

EvoPS: Evolutionary Patch Selection for Whole Slide Image Analysis in Computational Pathology

Saya Hashemian, Azam Asilian Bidgoli

Department of Computer Science and Physics, Wilfrid Laurier University, ON, Canada

Abstract

In computational pathology, the gigapixel scale of Whole-Slide Images (WSIs) necessitates their division into thousands of smaller patches. Analyzing these high-dimensional patch embeddings is computationally expensive and risks diluting key diagnostic signals with many uninformative patches. Existing patch selection methods often rely on random sampling or simple clustering heuristics and typically fail to explicitly manage the crucial trade-off between the number of selected patches and the accuracy of the resulting slide representation. To address this gap, we propose EvoPS (Evolutionary Patch Selection), a novel framework that formulates patch selection as a multi-objective optimization problem and leverages an evolutionary search to simultaneously minimize the number of selected patch embeddings and maximize the performance of a downstream similarity search task, generating a Pareto front of optimal trade-off solutions. We validated our framework across four major cancer cohorts from The Cancer Genome Atlas (TCGA) using five pretrained deep learning models to generate patch embeddings, including both supervised CNNs and large self-supervised foundation models. The results demonstrate that EvoPS can reduce the required number of training patch embeddings by over 90% while consistently maintaining or even improving the final classification F_1 -score compared to a baseline that uses all available patches' embeddings selected through a standard extraction pipeline. The EvoPS framework provides a robust and principled method for creating efficient, accurate, and interpretable WSI representations, empowering users to select an optimal balance between computational cost and diagnostic performance.

Keywords: Computational Pathology, Evolutionary Computation, Patch Selection, Whole Slide Images, Multi-Objective Optimization

1. Introduction

The field of digital pathology has fundamentally transformed the practice of medicine by digitizing traditional glass slides into high-resolution WSIs [1]. This transformation is critical for modern oncology, as it enables the development of artificial intelligence (AI) tools that analyze tissue morphology with unprecedented precision and scale. By leveraging these digital records, computational methods can improve the accuracy and reproducibility of cancer diagnosis, provide quantitative metrics for prognosis, and predict patient responses to specific therapies, thereby paving the way for personalized medicine [2].

While computational pathology has demonstrated significant potential, it faces a fundamental challenge related to the sheer scale of WSIs. At gigapixel resolution, a single WSI is far too large to be processed directly by standard deep learning models. To overcome this computational limitation, the standard approach is to divide each WSI into a collection of thousands of smaller, manageable tiles, known as patches [3]. This strategy transforms the problem from a single-image comparison into a more complex task of comparing large sets of patches, often framed within a Multiple Instance Learning (MIL) paradigm [4].

Adopting a patch-based strategy, however, introduces a significant challenge of scale. A single WSI is typically represented by thousands of high-dimensional embedding vectors, one for each patch. This data explosion creates several critical downstream challenges, including prohibitive memory and computational resource requirements, as well as reduced model interpretability, which is a key obstacle for Explainable AI (XAI) [5]. This also creates a significant risk of diagnostic signals from critical regions being diluted by a majority of uninformative patches [6]. This issue necessitates a method for intelligently reducing the number of patches to a smaller, more informative subset. The logical solution is, therefore, an effective patch selection strategy.

Patch selection [7] is a powerful way to enhance computational efficiency and model performance by focusing on the most informative regions of high-resolution images. Existing methods can be broadly categorized based on when the selection occurs in the computational pipeline: during preprocessing, as an integrated part of model training, or as a post-processing step on

extracted features.

The first category, **preprocessing-based selection**, aims to identify relevant regions directly from pixel data. This includes traditional heuristics that use saliency detectors to identify visually conspicuous areas [8], as well as unsupervised techniques like clustering on visual features (e.g., color, texture) to create a morphologically representative subset [9]. More recent methods in this category use anomaly detection to prune uninformative regions, either by creating a normal atlas to discard normal tissue [10] or by using generative models to select likely abnormal patches [11].

The second and largest category, **integrated selection**, incorporates the selection process directly into the model’s training loop. The most prominent of these are attention-based Multiple Instance Learning (MIL) models, where the network learns to assign importance scores to patches [12]. Novel attention mechanisms, such as Patch-to-Cluster (PaCa) attention, further refine this by grouping similar patches to reduce complexity [13]. Another powerful end-to-end approach is differentiable patch selection, which uses techniques like the Gumbel-Softmax trick to enable the network to learn *what* to select as part of the main training process [14]. Other cutting-edge strategies include using Reinforcement Learning (RL), where an agent like AgentViT is trained to sequentially select patches [15, 16]; using hierarchical distillation to pre-screen and eliminate irrelevant patches to accelerate inference [17]; leveraging multi-scale attention maps from low-resolution views to guide high-resolution patch selection [18]; and employing hybrid strategies that combine attention with clustering to find minimal patch sets [19].

A third category, which our work belongs to, is **post-embedding selection**. In this paradigm, feature embeddings are first extracted for a large set of candidate patches, and the selection process is then performed globally on this feature set. While the previously mentioned methods are powerful, they are typically designed to find a single, optimal subset guided by a single objective (e.g., accuracy). They do not provide a spectrum of solutions that would allow a user to select a model based on their specific needs—whether for a rapid, efficient screening or a more computationally intensive, high-confidence diagnosis. Our work directly addresses this gap.

The challenge of patch selection can be naturally formulated as a Multi-Objective Optimization Problem (MOP) [20]. On one hand, the goal is to maximize diagnostic accuracy, which requires identifying a sufficient set of the most informative patches. On the other hand, a competing goal is to minimize the number of selected patches to reduce computational costs,

improve model interpretability, and mitigate the effects of noisy or irrelevant data. These two objectives are inherently in conflict: an overly aggressive selection may discard diagnostically critical information, while a lenient one increases the computational burden. Therefore, an ideal solution is not a single patch subset, but rather a spectrum of solutions that represent the optimal trade-offs between patch-set size and performance.

Evolutionary Computation (EC) [21] offers a powerful paradigm for solving such MOPs. In medicine and bioinformatics, EC has been successfully applied to a range of complex tasks due to its ability to navigate vast search spaces. For example, in medical imaging, EAs are used to optimize the segmentation of tumors and organs from complex scans [22]. In the field of genomics, they are employed to solve feature selection problems, identifying small, informative subsets of genes from high-dimensional data for cancer classification [23]. Furthermore, in clinical oncology, they are critical for optimizing radiation therapy plans, where algorithms must balance the conflicting objectives of maximizing the dose to the tumor while minimizing exposure to surrounding healthy tissues [24]. A key advantage of modern, Pareto-based EC methods is their ability to handle each objective independently, rather than collapsing them into a single scalar value. By evolving a population of candidate solutions in parallel, these algorithms are uniquely capable of discovering a diverse set of trade-off solutions that approximate the true Pareto front in a single run [25]. Building on these principles, this paper makes several key contributions. This is the first work to formally define histopathology patch selection as a multi-objective optimization problem. We introduce **EvoPS (Evolutionary Patch Selection)**, a novel framework that operates on pre-extracted patch embeddings and leverages an evolutionary algorithm to find an optimal balance between the competing objectives of patch count and diagnostic accuracy. A primary contribution of this work is a large-scale validation of the EvoPS framework across four major cancer cohorts from The TCGA, using embeddings extracted from five distinct deep learning backbones, spanning both supervised CNNs and self-supervised foundation models. Ultimately, EvoPS generates a Pareto front of solutions for each setting, providing a principled method for selecting an optimal patch subset that fits a specific computational budget or accuracy requirement. The remainder of this paper is organized as follows. Section 2 reviews relevant background. Section 3 presents the proposed EvoPS framework in detail, including its formulation and optimization strategy. Section 4 describes the experimental setup. Section 5 reports and discusses the re-

sults, with comparisons to baseline and state-of-the-art approaches. Finally, Section 6 concludes the paper.

2. Background

Our proposed EvoPS framework is built upon the principles of multi-objective optimization and evolutionary computation. This section provides a brief overview of the core concepts that underpin our methodology.

Most real-world optimization problems involve not one, but several competing objectives that must be optimized simultaneously [26]. A Multi-Objective Optimization Problem (MOP) is one where the goal is to find a set of parameters that optimizes multiple functions at once. Unlike single-objective optimization, MOPs typically do not have a single best solution because the objectives are often in conflict; improving one objective may lead to the degradation of another. For example, in engineering design, one might want to minimize the cost of a product while simultaneously maximizing its performance and durability. Therefore, the goal in a MOP is not to find a single superior solution, but rather a set of solutions that represent the optimal trade-offs between the objectives.

To formally compare solutions and identify these optimal trade-offs, the concept of Pareto dominance is used. We consider a minimization problem with k objectives, where a solution $\mathbf{x}$ is evaluated by a vector of objective functions $\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_k(\mathbf{x}))$. A solution $\mathbf{x}_A$ is said to *dominate* another solution $\mathbf{x}_B$ (denoted as $\mathbf{x}_A \prec \mathbf{x}_B$) if and only if two conditions are satisfied:

$$\begin{aligned} \forall i \in \{1, \dots, k\} : & \quad f_i(\mathbf{x}_A) \leq f_i(\mathbf{x}_B) \\ \exists j \in \{1, \dots, k\} : & \quad f_j(\mathbf{x}_A) < f_j(\mathbf{x}_B) \end{aligned}$$

In other words, $\mathbf{x}_A$ is better or equal in all objectives and strictly better in at least one. A solution is called *Pareto optimal* (or non-dominated) if no other solution in the search space dominates it. The set of all such solutions is known as the *Pareto set*, and its projection onto the objective space forms the *Pareto front*, which represents the best possible trade-offs [27]. Non-dominated sorting is the algorithmic procedure used to rank a population of candidate solutions based on this concept. It works by iteratively identifying fronts, where the first front (F_0) consists of all non-dominated individuals in the population.

Beyond ranking solutions, a successful algorithm must also ensure they are well-distributed. To achieve this, a diversity preservation mechanism is employed, such as the *crowding distance*, which measures the density of solutions surrounding a particular point on the front. Influential evolutionary algorithms, such as NSGA-II, effectively combine non-dominated sorting with a crowding distance mechanism to guide the search towards a well-distributed and optimal Pareto front [25].

These concepts are integrated into the general workflow of a Multi-Objective Evolutionary Algorithm (MOEA) [28]. An MOEA maintains a *population* of candidate solutions, known as *individuals*, and evolves them over generations. For our problem, an individual represents a specific subset of patches. The general workflow follows an iterative cycle:

1. **Initialization:** An initial population of diverse, random individuals is generated.
2. **Fitness Evaluation:** The fitness of each individual is evaluated against the multiple objectives.
3. **Selection:** The best-performing individuals are selected to become parents based on their Pareto rank and diversity.
4. **Variation:** New offspring solutions are created from parents using genetic operators like crossover and mutation.
5. **Replacement:** The new offspring population replaces the previous generation, and the cycle repeats, with the population expected to converge towards the true Pareto front.

3. Methodology

This section details our proposed method for optimizing patch selection in WSIs for histopathological classification. A single WSI can contain thousands of patches, making it computationally expensive and memory-intensive to process them all. To address this challenge, our approach focuses on selecting only the most informative patches based on their *feature embeddings*, rather than using every available patch. The primary goal is to reduce memory and computational requirements while maintaining high diagnostic performance.

As illustrated in Figure 1, our pipeline begins with standardized WSI pre-processing and patch extraction, followed by feature embedding using several

deep learning backbones. The core of our methodology is a multi-objective optimization framework for patch selection, termed EvoPS, which identifies optimal subsets of patches by simultaneously minimizing the total patch count and maximizing classification performance (F_1 -score). The pipeline includes:

- (A) Original WSIs are first tiled into a set of patches. These patches are then processed by a pre-trained deep learning backbone to extract a fixed-dimensional feature embedding for each one.
- (B) The core of our method is to formulate the patch selection problem as a combinatorial multi-objective optimization problem and solve it using an evolutionary algorithm. A population of individuals, where each individual is a binary vector representing a subset of patches, is evolved over generations. Using genetic operators, new candidate solutions are created and evaluated based on two competing objectives: minimizing the number of patches and maximizing the F_1 -score. A selection mechanism then determines which solutions survive to the next generation.
- (C) The output of EvoPS is a Pareto front, which contains a set of optimal solutions that represent the best possible trade-offs between the two objectives. Each point on the front corresponds to a specific, sparse subset of selected patches, allowing a user to choose a final solution based on their desired balance between data efficiency and performance.

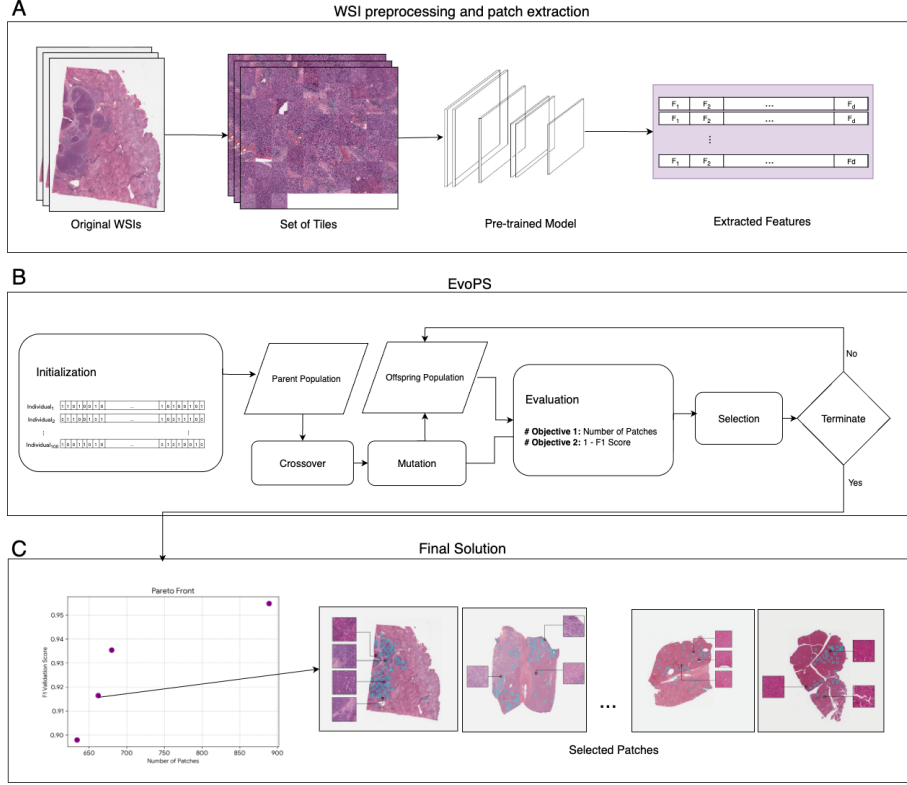


Figure 1: An overview of the three-stage EvoPS pipeline, consisting of (A) WSI preprocessing and feature extraction, (B) the EvoPS framework for multi-objective patch selection, and (C) the generation of a final Pareto front of optimal patch-subset solutions.

3.1. Dataset and Feature Extraction

Our study is based on a WSIs from The Cancer Genome Atlas (TCGA) [29], which focuses on four major cancer categories including Gastro, Pulmonary, Liver, and Mesenchymal, each composed of several distinct sub-types. Our selected cohort totaling 2,587 WSIs, was partitioned into training (2,091 WSIs; 80.8%), validation (245 WSIs; 9.5%), and test (251 WSIs; 9.7%) sets. Our EvoPS framework was applied independently to each of the four categories to discover a tailored subset of optimal patches. The detailed composition and split for each cancer sub-type are presented in Table 1. To computationally analyze a WSI, it is first necessary to perform patch extraction, a process where the gigapixel image is divided into a grid of smaller tiles from which feature embeddings can be extracted. To ensure this ini-

Table 1: Detailed Composition of the TCGA Dataset by Sub-Type for Training, Validation, and Test Sets

Category	Sub-Type	Train	Val	Test
Gastro	Colon Adenocarcinoma (COAD)	269	31	32
	Rectum Adenocarcinoma (READ)	106	9	12
	Esophageal Carcinoma (ESCA)	103	14	14
	Stomach Adenocarcinoma (STAD)	223	27	30
Pulmonary	Lung Adenocarcinoma (LUAD)	317	38	38
	Lung Squamous Cell Carcinoma (LUSC)	363	41	43
	Mesothelioma (MESO)	51	5	5
Liver	Cholangiocarcinoma (CHOL)	27	4	4
	Liver Hepatocellular Carcinoma (LIHC)	294	35	35
	Pancreatic Adenocarcinoma (PAAD)	110	12	12
Mesenchymal	Uveal Melanoma (UVM)	24	4	4
	Skin Cutaneous Melanoma (SKCM)	204	25	22
Total		2,091	245	251

tial set of patches is both manageable and diagnostically representative, we employ the Yottixel algorithm [30] as our intelligent sampling strategy. The Yottixel process begins by generating a robust tissue mask from a down-sampled slide thumbnail. This step involves a color space transformation to Hematoxylin-Stain-Deconvolution (HSD), which effectively isolates stained tissue from the background. Following masking, the algorithm proceeds with a two-stage clustering approach. First, all candidate patches within the tissue region are grouped by color similarity using K-Means on their mean RGB values. Second, within each of these color clusters, a spatially diverse subset is selected using another K-Means on the patch coordinates. This two-stage process ensures that the final selection of patches represents both the varied morphology and the spatial distribution of the tissue. All selected patches are then standardized to a physical size of 1000×1000 pixels at an equivalent of 20× magnification.

Following preprocessing, the selected patches for each WSI are organized into a Mosaic which is a structured file containing their coordinates and metadata. Mosaic serves as a consistent input for extracting feature vectors

using five distinct deep learning backbones. The evaluated models span both supervised and self-supervised paradigms and include two supervised Convolutional Neural Networks (CNNs), KimiaNet [31] and DenseNet [32], which both produce 1024-dimensional vectors and three large-scale foundation models: PHIKONv2 [33] (1024 dimensions), Virchow2 [34] (1280 dimensions) and UNI-2H [35] (1536 dimensions).

3.2. *EvoPS: Evolutionary Patch Selection*

The core of our approach is EvoPS, a multi-objective evolutionary framework that discovers a subset of patches’ embeddings from the training set that optimally trades off the number of selected patches with classification error. The patch selection search is performed exclusively on the embeddings of training set; EvoPS determines which training patches to retain while the validation set remains unchanged. All validation slides keep their full set of patches’ embeddings and are used solely to evaluate how well the selected training subset generalizes. In this way, EvoPS identifies a compact yet informative subset of training patches that yields strong classification performance on unseen data.

Problem Formulation and Objective Functions: An individual solution is represented by a binary vector $\mathbf{x} \in \{0, 1\}^P$, where P is the total number of patches across all training slides. For each patch i , a value of $x_i = 1$ indicates the patch is selected, while $x_i = 0$ indicates it is discarded. The vector is structured so that patches from the same WSI occupy a contiguous block, allowing for efficient slide-level operations. EvoPS simultaneously optimizes two competing objectives for each individual solution $\mathbf{x}$:

1. Number of Patches (f_1): The primary goal is to use as few patches as possible. This objective directly minimizes the total count of selected patches, encouraging the algorithm to find smaller and more efficient solutions. It is defined as:

$$f_1(\mathbf{x}) = \frac{\sum_{i=1}^P x_i}{P}$$

which computes the proportion of selected patches in the individual (i.e., the fraction of training patches retained).

2. Classification Error (f_2): The second goal is to ensure the selected patches are diagnostically informative. We use a k -Nearest Neighbors

(k -NN) algorithm, a standard method for finding the most similar items in a dataset [36], to evaluate each solution. For a given individual x , each training slide is represented by averaging the embeddings of its selected patches. The performance is then evaluated on the validation set, where each validation slide is represented by averaging all of its patches. The k -NN algorithm finds the top k most similar training slides for each validation slide, and the performance is scored based on whether the labels of these neighbors match. This performance is formally measured using the weighted F_1 -score, which is derived from Precision and Recall:

For each class c , the F_1 -score is:

$$F_{1,c} = \frac{2 \cdot \text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c}.$$

Let n_c be the number of samples in class c and N the total number of samples. The Weighted F_1 is defined as:

$$F_{1,\text{weighted}} = \sum_{c=1}^C \frac{n_c}{N} \cdot F_{1,c}.$$

To define a minimization-minimization problem, our second objective is defined as one minus this weighted F_1 -score:

$$f_2(\mathbf{x}) = 1 - F_{1,\text{weighted}}(\text{validation set})$$

The evolutionary process evolves a fixed-size population of individuals over a set number of generations. The genetic operators are designed to always maintain a key constraint: every slide must be represented by at least one patch.

Initialization: Each individual solution in our framework is represented by a single, one-dimensional binary vector, as conceptually illustrated in Figure 2. This vector’s length is equal to the total number of patches across all training WSIs, with each position corresponding to a unique patch. A value of ‘1’ at a given position indicates that the corresponding patch is selected, while a ‘0’ means it is discarded. For example, consider the segment for WSI₁ in the figure. If this WSI is represented by a total of 5 patches, a possible

binary vector for this segment could be $[1, 0, 1, 1, 0]$. This would mean that for this particular slide, three patches from WSI_1 are selected (the first, third, and fourth), while two are not. This structure is maintained for all slides.

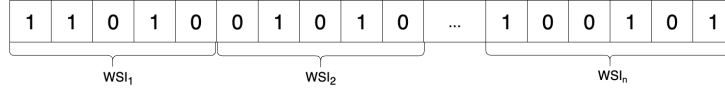


Figure 2: Conceptual illustration of an individual solution as a binary vector. Each segment (WSI_1 , WSI_2 , etc.) represents all patches from a single WSI. A '1' indicates a selected patch, while a '0' indicates a discarded patch.

A purely random process for creating the initial population risks generating invalid solutions where an entire WSI might accidentally be excluded (i.e., have no patches selected). To prevent this, for each WSI, one of its corresponding patches is randomly selected, and its value in the binary vector is assigned to 1. After ensuring every slide is represented in each individual, we introduce diversity by assigning a value of 1 to additional patches. For each individual, a random number of these extra patches is selected from a uniform distribution, and they are assigned to random locations across the entire vector. This process creates an initial population where every solution is valid and there is a wide, evenly distributed range of total patch counts.

Crossover: New offspring are generated by applying a uniform crossover operator to two parent solutions, where their binary vector information is exchanged on a gene-by-gene basis. However, this standard operation can inadvertently produce invalid offspring where an entire WSI is left with no selected patches. To prevent this, our process includes a “safe” repair mechanism to maintain the per-slide coverage constraint. After the initial gene exchange, each new offspring is validated. If an offspring is found to have no selected patches for a particular WSI (i.e., its corresponding vector segment contains all zeros), the entire segment for that slide is replaced by copying it from one of the two original parents, chosen at random. This repair step guarantees that all generated offspring are valid solutions.

Mutation: Mutation is applied to each new offspring to introduce random variations, which helps the algorithm explore the search space and prevent premature convergence. We employ a bit-flip mutation operator, where each bit in an individual’s binary vector has a small probability of being inverted (i.e., a 0 becomes a 1, or a 1 becomes a 0). However, this random

process could deactivate the last remaining selected patch for a particular slide, which would result in an invalid solution. To prevent this, the mutation process is also “safe” to ensure solutions remain valid. After mutation, each individual is validated to check that the per-slide coverage constraint is still met. If a mutation event causes a WSI’s segment to have no selected patches, the repair mechanism restores validity by randomly selecting one patch within that specific segment and assigning its value back to 1.

Fitness Evaluation: After the genetic operators create new offspring, each individual in the population is evaluated to determine its fitness. This evaluation is based on the two competing objectives defined previously: minimizing the number of selected patches and maximizing classification performance. The process for a given individual is as follows:

First, a slide-level training set is constructed from the selected patches. This is achieved by identifying all patches where the individual’s binary vector has a value of 1 and grouping them by their source WSI. For each slide that contains at least one selected patch, a single representative feature vector is calculated by averaging the embeddings of those selected patches.

This collection of averaged slide vectors then serves as a reference library for our similarity search algorithm. To determine the performance, each validation slide (represented by an average of all its patches) is used as a query. The algorithm searches the reference library to find the top k training slides with the most similar feature vectors—the nearest neighbors. The prediction for the validation slide is then determined by the labels of these neighbors. The final performance score is calculated as one minus the $F_{1,\text{weighted}}$. These two values—the patch count and the classification error—constitute the individual’s fitness.

Selection: To form the next generation, EvoPS employs an elitist selection mechanism that evaluates both parents and offspring together. The merged parent and offspring population is then sorted into a hierarchy of non-dominated fronts ($F_0, F_1, F_2, \dots$). Individuals on the first front (F_0) represent the best current trade-off solutions, as no other solution in the population is superior to them across both objectives. To maintain diversity, a crowding distance metric is then calculated for individuals within the same front, favoring those that are more isolated from their neighbors.

Finally, the new population is built by selecting individuals based on this two-level criterion: solutions with a better non-domination rank are always chosen first. If a choice must be made between individuals of the same rank, those with a greater crowding distance are prioritized. This selection

Algorithm 1 The EvoPS Algorithm

Input: Dataset D , Backbone Model B

Output: Pareto Front F_{pareto} , Performance Scores S

```
procedure EvoPS( $D, B$ )
  // Phase 1: Data Preparation
  Patches  $P \leftarrow$  YottixelPatchExtraction( $D$ )
  Embeddings  $E \leftarrow$  FeatureExtraction( $P, B$ )
  // Phase 2: Evolutionary Optimization
  Population Pop  $\leftarrow$  InitializePopulation( $E$ )
  for  $g \leftarrow 1$  to NumOfGeneration do
    // Crossover
    Offspring  $\leftarrow$  Crossover(Pop)
    // Mutation
    MutatedOffspring  $\leftarrow$  Mutation(Offspring)
    Combined C  $\leftarrow$  Pop  $\cup$  MutatedOffspring
    // Calculate objectives for all individuals
    EvaluateFitness(C)
    // Selection
    Pop  $\leftarrow$  SelectSurvivors
  end for
  // Phase 3: Final Evaluation
   $F_{pareto} \leftarrow$  GetNonDominatedFront(Pop)
   $S \leftarrow$  EvaluateOnTestSet( $F_{pareto}$ )
  return  $F_{pareto}, S$ 
end procedure
```

continues until the population for the next generation is filled.

3.3. Final Performance Evaluation

Once the EvoPS optimization is complete, the quality of the solutions on the final Pareto front is assessed on a held-out test set. This final step measures the real-world performance of the selected patch subsets on entirely unseen data. For each solution on the Pareto front, a sparse training set is created using only the patches indicated by its binary vector. Our similarity search algorithm is then trained on this sparse set, where each training slide is represented by the average of its selected patches. The algorithm’s perfor-

mance is then recorded on the test set (where each test slide is represented by an average of all its patches). For an overall view of the EvoPS method, the complete pipeline is summarized in Algorithm 1.

Algorithm 2 The EvaluateFitness Function

```

function EVALUATEFITNESS(Population  $C$ , Embeddings  $E$ )
  for all individual  $\in C$  do
    // Objective 1: Number of Patches
     $obj_1 \leftarrow$  count of 1s in individual.vector
    // Objective 2: Classification Error
    aggregated_slide_vectors  $\leftarrow$  Aggregation(individual,  $E.train\_set$ )
     $F_1\_score \leftarrow$  SimilaritySearch(aggregated_slide_vectors,  $E.validation\_set$ )
     $obj_2 \leftarrow 1.0 - F_1\_score$ 
    individual.fitness  $\leftarrow (obj_1, obj_2)$ 
  end for
end function

```

4. Experimental Setup

This section details the specific configurations, hyperparameters, and implementation details used to validate our EvoPS framework.

4.1. Dataset and Feature Extraction

The TCGA dataset was partitioned into training, validation, and test sets as Presented in Table 1. The fitness of individuals during the evolutionary process was evaluated using the validation set, while the final performance of the solutions was measured on the held-out test set.

Patch extraction for all WSIs was performed using the Yottixel algorithm, which was configured with a tissue coverage threshold of 0.80. The algorithm was set to initially group patches into $k = 9$ color clusters and select the final 8% of patches from each cluster via spatial sampling. All patches were standardized to a physical size of 1000×1000 pixels at $20\times$ magnification. Feature embeddings were extracted for these mosaics using the five backbones described previously: KimiaNet, DenseNet, PHIKONv2, Virchow2, and UNI-2H. These models were selected to provide a comprehensive evaluation across different types of DNN architectures and training

paradigms. KimiaNet is a CNN specifically trained on TCGA whole-slide images, making it well aligned with histopathology domain characteristics. DenseNet, in contrast, is a widely used CNN pretrained on ImageNet and serves as a representative of general-purpose feature extractors. The remaining three models, PHIKONv2, Virchow2, and UNI-2H, are foundation models for computational pathology that learn from extremely large-scale histopathology datasets.

4.2. EvoPS Configuration

As mentioned earlier, EvoPS is applied independently for each organ class (i.e., category). In other words, the patch selection process is performed separately for each organ, allowing the evolutionary search to identify organ-specific discriminative subsets of patches rather than enforcing a shared selection across all classes. The multi-objective evolutionary algorithm at the core of EvoPS was configured with a population size of 100 individuals evolved over 50 generations. The genetic operators included a safe uniform crossover with a swap probability of $p = 0.9$ and a safe bit-flip mutation with a probability of $p = 0.01$ per gene. The fitness evaluation’s similarity search was performed using a k -Nearest Neighbors algorithm with $k = 5$.

All experiments were conducted on a system equipped with a GPU, leveraging CUDA or MPS for hardware acceleration where available. To ensure the robustness of our findings, each of the 20 experimental settings (4 TCGA cohorts $\times$ 5 feature backbones) was repeated 10 times with different random seeds. The final reported results are the averaged outcomes of these 10 independent runs, with the primary evaluation metric being the weighted F_1 -score.

For performance evaluation, we report three settings:

- **Baseline:** The performance achieved using all available patches from the training set.
- **EvoPS (Best Val):** The solution from the Pareto front that achieved the highest F_1 -score on the validation set, and its corresponding performance on the test set. This represents the most practical application of our method.
- **EvoPS (Best Test):** The solution from the Pareto front that achieved the highest F_1 -score on the test set, representing the framework’s optimal potential.

For all experiments, we report two primary evaluation metrics: (i) the number of selected patches from the training set, and (ii) the resulting F_1 -score of the corresponding solution on the Pareto front. These metrics jointly quantify the trade-off between compactness of the selected patch set and classification performance.

5. Results Analysis

To evaluate the performance of our EvoPS framework, we conducted experiments across four TCGA cohorts using five distinct feature-extraction backbones. The results are summarized in Table 2. These findings reveal a crucial trade-off between training data efficiency and classification performance. The tables present three key scenarios for comparison including Baseline, EvoPS (Best Val), and EvoPS (Best Test).

A primary and universal outcome of applying EvoPS is a drastic reduction in the number of training patches required. As detailed in Table 2, the framework consistently reduced the patch set size by 85% to over 98% across all backbones and cohorts. For instance, with the Virchow2 backbone on the Mesenchymal cohort, EvoPS identified a critical subset of patches that was 98.22% smaller than the baseline set. This demonstrates the framework’s profound capability to enhance computational and data efficiency by isolating the most informative regions within whole slide images.

This efficiency gain, however, introduces a nuanced trade-off with classification accuracy. In several scenarios, EvoPS achieves the ideal outcome: improving performance with significantly less data. A notable success is observed with the DenseNet backbone on the Liver cohort, where a 90.86% patch reduction led to a substantial F_1 -score increase of +4.92 points over the baseline. Similarly, KimiaNet on the Pulmonary cohort improved its F_1 -score by +0.79 points while using 95.22% fewer patches. These results suggest that by removing redundant or uninformative patches, EvoPS can act as a data regularizer, helping the model to learn more generalizable features.

Conversely, the results also highlight that aggressive patch selection is not universally beneficial and its effectiveness is highly dependent on the feature-extraction backbone. The UNI2H model, for example, consistently exhibited a performance decline across all four cohorts after patch selection, with F_1 -score drops ranging from -2.77 to -4.06 points. This suggests that some architectures may be more sensitive to data reduction, perhaps requiring a denser sampling of patches to build robust representations. In contrast,

models like DenseNet and KimiaNet appear more adept at generalizing from the compact, curated subsets identified by EvoPS. Overall, our findings indicate that EvoPS is a powerful tool for optimizing the data-to-performance trade-off in computational pathology, though its application is most effective when tailored to the specific characteristics of the model backbone.

Table 2: Performance summary of EvoPS. The number of patches is reported for both the **baseline** (i.e., using all training patches) and the EvoPS-selected subsets corresponding to the two solutions: Best Validation and Best Test on the Pareto front. Similarly, the F_1 -scores on the validation and test sets are reported for both the baseline and the EvoPS-selected patch subsets. This allows us to directly compare the performance and compactness of the selected patch sets against using all available patches. Bolded values indicate the best result for each comparison: the lowest patch count, the highest validation F_1 -score, and the highest test F_1 -score. The final row summarizes the win/loss/tie record of EvoPS against the Baseline on the best test F_1 -score across all 20 experiments.

Backbone	Cohort	Number of Patches (Training Set)			F1-Score (%)				
		Baseline	EvoPS		Validation		Test		
			Best Val	Best Test	Baseline	EvoPS	Baseline	EvoPS	
								@ Best Val	Best
				(Best)					
KimiaNet	Gastro	19,916	2,123	1,397	74.20	84.04	73.78	74.20	77.19
	Liver	16,331	856	766	91.65	95.48	88.44	87.96	88.34
	Mesenchymal	7,079	343	343	100.0	100.0	91.19	91.19	91.19
	Pulmonary	110,684	5,288	3,557	90.21	97.62	83.26	84.05	86.54
DenseNet	Gastro	19,916	2,614	1,450	50.81	72.45	55.53	51.58	58.57
	Liver	16,331	1,493	1,038	85.69	96.27	77.56	82.48	84.07
	Mesenchymal	7,079	438	348	92.09	100.0	88.63	86.53	88.18
	Pulmonary	110,684	15,273	8,331	66.91	82.87	59.80	64.67	69.61
Phikon V2	Gastro	11,861	1,291	565	70.76	83.65	59.27	57.50	62.76
	Liver	117,072	5,613	3,854	88.60	97.71	81.00	83.08	86.09
	Mesenchymal	52,485	1,624	1,407	96.58	100.0	93.64	94.57	95.66
	Pulmonary	153,388	16,773	11,379	65.87	79.87	76.03	69.86	72.98
Virchow2	Gastro	11,861	1,128	695	70.92	84.65	70.61	65.83	71.28
	Liver	117,072	3,533	3,453	93.73	97.91	92.44	94.57	95.12
	Mesenchymal	52,485	936	806	96.58	100.0	100.0	98.90	99.27
	Pulmonary	153,388	13,894	9,096	84.76	96.15	78.41	80.42	82.57
UNI2H	Gastro	11,861	1,909	848	75.54	84.57	70.95	66.89	70.08
	Liver	117,072	4,573	3,335	95.97	97.94	95.47	92.19	94.80
	Mesenchymal	52,485	1,010	838	96.58	100.0	100.0	96.71	97.07
	Pulmonary	153,388	22,656	7,502	84.38	96.49	80.99	78.22	83.32
Summary (EvoPS vs. Baseline)							12 Wins / 7 Losses / 1 Tie		

To quantify the data reduction achieved by EvoPS, we analyzed the average number of patches selected per WSI across the 10 experimental runs. Table 3 compares the average number of patches per WSI in the baseline training sets versus the optimized subsets found by EvoPS for each backbone. The results demonstrate a dramatic and consistent reduction in data across all models and cohorts. For instance, in the Pulmonary cohort, EvoPS reduced the average patch count from over 200 per WSI in the baseline to as

few as 9 for the foundation models. This represents a data reduction of over 95%, highlighting the framework’s ability to identify a highly compact and efficient subset of patches for downstream tasks.

Table 3: Number of Patches per WSI for Baseline vs. EvoPS. Note the two different baseline patch sets for KimiaNet/DenseNet vs. the three foundation models.

Cohort	Sub-Type	Baseline		EvoPS (# of Patches per WSI)				
		Kimia/Dense	Found. Models	KimiaNet	DenseNet	PhikonV2	UNI2H	Virchow2
Gastro	COAD	28.94	149.93	2.14	2.21	7.27	10.83	8.77
	READ	24.38	101.70	1.90	2.03	5.25	7.56	6.43
	ESCA	37.89	249.68	2.87	2.58	11.37	17.54	14.16
	STAD	36.55	187.33	2.42	2.51	8.84	13.27	10.87
Liver	CHOL	48.54	382.25	2.10	2.68	12.42	10.47	10.90
	LIHC	40.15	271.76	1.91	2.52	8.90	7.80	7.98
	PAAD	34.58	244.14	1.79	2.37	8.21	6.92	7.39
Pulmonary	LUAD	161.17	213.46	5.24	12.13	16.05	10.78	12.75
	LUSC	165.82	212.60	5.48	12.37	16.05	10.79	12.69
	MESO	23.54	167.60	1.65	2.54	12.82	8.90	10.15
Mesenchymal	UVM	25.60	114.79	1.39	1.56	3.82	2.54	2.36
	SKCM	31.37	243.77	1.50	1.58	6.24	4.18	3.67

To visually illustrate the impact of our framework, Figure 3 presents a comprehensive comparative analysis across all five feature backbones. The figure is organized into a 2×5 grid, where each column is dedicated to a specific backbone. The top row of subplots directly compares the number of training patches used by the baseline versus the optimized subset selected by EvoPS. The bottom row correspondingly illustrates the resulting performance on the test set, comparing the F_1 -scores for both approaches. This consolidated view effectively demonstrates the trade-off between data efficiency and diagnostic accuracy, allowing for a clear assessment of the framework’s impact on each model and cohort.

To provide a more granular view of classification performance, we analyzed the confusion matrices for the UNI2H backbone. Figure 4 presents a side-by-side comparison for each of the four cancer cohorts. The top row shows the confusion matrix for the Baseline model (trained on all patches), while the bottom row shows the average confusion matrix from the 10 independent EvoPS runs. This visualization allows for a detailed analysis of class-specific performance, highlighting how EvoPS impacts the misclassification patterns between different cancer sub-types.

The analysis reveals that EvoPS can significantly refine decision boundaries between histologically similar classes. The most notable example is in the Gastro cohort, where the baseline model frequently misclassified Rectum

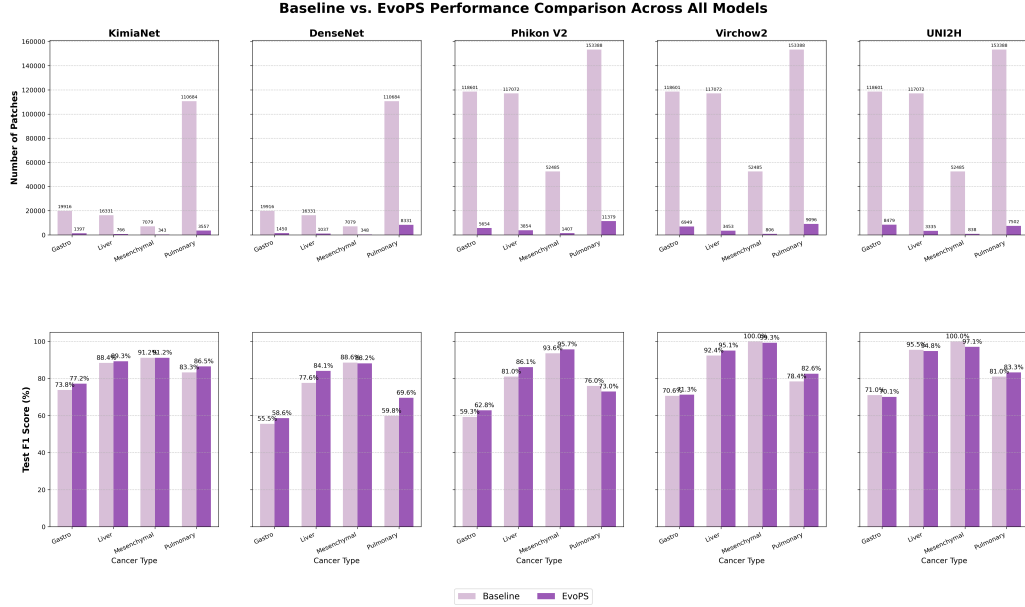


Figure 3: A Comprehensive performance comparison between the Baseline and EvoPS across all five feature backbones. The top row of plots details the number of training patches, illustrating the significant reduction achieved by EvoPS. The bottom row presents the corresponding Test F₁-scores for each cohort. Each column is dedicated to a specific deep learning model, from left to right: KimiaNet, DenseNet, Phikon V2, Virchow2, and UNI2H.

Adenocarcinoma as Colon Adenocarcinoma (13 instances). EvoPS reduced this specific confusion by over 34%, correctly reassigning several cases. Similarly, in the Pulmonary cohort, EvoPS reduced the confusion between Lung Squamous Cell Carcinoma and Lung Adenocarcinoma.

However, the results also show the nuances of patch selection. For cohorts with fewer, more distinct classes like Mesenchymal, the baseline performance was already near-perfect. Here, the aggressive patch reduction by EvoPS maintained high accuracy on the majority class (Skin Cutaneous Melanoma) but introduced minor misclassifications for the rare class (Uveal Melanoma). This indicates that while EvoPS is highly effective at reducing inter-class confusion in complex, multi-class scenarios, its benefits may be less pronounced for simpler classification tasks where baseline performance is already saturated.

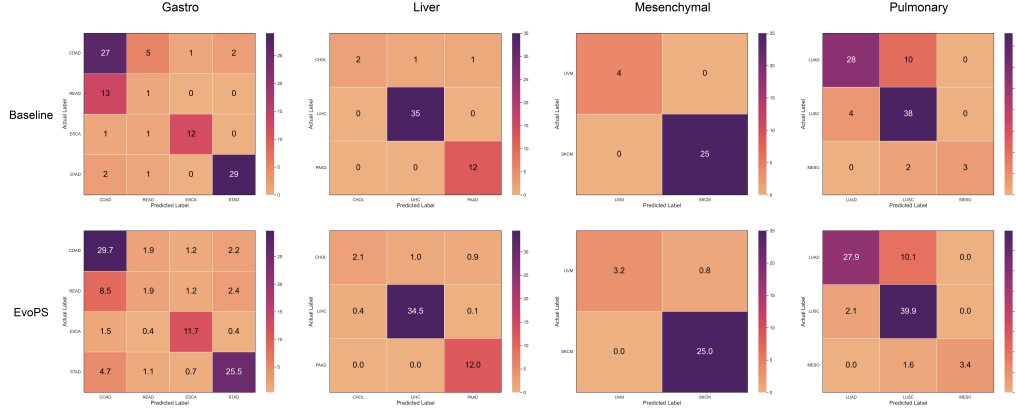


Figure 4: Confusion matrix comparison for the UNI2H model. The performance of the baseline model is detailed in the top row, whereas the performance of the EvoPS framework is illustrated in the bottom row. The results are categorized by tissue type, with each column representing Gastro, Liver, Mesenchymal, and Pulmonary, respectively.

5.1. Feature Space Visualization

To qualitatively assess the impact of our framework on the feature space, we visualized the patch embeddings using t-SNE, as shown in Figure 5. This figure presents a direct visual comparison in a 4×2 grid. The left column displays the t-SNE projections for the Baseline (using all patches), while the right column shows the projections for the optimized subsets selected by EvoPS. Each row corresponds to one of the four cancer cohorts: Liver, Mesenchymal, Pulmonary, and Gastro. Across all four cohorts, the t-SNE visualizations reveal a consistent and significant improvement in the feature space structure with the patches selected by EvoPS. In general, the EvoPS subsets produce clusters that are visibly denser, more compact, and more clearly separated than those from the Baseline. This effect is most striking in the Liver and Mesenchymal cohorts. For the Liver cohort, the three cancer sub-types, which are broadly dispersed in the Baseline, resolve into three distinct and well-defined clusters with the EvoPS-selected subset. Similarly, for the Mesenchymal cohort, the Uveal Melanoma sub-type, initially a loose collection of points within the larger Skin Cutaneous Melanoma cloud, is condensed by EvoPS into tight, highly separable clusters. Even in the more challenging Gastro and Pulmonary cohorts, where sub-types exhibit initial overlap, our method imposes more structure by pruning outlier patches, resulting in more compact cluster cores and better visual separation.



Figure 5: t-SNE visualization comparing the patch embedding distributions for the Baseline versus EvoPS frameworks. The left column displays the feature space using all available patches (Baseline), while the right column displays the feature space using only the patch subset selected by EvoPS. Each row corresponds to a different cancer cohort, from top to bottom: Gastro, Pulmonary, Liver, and Mesenchymal, respectively.

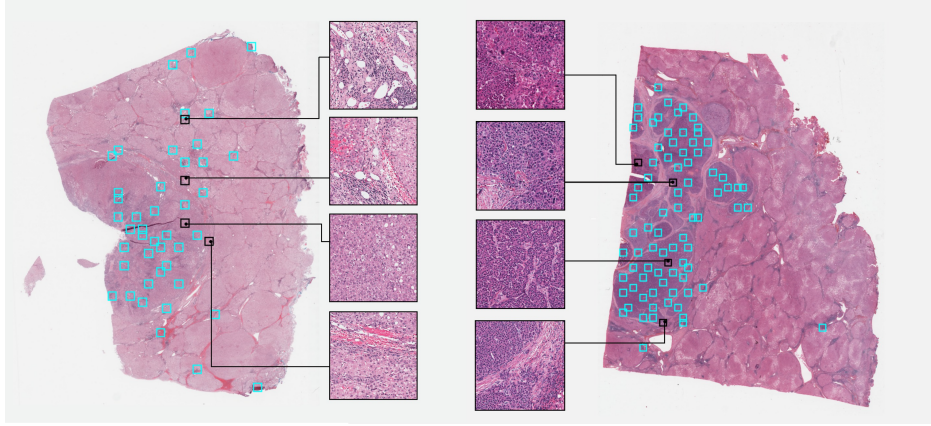


Figure 6: Visualizing EvoPS patch selection for a representative WSI. Blue squares indicate all patches extracted during preprocessing. Black squares highlight the sparser, diagnostically informative set of patches selected by EvoPS for a specific Pareto-optimal solution. Zoomed views show histological details of selected regions.

These qualitative results strongly support our quantitative findings, visually demonstrating that EvoPS successfully identifies a core set of the most discriminative patches while effectively discarding those that are redundant or lie in ambiguous regions of the feature space.

To visually illustrate the efficacy of EvoPS, Figure 6 presents representative examples of patch selection from two distinct WSIs. The blue squares indicate the initial set of all patches extracted during the WSI preprocessing stage. The black squares highlight the set of patches chosen by the EvoPS framework for a specific solution on the Pareto front. These black squares denote the patches identified as most informative by our multi-objective evolutionary algorithm, achieving a balance between minimizing the total patch count and maximizing diagnostic accuracy. The zoomed-in views provide a closer look at the histological details of the extracted EvoPS-selected patches, demonstrating the algorithm’s ability to focus on diagnostically relevant regions while drastically reducing the overall data volume. The visual examples demonstrate that the patches selected by EvoPS consistently localize to histopathologically informative regions. The highlighted patches correspond to areas containing tumor cell clusters, irregular glandular architecture, and dense stromal infiltration, which are characteristic morphological patterns associated with malignancy. In contrast, large homogeneous tissue regions, fatty tissue, and normal background structures are largely excluded.

6. Conclusion

In this paper, we addressed the significant challenge of scalability and information redundancy in patch-based analysis of Whole-Slide Images. We introduced EvoPS, a novel framework that formulates patch selection as a multi-objective optimization problem, leveraging an evolutionary algorithm to find an optimal balance between the number of patches and classification accuracy. Unlike prior approaches that return a single solution or rely on implicit importance scores, EvoPS explicitly constructs a Pareto front of optimal trade-offs, enabling flexible selection of patch subsets based on computational constraints or diagnostic performance requirements. Our comprehensive experiments across four distinct cancer cohorts and five different feature backbones consistently demonstrate the effectiveness of this approach. The results show that EvoPS can drastically reduce the required number of training patches, in many cases by over 90%, while simultaneously maintaining or even improving diagnostic performance on unseen test data. The primary significance of this work lies in its ability to generate a Pareto front of solutions, providing a principled and explicit map of the trade-off between computational efficiency and model accuracy. This empowers researchers and clinicians to select a patch subset tailored to their specific needs, whether it's a lightweight model for rapid screening or a high-performance model for detailed diagnostics. Furthermore, by isolating a compact set of the most informative patches, EvoPS offers a path toward more efficient and interpretable models in computational pathology, helping to uncover the key tissue regions that drive diagnostic decisions. As future work, we plan to incorporate additional objective functions that explicitly account for robustness and bias mitigation, enabling EvoPS to produce patch subsets that are not only compact and discriminative, but also resilient to data imbalance and cohort-specific artifacts.

References

- [1] J. van der Laak, G. Litjens, F. Ciompi, The 2030 vision for digital pathology, *Nature Reviews Cancer* 23 (6) (2023) 369–383.
- [2] A. A. Bidgoli, S. Rahnamayan, T. Dehkharghanian, A. Riasatian, H. R. Tizhoosh, Evolutionary computation in action: Hyperdimensional deep embedding spaces of gigapixel pathology images, *IEEE Transactions on Evolutionary Computation* 27 (1) (2022) 52–66.

- [3] W. Bulten, W. Błażej, M. Mo-Aroei, H. Huisman, J. van der Laak, F. Ciompi, G. Litjens, Deep learning for digital pathology image analysis: a comprehensive tutorial, *Medical Image Analysis* 77 (2022) 102387.
- [4] G. Campanella, M. G. Hanna, L. Geneslaw, A. Miraflor, V. Werneck Krauss Silva, K. J. Busam, E. Brogi, V. E. Reuter, D. S. Klimstra, T. J. Fuchs, Clinical-grade computational pathology using weakly supervised deep learning on whole slide images, *Nature Medicine* 25 (8) (2019) 1301–1309.
- [5] E. Chelebian, J. van der Laak, F. Ciompi, Explainable artificial intelligence in computational pathology: A review of the state-of-the-art, *Medical Image Analysis* 91 (2024) 103009.
- [6] X. Wang, P. Chen, W. Wang, S. Zhang, D. Wu, Weakly supervised deep learning for her2 score prediction in breast cancer, *Medical Image Analysis* 89 (2023) 102914.
- [7] D. Yadav, D. Mehra, A survey on patch selection methods for histopathological image classification, *Multimedia Tools and Applications* 82 (12) (2023) 17753–17789.
- [8] G.-L. Li, Z.-W. Liu, K. Li, Y.-H. Chen, S.-J. Wang, A review of saliency detection: From traditional methods to recent deep learning-based methods, *Electronics* 10 (9) (2021) 1053. doi:10.3390/electronics10091053.
- [9] H. Anwar, F. Zeshan, Clustering based patch selection for weakly supervised classification of histopathology images, *Biomedical Signal Processing and Control* 80 (2023) 104332.
- [10] P. Nejat, A. Alsaafin, G. Alabtah, N. I. Comfere, A. R. Mangold, D. H. Murphree, P. Zot, S. Yasir, J. J. Garcia, H. R. Tizhoosh, Creating an atlas of normal tissue for pruning wsi patching through anomaly detection, *Scientific Reports* 14 (2024).
- [11] T. M. Dang, Q. Zhou, Y. Guo, H. Ma, S. Na, T. B. Dang, J. Gao, J. Huang, Abnormality-aware multimodal learning for wsi classification, *Frontiers in Medicine* 12 (2025) 1546452.

- [12] M. Y. Lu, D. F. Williamson, T. Y. Chen, R. J. Chen, M. Barberi, F. Mahmood, Data-efficient and weakly supervised computational pathology on whole-slide images, *Nature Biomedical Engineering* 5 (6) (2021) 555–570.
- [13] R. Grainger, T. Paniagua, X. Yin, N. de Freitas, A. Mahendran, J.-B. Cordonnier, Paca-vit: Learning patch-to-cluster attention in vision transformers, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 15606–15615.
- [14] J.-B. Cordonnier, A. Mahendran, P. H. S. Torr, N. de Freitas, Differentiable patch selection for image recognition, in: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 16496–16505.
- [15] M. Raza, R. Awan, R. M. S. Bashir, T. Qaiser, N. M. Rajpoot, Dual attention model with reinforcement learning for classification of histology whole-slide images, *Computerized Medical Imaging and Graphics* 118 (2024) 102466.
- [16] S. D’Ascoli, A. Vahdat, A. Mallya, J. Kautz, Adaptive patch selection to improve vision transformers through reinforcement learning, in: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024, pp. 2533–2543.
- [17] J. Dong, J. Jiang, K. Jiang, J. Li, Y. Zhang, Fast and accurate gigapixel pathological image classification with hierarchical distillation multi-instance learning, *arXiv preprint arXiv:2502.21130* (2025).
- [18] M. Feng, K. Xu, N. Wu, W. Huang, Y. Bai, C. Wang, H. Wang, Trusted multi-scale classification framework for whole slide image, *arXiv preprint arXiv:2207.05290* (2022).
- [19] Y. Liu, Y. Lin, D. Shen, H. Li, Y. Wang, Q. Wang, X. Liu, S. Xu, Minimal high-resolution patches are sufficient for whole slide image representation via cascaded dual-scale reconstruction, *arXiv preprint arXiv:2508.01641* (2025).
- [20] A. A. Bidgoli, S. Rahnamayan, Multi-phase evolutionary search space shrinking for large-scale multi-objective feature selection, *Applied Soft Computing* (2025) 113755.

- [21] R. Miikkulainen, S. Forrest, A biological perspective on evolutionary computation, *Nature Machine Intelligence* 3 (1) (2021) 9–15.
- [22] N. S. Pun, S. Agarwal, Deep learning and genetic algorithm for covid-19 detection using chest x-ray images, *Multimedia Tools and Applications* 81 (17) (2022) 23589–23616.
- [23] F. L. Azevedo, J. García-Nieto, M. Gómez-Albarrán, F. J. Aguilera-Rueda, A multi-objective genetic algorithm for feature selection and parameter optimization in machine learning, *Expert Systems with Applications* 216 (2023) 119427.
- [24] K. Zeng, D. Nguyen, C.-T. Ly, D. Ruan, Multi-objective radiotherapy treatment planning based on machine learning: A review, *Computers in Biology and Medicine* 153 (2023) 106512.
- [25] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, *IEEE Transactions on Evolutionary Computation* 6 (2) (2002) 182–197.
- [26] K. Miettinen, *Nonlinear Multiobjective Optimization*, Springer, 1999, foundational reference for multi-objective optimization and Pareto concepts. doi:10.1007/978-1-4615-5563-6.
- [27] K. Deb, *Multi-objective optimization using evolutionary algorithms*, Vol. 16, John Wiley & Sons, 2001.
- [28] C. A. C. Coello, D. A. V. Veldhuizen, G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, 2nd Edition, Springer, New York, 2007, comprehensive overview of multi-objective evolutionary algorithms.
- [29] C. G. A. R. Network, et al., The cancer genome atlas pan-cancer analysis project, *Nature genetics* 45 (10) (2013) 1113–1120.
- [30] R. Ingale, S. Nadeem, F. Mahmood, J. Saltz, A. H. Martel, H. R. Tizhoosh, Yottixel - an image search engine for large archives of histopathology whole slide images (2022). arXiv:2209.11821.
- [31] H. Akbari, A. Gostar, S. Daneshvar, M. Lotfi, A. Rezaeian, S. Kalra, Kimianet: a deep learning framework for the classification of renal cell

- carcinoma in histopathology images, *Scientific reports* 11 (1) (2021) 13486.
- [32] G. Huang, Z. Liu, L. Van Der Maaten, K. Q. Weinberger, Densely connected convolutional networks, in: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.
 - [33] P. Shvetsov, M. Borys, J. Van der Laak, G. Litjens, Phikonv2: A foundation model for computational pathology instructed by diverse sources of knowledge, *arXiv preprint arXiv:2404.05834* (2024).
 - [34] P. Shvetsov, M. Borys, J. van der Laak, G. Litjens, Virchow2: A foundation model for pan-tissue computational pathology, *arXiv preprint arXiv:2407.13539* (2024).
 - [35] J. Chen, A. Li, Y. Liu, Z. Chen, Z. Zhang, et al., Uni-2h: A universal model for histopathology image analysis, *arXiv preprint arXiv:2405.09348* (2024).
 - [36] T. M. Cover, P. E. Hart, Nearest neighbor pattern classification, *IEEE Transactions on Information Theory* 13 (1) (1967) 21–27, seminal paper introducing the k-Nearest Neighbor classification method. doi:10.1109/TIT.1967.1053964.
URL <https://doi.org/10.1109/TIT.1967.1053964>