB of HBM3 memory each, two Intel Xeon Platinum 8468 CPUs and 2.0 TiB of memory.

C.5 Aggregation Scaling

The classification results corresponding to the computational scaling experiment in Sec. 5 are reported in Table 7. For token budgets up to 6,144 tokens, mLSTM and full pairwise self-attention achieve comparable classification performance across all three metrics, despite their different computational scaling. This indicates that the reduced computational and memory requirements of mLSTM aggregation do not come with a substantial degradation in predictive performance in the tested range. At 21,824 tokens, the mLSTM-based aggregation remains computationally feasible, whereas the corresponding self-attention aggregation cannot be evaluated because it exceeds the available GPU memory. The results for 700, 1,544, and 6,144 tokens are averaged over three runs; the 21,824-token mLSTM result is from a single run.

Table 6: WEAVE hyperparameters.

Hyperparameter	**Value**
Feature dimension C	2048
Stride for energy-based scoring s_{score}	96
Stride for patch tokenization s_{patch}	48
Initial temperature β	1.0
Temperature range	$[0.01, 50]$
Softplus sharpness τ	20
Score contrast γ	2
Score floor ε	10^{-6}
mLSTM token embedding dimension C'	1024
mLSTM encoder blocks	2
mLSTM heads	4
mLSTM head dimension d_h	64
mLSTM FFN expansion ratio	$4\times$
mLSTM drop-path	0.07
Weight decay	2×10^{-4}
Scoring prototypes learning rate	10^{-5}
mLSTM learning rate	10^{-5}
Classification head learning rate	10^{-5}
Number of prototypes	10

Table 7: Classification performance of mLSTM-based aggregation and full pairwise self-attention on DHMC at different token budgets. Results for 700, 1,544, and 6,144 tokens are averaged over three runs. The 21,824-token mLSTM result is from a single run. Self-attention ran out of GPU memory at this token count.

Mean tokens / slide	**mLSTM**			**Self-attention**		
	Acc	F1	AUC	Acc	F1	AUC
700	0.875 $\pm$0.016	0.841 $\pm$0.024	0.977 $\pm$0.011	0.848 $\pm$0.062	0.826 $\pm$0.040	0.978 $\pm$0.008
1,544	0.837 $\pm$0.025	0.816 $\pm$0.046	0.971 $\pm$0.007	0.881 $\pm$0.027	0.857 $\pm$0.022	0.983 $\pm$0.006
6,144	0.857 $\pm$0.045	0.851 $\pm$0.025	0.979 $\pm$0.010	0.841 $\pm$0.072	0.837 $\pm$0.062	0.977 $\pm$0.008
21,824	0.906	0.861	0.982	–	–	–