ARTICLES
https://doi.org/10.1038/s41551-022-00929-8
nature biomedical engineering

Check for updates

OPEN

Fast and scalable search of whole-slide images via self-supervised deep learning

Chengkuan Chen[1,2,3,4], Ming Y. Lu[1,2,3,4,5,7], Drew F. K. Williamson[1,2,3,7], Tiffany Y. Chen[1,3,4], Andrew J. Schaumberg[1] and Faisal Mahmood[1,2,3,4,6] ✉

The adoption of digital pathology has enabled the curation of large repositories of gigapixel whole-slide images (WSIs). Computationally identifying WSIs with similar morphologic features within large repositories without requiring supervised training can have significant applications. However, the retrieval speeds of algorithms for searching similar WSIs often scale with the repository size, which limits their clinical and research potential. Here we show that self-supervised deep learning can be leveraged to search for and retrieve WSIs at speeds that are independent of repository size. The algorithm, which we named SISH (for self-supervised image search for histology) and provide as an open-source package, requires only slide-level annotations for training, encodes WSIs into meaningful discrete latent representations and leverages a tree data structure for fast searching followed by an uncertainty-based ranking algorithm for WSI retrieval. We evaluated SISH on multiple tasks (including retrieval tasks based on tissue-patch queries) and on datasets spanning over 22,000 patient cases and 56 disease subtypes. SISH can also be used to aid the diagnosis of rare cancer types for which the number of available WSIs is often insufficient to train supervised deep-learning models.

The increasing availability of technologies allowing for the routine creation of high-resolution whole-slide images (WSIs) has triggered tremendous excitement for the field of digital pathology. Whereas the rich morphologic content analysed by pathologists was once locked in glass slides, whole-slide imaging systems now allow pathologists and researchers to access that data digitally without a microscope at hand. Studies demonstrating non-inferiority of WSIs[1–3] and Food and Drug Administration (FDA) approvals for primary diagnosis to be performed on WSIs mean that pathologists can now adopt these systems for clinical use. However, as institutions scan and store an increasing number of images, they often turn to WSI storage and retrieval paradigms identical to that used for their glass slides—large repositories of data searchable through patient identifiers, case number, date of procedure, pathology report and so on, without leveraging the digital morphologic content of the images themselves.

Meanwhile, the revolution of artificial intelligence[4,5] (for example, deep learning) in recent years has shown potential in various tasks in pathology that range from disease diagnosis, prognosis and integrative multi-omic analysis[6–17]. However, a majority of computational pathology methods are based on supervised deep learning using slide or case level labels to tackle classification or ranking problems. By comparison, an image search tool that harnesses the rich, spatially resolved information in pathology images is much more powerful for a variety of different applications. For example, finding cases with similar morphologic features can assist in diagnosing rare diseases and unusual conditions that may not have enough cases available for accurate supervised classification models to be developed. Other examples include finding cases with similar morphologies to predict outcome for clinical trials with limited samples, identifying similar cases for teaching and parsing large historical repositories in the absence of electronic pathology reports. A critical challenge that hinders large scale, efficient adoption of histology whole-slide image search and retrieval systems is scalability. This is a unique challenge for WSI retrieval systems[18] as compared with other image databases since they need to efficiently search a growing number of slides that can each consist of billions of pixels and be several gigabytes in size.

Due to the computationally prohibitive size of WSIs, most approaches split them into smaller image patches and either focus on patch or region of interest (ROI) retrieval that is tailored to specific applications[19–33]. These implementations often need expert pathologists to exhaustively delineate the ROIs, making the system difficult to scale. Recent work has demonstrated promising patch retrieval results without using manual labels by comparing patches in a continuous embedding space using an encoder pretrained by deep metric learning on a large cohort of natural images[34]. However, this approach is limited to small image patches, required considerable computing resources and had a search speed that scaled with the size of the database. Recent work has also shown that representative image patches, embedded into binarized features using encoders pre-trained on real world images, can be used for WSI level retrieval[35,36]. However, the disadvantage of this method is slow search speed on larger datasets due to $O(n\log(n))$ computational complexity, where n is the number of WSIs in the database. Additionally, the reported performance degrades when the distribution of the numbers of slides is skewed towards a subset of anatomical sites, which is commonly seen in real-world histology datasets. Other recent studies propose improved feature representation for WSIs by creating permutation invariant embeddings[37] or fine-tuning pretrained networks on data with morphological information[38]. Scalibility to large histological datasets that capture

[1]Department of Pathology, Brigham and Women's Hospital, Harvard Medical School, Boston, MA, USA. [2]Department of Pathology, Massachusetts General Hospital, Harvard Medical School, Boston, MA, USA. [3]Cancer Program, Broad Institute of Harvard and MIT, Cambridge, MA, USA. [4]Cancer Data Science Program, Dana-Farber Cancer Institute, Boston, MA, USA. [5]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA. [6]Harvard Data Science Initiative, Harvard University, Cambridge, MA, USA. [7]These authors contributed equally: Ming Y. Lu, Drew F. K. Williamson. ✉e-mail: faisalmahmood@bwh.harvard.edu

real-world imbalances in disease types is crucial for a practical and broadly applicable search engine for histology.

Here we propose self-supervised image search for histology (SISH) as a search pipeline that addresses the issues summarized above. SISH theoretically achieves constant time query speed by representing a WSI as a set of integers and binary codes, and does not require any pixel or ROI level annotations. We evaluate SISH on several tasks: first, performance on disease subtype retrieval from a fixed anatomic site is assessed on three cohorts of data, specifically, the primary diagnostic slides in The Cancer Genome Atlas (TCGA)[39], the Clinical Proteomic Tumour Analysis Consortium (CPTAC)[40] and slides digitized in-house at the Brigham and Women's Hospital (BWH). Second, performance on retrieving slides from the same anatomic site as the query, which we evaluate using the TCGA dataset. In total, we used 22,385 diagnostic whole-slide images across 13 anatomic sites and 56 disease subtypes. Third, we demonstrate the utility of SISH for diagnosis of rare cancer types using in-house and TCGA data. Fourth, we show that SISH can be used for patch-level similar morphologic feature search on a variety of different disease models.

SISH theoretically achieves $O(1)$ constant speed complexity for search, insertion and deletion operations (that is, the speed of the operations does not depend on the size of the database) and supports both slide and patch-level retrieval. SISH leverages a set-based representation of WSI which has better transparency and does not need further supervised training as compared to a continuous vector representation[37]. Specifically, we sample a subset of representative patches (termed a 'mosaic') for each WSI by clustering at low resolution to address the gigapixel size of WSIs. SISH uses a Vector Quantized-Variational AutoEncoder (VQ-VAE)[41] trained on a large dataset in a self-supervised manner and leverages the learned, discrete latent codes to create integer indices for patches in a WSI mosaic. VQ-VAE is a self-supervised approach that learns to generate a small number of descriptive latent codes for each input object. With the integer representation of a slide, we can benefit from the O(loglog(M)) search, insertion and deletion speed for integers whose values are within the range [0, M], provided by the Van Emde Boas tree (vEB tree)[42], where M is a fixed constant in our pipeline (see Methods for additional details). Our approach does not require comparing the regions of the query WSI against regions of every other WSI in the database: we use the vEB tree to first identify a constant number of potential candidates for each patch in the query mosaic and then use a ranking module to identify the most promising patches that are useful for retrieval (see Supplementary Table 17). These patches often contain meaningful ROIs, and they can be visualized by the human user to provide model interpretability, which is desirable in medical applications to enable more transparent and informed decision-making. Finally, we make the source code of SISH open access (https://github.com/mahmoodlab/SISH) for future studies. An overview of the SISH pipeline is shown in Fig. 1, and the detailed search process is illustrated in Fig. 2.

SISH begins by distilling a mosaic representation of a given slide[36] (Fig. 1a). To select the patches used for representing the slide, we use two-stage K-means clustering. Specifically, we first apply K-means clustering on the red, green and blue (RGB) histogram features extracted from patches at ×5 magnification, followed by K-means clustering on the coordinates of patches at ×20 magnification within each initial cluster. We extract image patches corresponding the coordinates of the set of final cluster centres and use them as a mosaic representation of the given slide. To convert the patches into a set of integers and binary codes (Fig. 1b), we train a VQ-VAE, which is a variant of the Variational Autoencoder[43] that gives the input a discrete latent code from a codebook learned on the TCGA slides at ×20 scanner magnification (×200 effective magnification). The codebook generated from the VQ-VAE is held constant throughout all experiments in our study and is not retrained on any of the independent datasets. We use the encoder of the pretrained VQ-VAE along with the learned codebook to encode the patches at ×20 magnification and extract patch features by using a Densenet[44] model and a binarization algorithm. The last step is to convert the discrete latent codes into integers to store the mosaics in the vEB tree. We feed the latent codes of the mosaics into a pipeline composed of a series of average pooling (AvgPools), summation and shift operations. The intuition behind this pipeline is to summarize the information in each scale via summation, then store it into a different range of digits in an integer.

During search (Figs. 1c and 2), we extract the features of the preprocessed mosaic of the query whole-slide image and then apply the proposed guided search algorithm (GSA) to find the most similar results of each query mosaic. The design principle of GSA is to find a fixed number of nearest neighbours using the vEB and only select the neighbours whose Hamming distances from the patches in the query mosaic are below a certain threshold θ_h. The search result of each patch in the mosaic is a list of patches. Each patch contains metadata that document the name of the slide where the patch is located, the diagnosis of the slide and the Hamming distance between the patch in the database and that in the query mosaic. Once each patch in the query mosaic gets its search results, our ranking algorithm ranks the candidate patches used to retrieve the final top-K similar slides. We collect all slides that appear in the search results from the candidate patches and sort them on the basis of Hamming distance in ascending order to return the top-K similar slides. See Methods for additional details.

In the proceeding sections, we demonstrate the performance of SISH for: (1) disease subtype retrieval using public cohorts (TCGA and CPTAC), (2) disease subtype retrieval in an independent cohort (BWH in-house data) to test generalizability, (3) anatomic site retrieval and (4) patch-level retrieval using five different datasets: colon tissue (Kather100k[45]), lung tissue (WSSS4LUAD[46]), a general morphologic atlas of digital pathology (ADP[47]), breast tissue (BCSS[48]) and prostate tissue (BWH in-house data).

Results

Disease subtype retrieval. We first evaluate the performance of SISH on disease subtype retrieval using the TCGA, and we report the majority top-k accuracy (mMV@k, $k=1,3,5$), which assesses how often the majority slide diagnosis in the top-k results matches the ground truth from the query. We use mMV@k as the primary metric for comparison because it is stricter than the commonly used top-k accuracy (see Methods for more details). We built the SISH pipeline on slides from each anatomic site and tested whether SISH can retrieve slides with the correct diagnosis. Overall, SISH outperformed Yottixel by achieving 45.51%, 25.62% and 5.33% higher macro-averaged mMV@1, 3 and 5, respectively, as shown in Fig. 3a–c (see detailed numerical results and individual slide retrieval results in Supplementary Tables 1 and 7). We used macro-averaging because rare cases in an unbalanced real-world histology database are as crucial as the more common ones. The improvement in performance could be partly attributed to the fact that the VQ-VAE is trained on the TCGA albeit in a self-supervised manner without using any external supervisory labels. However, the primary role of the VQ-VAE is to improve the query speed, and we investigated the role of the VQ-VAE in improving the performance (Supplementary Fig. 3). Subsequent sections demonstrate results on several datasets that were not used to train the VQ-VAE. To further investigate the results, we created the confusion matrix and Hamming distance matrix in Supplementary Fig. 1 for each site.

In addition, the speed advantage of SISH becomes pronounced especially after the number of slides in the database exceeds 1,000 (Fig. 3f). The median query speed of SISH remains near-constant despite the growing number of slides, as expected from our theoretical results analysis. We perform more experiments to demonstrate

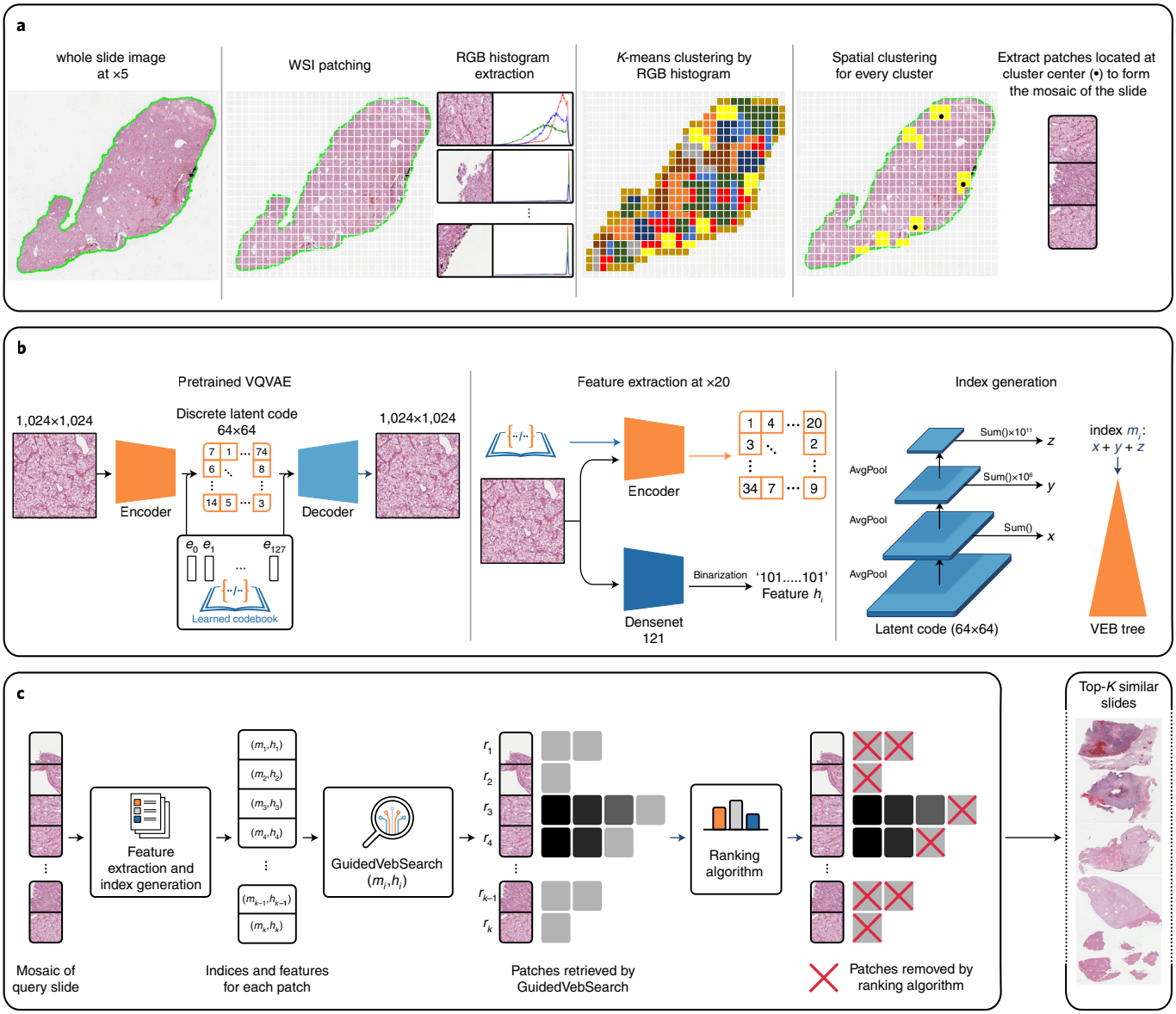


Fig. 1 | Overview of the SISH pipeline. a, After tissue segmentation, we tile the foreground regions and perform two-stage *K*-means clustering to select representative patches to include in the WSI mosaic. We first cluster all patches based on their RGB histogram features. In each cluster generated from the first stage (for example, the yellow cluster shown in the figure), we perform *K*-means clustering again using the spatial coordinates of each patch as features (spatial clustering), extract the patches that correspond to the coordinates of each resulting cluster centre (black dots) and add them to the mosaic of the slide. **b**, We pretrain a VQ-VAE on tissue patches from slides in the TCGA and save its encoder and codebook for feature extraction. For each patch in the mosaic, the VQ-VAE encoder is used to compute its discrete latent representation and a Densenet121 encoder is used to obtain a binarized texture feature vector. Finally, we feed the discrete latent representation into another pipeline composed of a series of average pooling (AvgPool), shift and summation operations to get an integer index for the patch, then use the vEB tree to construct the index structure for search. **c**, For a given query slide preprocessed as a mosaic representation, we feed the mosaic into the feature extractor to compute the integer indices and binarized texture features of each patch in the mosaic, then apply our search and ranking algorithm to filter the candidate patches. See Fig. 2 for more details.

that SISH is scalable to thousands of slides (see Speed and interpretability). Since the ranking algorithm plays a crucial role in the success of SISH, we conduct an ablation study to validate all steps in the ranking module and show that SISH achieved the best performance by including all steps (red line in Fig. 3e).

We also combined kidney renal clear cell carcinoma (KIRC), uterine corpus endometrial carcinoma (UCEC), cutaneous melanoma (SKCM), lung adenocarcinoma (LUAD) and lung squamous cell carcinoma (LUSC) data from CPTAC and TCGA to test performance on a mixed public cohort with the results reported in Fig. 3d. After combining the two datasets, the distribution of the dataset over all sites became more skewed, but the performance of SISH did not vary substantially in most cases. This result further shows that SISH can address, to a degree, dataset imbalance commonly presented in the real world. The only exception was pulmonary-mesothelioma, for which the site where the disease is located was highly unbalanced. Individual retrieval results are available in Supplementary Table 8.

Disease subtype retrieval in independent cohort. WSIs can have large scale domain shift across institutions and medical centers due to variability in tissue preparation, staining and scanning protocols. Therefore, it is essential to validate that the self-supervised

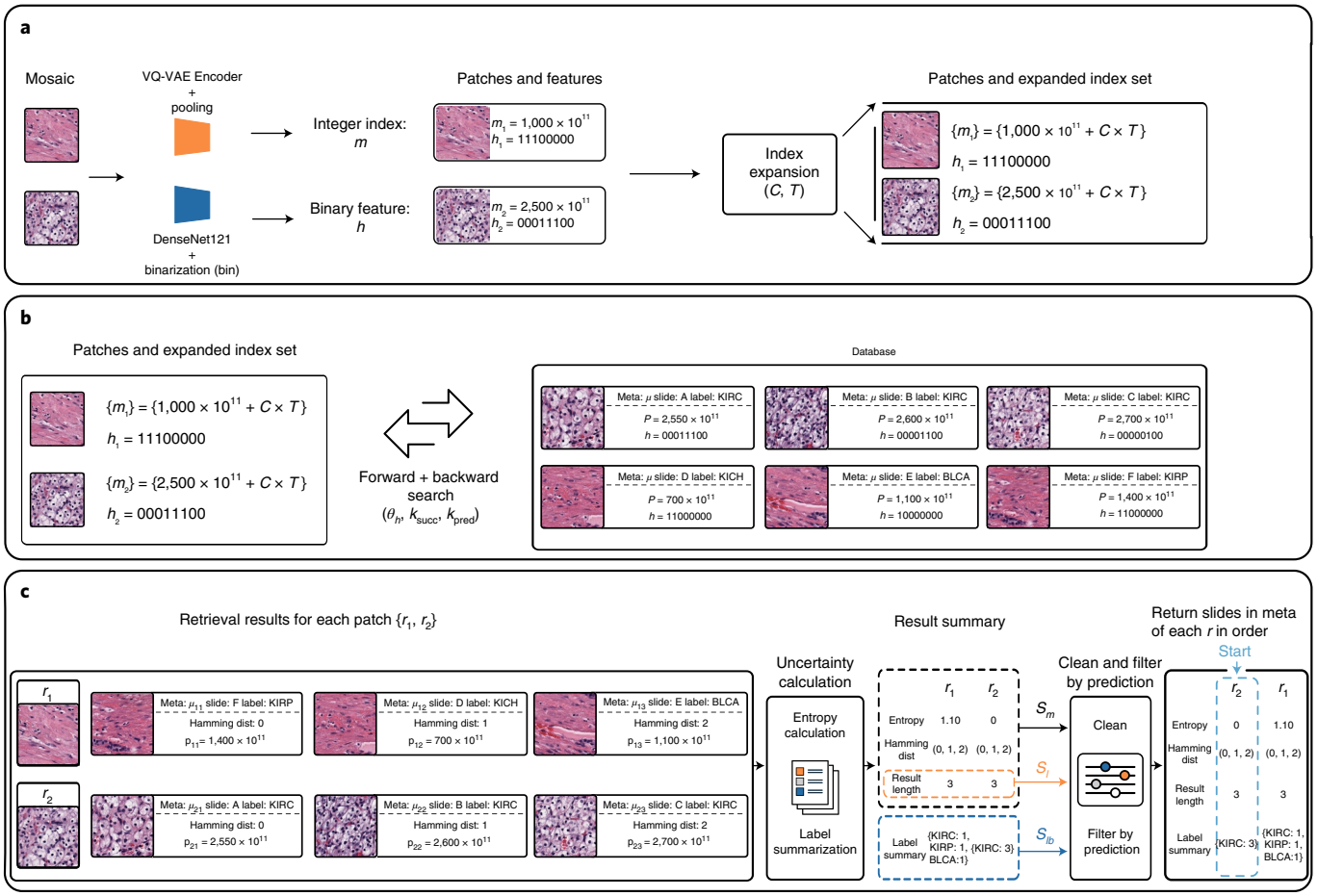


Fig. 2 | Detailed illustration of search. a, Starting from the mosaic of a WSI where a patch could contain normal tissue or morphology of a cancer subtype, SISH encodes each patch into both an integer and binary string representation, using a VQ-VAE encoder and a DenseNet121 encoder pretrained on ImageNet respectively. The pooling operation consists of a series of average pool, summation and multiplication explained in Methods. The binarization process converts a continuous vector to a binary string by starting from ∞, then walking through all elements in the vector to compare the value of the current element to that of the next one. If the next value is smaller than the current, it assigns the current value to 0, and 1 otherwise. Afterwards, for each patch in the mosaic, SISH expands its index into a set of candidate indices. *C* and *T* are hyperparameters used during expansion (see Guided search algorithm section of Methods). **b**, For each patch, SISH applies the search function to each index in the set of candidate indices. The search function returns the patches within k_{succ} successors and k_{pred} predecessors in the database whose Hamming distance from the patch in the query mosaic is smaller than θ_h. Each patch in the database is associated with an index *p* and metadata μ defined in Methods. **c**, For each result *r*, SISH calculates its entropy (by considering the frequency of occurrence of slide labels) and returns summary variables S_m, S_l and S_{lb} for cleaning. In the example shown in the figure, the cleaning function removes outliers in $\{r_1, r_2\}$. A result *r* is considered as an outlier if its length (number of patches retrieved) is greater than the O_h or smaller than the O_l percentile. At the same time, the function also removes patches within each *r* whose Hamming distance is greater than the average of the top-*k* in *r*. Lastly, SISH takes a majority vote of the top-5 slide labels within each *r* to remove patches whose slide labels disagree with the majority vote and extract the slides from the *r* with the lowest entropy (see the corresponding sections in Methods for details of entropy calculation, clean and filter-by-prediction).

VQ-VAE trained on TCGA is robust and adaptable to a different out of domain dataset. From the WSI database at Brigham and Women's Hospital, we collected 8,035 diagnostic slides that span 9 anatomic sites with 37 primary cancer subtypes. For each anatomic site, we built our pipeline separately and reported the mMV@1,3 and 5 scores along with mean average precision at 5 (mAP@5) as a ranking-aware metric (see Evaluation in Methods). SISH performed better than Yottixel by achieving 7.87%, 5.33% and 5.33% higher macro-averaged mMV@1,3 and 5 scores, respectively, as reported in Fig. 4a–c (see numerical results and retrieval results of each slide in Supplementary Tables 2 and 9). Furthermore, SISH outperformed Yottixel in mAP@5 in 34 out of 37 subtypes, leading to a 9.5% higher macro-average mAP@5 (Fig. 4d). We also report the confusion and Hamming distance matrix for each anatomic site (Supplementary Fig. 2). Note that we did not use fine-tune or any other form of domain adaptation to refine our self-supervised encoder in this cohort, showing the generalizability of our encoder trained only on TCGA. While this adaptability mimics a common scenario where the self-supervised encoder is generically trained and the database is built for individual cohorts, we also investigated cross encoder and cross database retrieval by keeping both the self-supervised encoder and database developed on the TCGA constant and querying using cases from CPTAC and BWH data for the same set of diagnosis (see Supplementary Fig. 7). We found that while there is a drop in performance, our approach was generally adaptable.

Rare disease subtype retrieval. The number of archival slides for rare diseases is usually fewer than that of common ones, making it challenging to train an efficient supervised classifier using modern machine learning methods. To further investigate the clinical

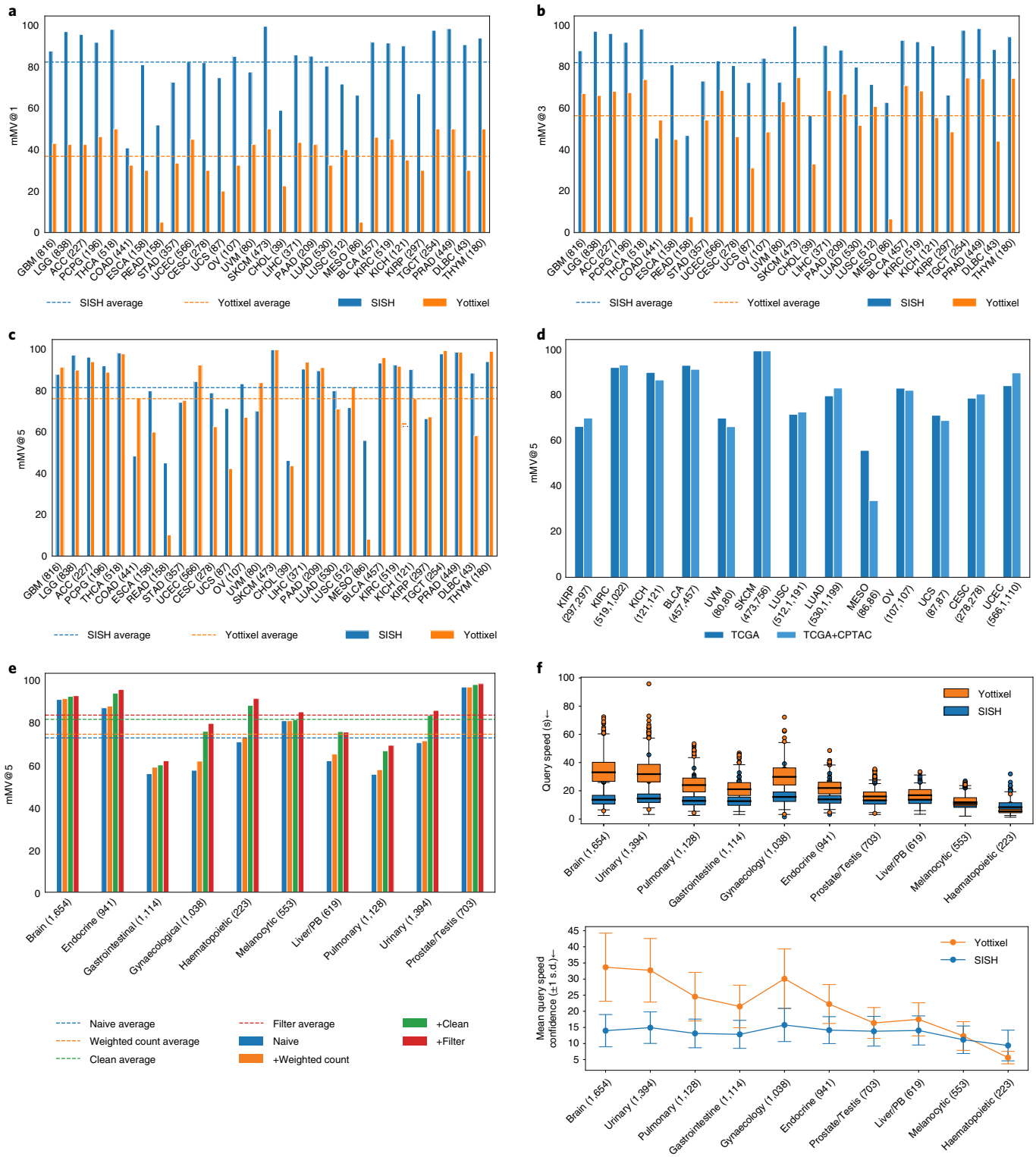


Fig. 3 | Disease subtype retrieval in public cohorts. a–c, Macro-average mMV@1,3 and 5 of SISH and Yottixel on the TCGA anatomic sites. SISH has better performance in all metrics, especially mMV@1 and mMV@3. **d**, Comparison between SISH on TCGA and TCGA+CPTAC cohorts. The performance does not vary before and after mixing with CPTAC cohorts for most cases. **e**, Ablation study result for the ranking module of SISH. We observed that SISH achieves best performance in the setting where all functions are applied (+filter) (details of each setting are described in Methods (Ablation study). **f**, Top: query speed comparison between SISH and Yottixel for each site. The box extends from the first quartile (Q1) to the third quartile (Q3) of the data and the whiskers extend from the box by 1.5× the interquartile range (IQR). Bottom: the mean confidence (±1 s.d.) of query speed between SISH and Yottixel. It is crucial to note that SISH is 2× more effective when the number of slides is over 1,000 (details on the study of speed is reported in Speed and interpretability. Numbers in parentheses denote the number of WSIs for all panels except **d**, where numbers in parentheses denote the number of WSIs in TCGA and TCGA+CPTAC, respectively.

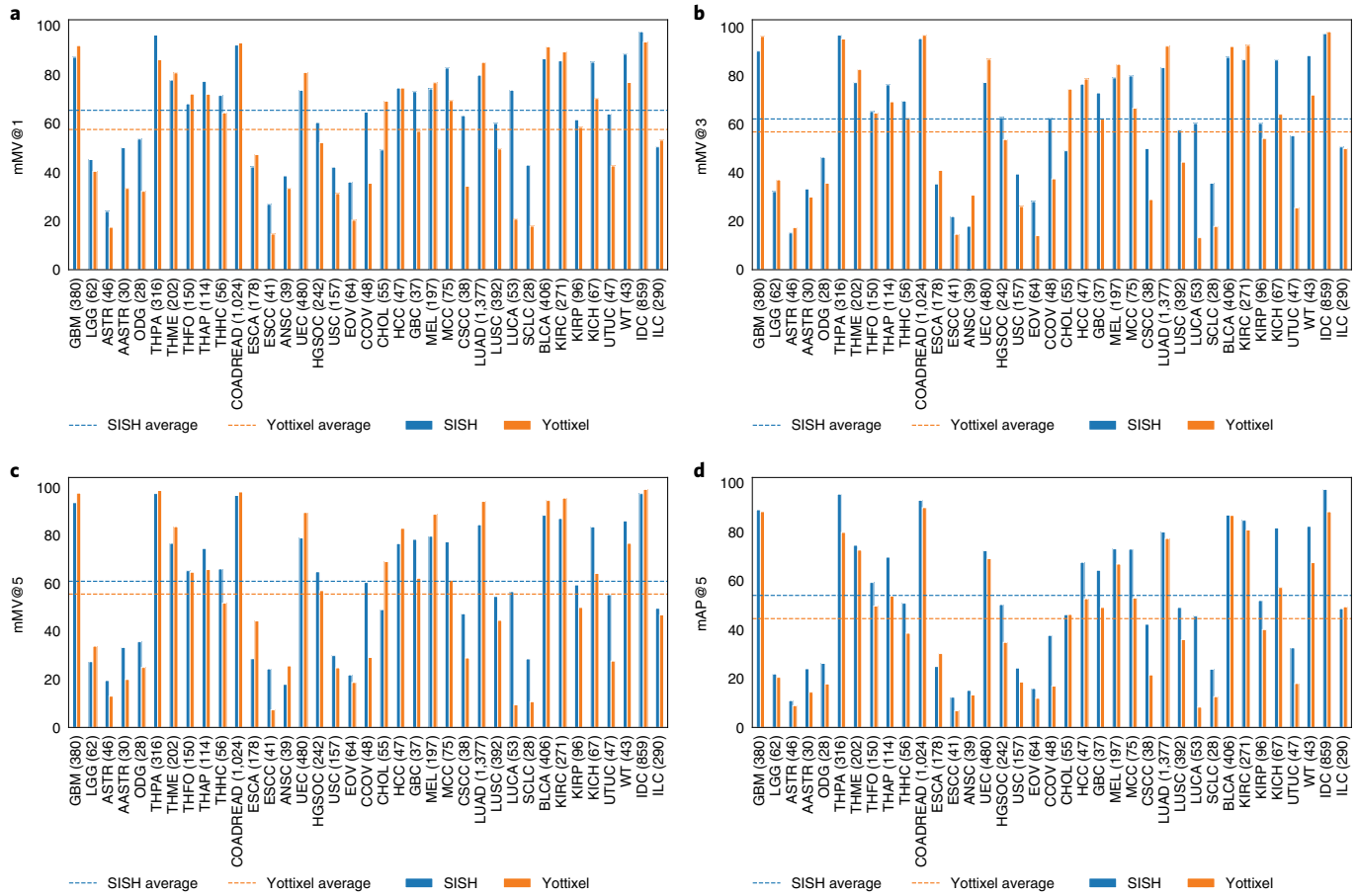


Fig. 4 | Adapting to the BWH independent test cohort. a–c, Average mMV@1, 3 and 5 scores of SISH and Yottixel for each subtype in BWH general cohorts. SISH achieved higher scores than Yottixel by 7.87%, 5.33% and 5.33% for mMV@1, 3 and 5, respectively. **d**, mAP@5 score of SISH and Yottixel. SISH outperformed Yottixel in 34 out of 37 subtypes, resulting in 9.5% higher mAP@5 score. Numbers in parentheses denote the number of WSIs.

value of SISH in assisting with the diagnosis of rare diseases, we conducted an experiment specifically on rare cancer types by combining our BWH cohort and TCGA, resulting in 1,785 slides for 23 rare-type cancers from 7 anatomic sites. The performance of SISH was on par with that of Yottixel for the mMV@5 metric, but achieved slightly better macro-averaged mMV@1 and 3 scores that were 4.56% and 2.42% higher, respectively, as shown in Fig. 5a–c (see numeric results and retrieval results of each slide in Supplementary Tables 3 and 10). Although the performance gap of mMV@1, 3 and 5 scores between SISH and Yottixel was small, SISH achieved better mAP@5 in 22 out of 23 subtypes, which resulted in an overall 9.82% performance improvement (Fig. 5d). We additionally investigate the performance of SISH against a supervised classifier trained using limited WSI data for a subset of rare diseases and found that supervised classifiers can have a high variance across cross-validation folds. SISH also performed better than the average across all folds (Supplementary Table 6).

Anatomic site retrieval. Although the anatomic site from which tissue is resected is always known, historical archival repositories that may not have corresponding digitised pathology reports and electronic medical records will significantly benefit from automated anatomic site identification. We used the diagnostic slides from TCGA and followed ref. [36] to group slides into 13 categories, resulting in 11,561 WSIs. We built our SISH pipeline on this database with the goal to retrieve slides with the same anatomic site as the query. SISH achieved 68.52% mMV@10 on average, which is comparable to Yottixel's performance (67.42%) (Fig. 6a). We used mMV@10 for comparison in this experiment as this was the best performance reported in the previous study[36]. However, we note that SISH is over 15× faster than Yottixel as shown in Fig. 6b, although the classification performance gap between the two methods is small. A detailed comparison of speed between SISH and Yottixel can be found in Speed and interpretability, and individual retrieval results are available in Supplementary Table 11.

Analysis of speed and interpretability. We show how SISH allows the user to interpret the results of a query slide in Extended Data Fig. 1. For a query slide, SISH returns the regions in the slide that are useful for defining the similarity of the slides. This property allows us to examine these regions and ensure that the search system returns results based on reasonable morphologic evidence as agreed by pathologists instead of meaningless regions such as debris. More examples are shown in Extended Data Figs. 2–4. We conducted three interpretation studies using KIRC, ovarian serous cystadenocarcinoma (OV) and stomach adenocarcinoma (STAD) in TCGA to understand SISH's interpretability across different levels of performance (in terms of differences in mMV@5 scores). For each study, we randomly selected 30 queries that contained at least 1 correct retrieval and then extracted the ROIs found in the query slide. We asked a board certified pathologist to rate whether the ROIs agree with their judgements by 'agree', 'partially agree' (that is, if the pathologist agrees with at least one of the ROIs) and 'disagree'. The key finding was that the ratio of 'agree' plus 'partially agree' exceeded 70% in all studies (Supplementary Fig. 6).

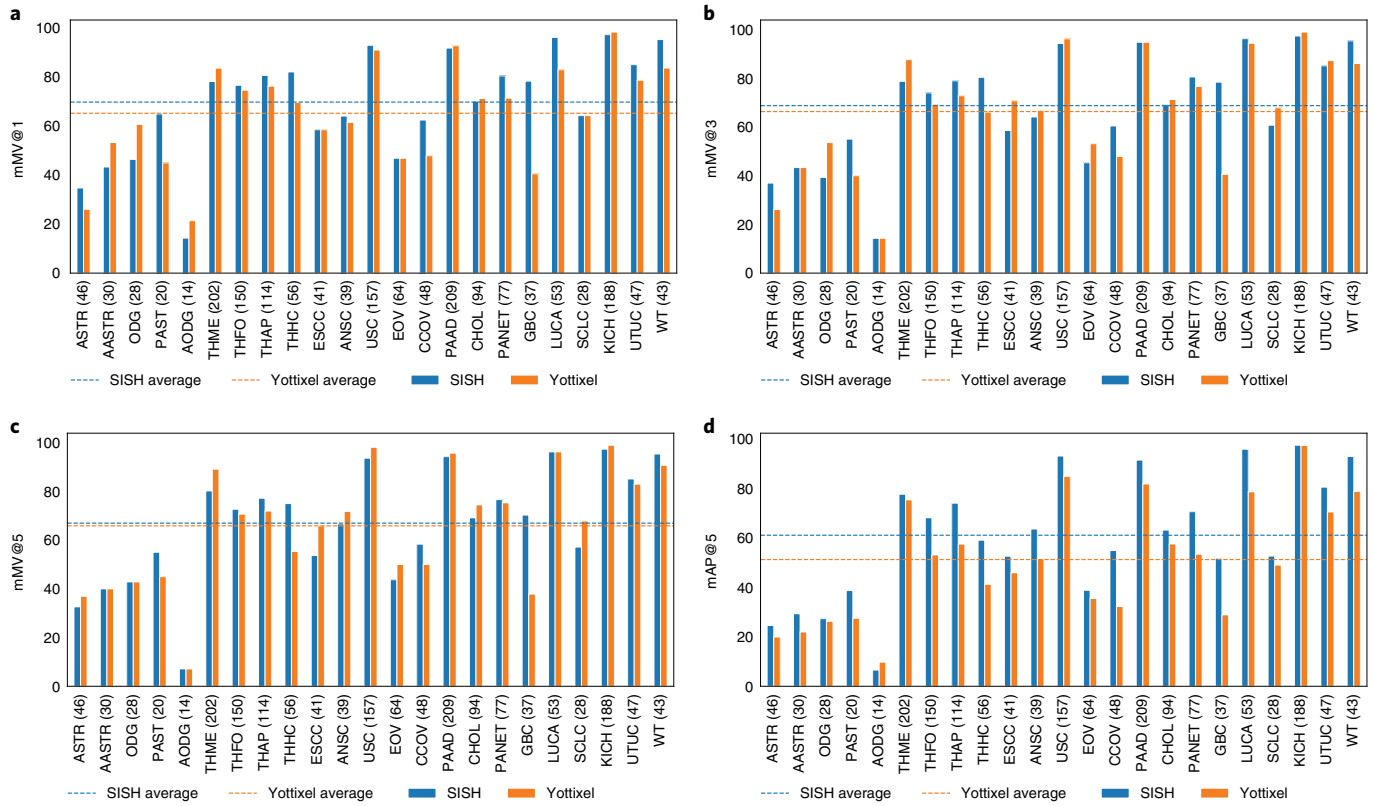


Fig. 5 | Adapting to rare cancer types. a–**c**, Macro-average mMV@1, 3 and 5 scores of SISH and Yottixel in each rare cancer subtype. SISH achieved higher mMV1 and 3 scores than Yottixel by 4.56% and 2.42%, respectively. **d**, Macro-average mAP@5 score. SISH outperformed Yottixel in 22 out of 23 subtypes by 9.82%. Numbers in parentheses denote the number of WSIs.

We used the TCGA dataset from the anatomic site retrieval experiment to evaluate query speed. We applied weighted sampling to select slides from each site to create databases of size 500, 1,000, 2,000, 3,000, 4,000, 5,000, 7,000 and 9,000, and the overall dataset with 11,561 slides. We implemented both methods in Python and evaluated them on the same machine for fair comparison. The average query speeds of both methods are reported in Fig. 6b. Since we observed that Yottixel was inefficient beyond 3,000 slides, we used the same 100 queries sampled from the databases to calculate the average query speed of SISH and Yottixel instead of using all data when the size of the database exceeded 3,000. By comparison, the average query speed of SISH remained almost constant, with low variances across the range of database sizes, consistent with our theoretical results. This result demonstrates that SISH can scale with the growing number of slides while maintaining a relatively constant query speed.

Patch-level retrieval. For patch-level retrieval, we viewed each patch query as a single mosaic fed into the SISH search pipeline. Since there is only one patch in the mosaic, there is no need for the ranking module. We obtained the top-K results by directly sorting the predictions by their Hamming distances. We collected patch data across 4 anatomic sites (lung, breast, colon and prostate) from 6 cohorts: TCGA, BWH, Guangdong Provincial People's Hospital (GDPH), Atlas, National Center for Tumor Diseases (NCT) biobank, and University Medical Center Mannheim (UMM). We evaluated SISH on 10 tissue types and built database sizes from 4,000 to 13.4 million from the collected patch data. More details can be found in the Methods. We only compared SISH to Yottixel as neither was specifically designed for patch retrieval, unlike other published methods.

We report the mMV@5 patch retrieval performance of Kather100k[45], WSSS4LUAD[46], BWH prostate, Atlas[47] and BCSS[48] in Fig. 7a–e. The performance of SISH is on par with that of Yottixel in all experiments, SISH's mean query speed was faster than Yottixel's by three times when the size of the database reached 100,000, as shown in Fig. 7f (0.18 ± 0.06 s versus 0.53 ± 0.04 s). To further test the query speed on larger databases, we created two large databases by mixing Kather100k with colon adenocarcinoma and rectal adenocarcinoma in TCGA (TCGA-Kather) and BCSS with Breast in TCGA (TCGA-BCSS). Both TCGA cohorts were patched at 256×256. The size of the merged databases reached 5.5 million and 13.4 million images, respectively. The mean query speed of SISH reached 120 times faster than Yottixel on TCGA-Kather and 230 times faster on TCGA-BCSS (0.27 ± 0.036 s versus 32.44 ± 1.02 s and 0.32 ± 0.016 s versus 74.55 ± 2.00 s), as shown in Fig. 7f. On databases of size over 10 million, we observed a performance improvement in speed over previous approaches[34]. We report more comparisons with different metrics and show examples in Extended Data Figs. 5–10 (see individual retrieval results in Supplementary Tables 12–16).

Discussion

In summary, we show that SISH addresses several of the key challenges in whole-slide image search: speed, accuracy and scalability. Our experiments demonstrate that SISH is an interpretable histology image search pipeline that achieves constant search speed after training with only slide-level labels. We also demonstrate that SISH has strong performance on large and diverse datasets, can generalize to independent cohorts as well as rare diseases and, finally, that it can be used as a search engine not just for WSIs, but also for image patch retrieval.

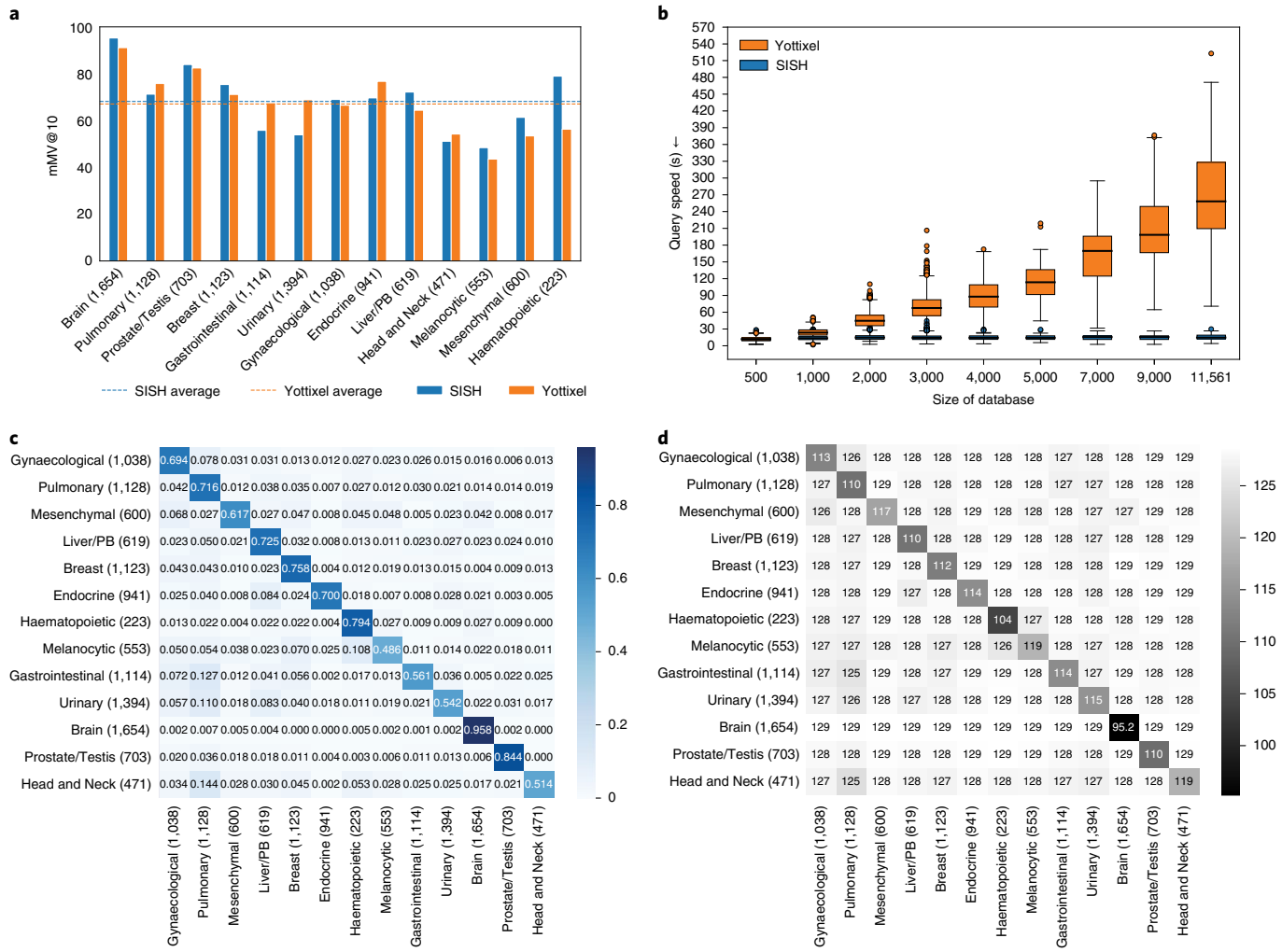


Fig. 6 | Performance on anatomic site retrieval and speed. a, mMV@10 comparison between SISH and Yottixel. **b**, Speed comparison between SISH and Yottixel in the TCGA anatomic site retrieval. SISH is faster than Yottixel by approximately 15× when the number of slides is over 10,000. See Speed and interpretability for more details. The box extends from Q1 to Q3 of the data and the whiskers extend from the box by 1.5 × IQR. **c**,**d**, Confusion matrix (**c**) and Hamming distance matrix (**d**) of SISH for anatomic site retrieval. The *x* and *y* axes correspond to the model prediction and ground truth, respectively. The sharp diagonal line in both matrices show that SISH can retrieve the correct results and avoid the dissimilar ones in most cases. Numbers in parentheses denote the number of WSIs from each site. PB, pancreaticobiliary.

Search functions have transformed and enabled our modern digital lives, and digitization of pathology reports (as well as medical records themselves) represent a similar transformation for pathology practices. Searching databases of digitized pathology report texts has proved to be extremely useful in clinical practice, from seeing how colleagues have signed out similar cases to identifying cases for research or quality control initiatives. However, pathology reports capture only a tiny portion of the information contained in the associated slides, and that information can be quite limited. Depending on the pathology practice, there may be little description of the tissue on a set of slides beyond only what was necessary to make a particular diagnosis or was required for structured/synoptic reporting. Additionally, what descriptions are present using the language of pathology (for example, 'myxoid', 'salt-and-pepper chromatin' or 'micropapillary') are not localized to a particular slide or region of interest.

By comparison, a search function that harnesses the rich, spatially resolved information in pathology images is much more powerful for certain applications. Use-cases of such a system include: (1) pathology trainees finding cases with similar morphologies to learn how their mentors would diagnose the case of interest; (2) disease subtyping to provide more evidence to a pathologist for a particular diagnosis; (3) researchers identifying tumours that share certain features for clinical or genomic correlations; (4) assistance in diagnosis of rare morphological findings; (5) quality control functions to identify potential sample swaps, patient misidentification, or outlier detection; and (6) primary site suggestions for metastases (7) disentangling large historical repositories in the absence of electronic medical and pathology reports.

Additionally, the importance of performant image search systems will probably increase in the near future, as the penetrance of slide scanning solutions in pathology practices grows. As institutions' WSI repositories grow to hundreds of thousands or millions of slides, only systems with constant or near-constant search speed and the ability to operate without pixel-level annotations will reasonably be able to scale and will be deployed for use in clinical practice. Interpretability of such a system will allow quality control of the search itself to be baked in, giving users the ability to troubleshoot searches on the fly. A caveat, however, to the interpretability of SISH is that it only returns relevant ROIs and as with many deep learning systems the feature representation itself largely remains abstract[49].

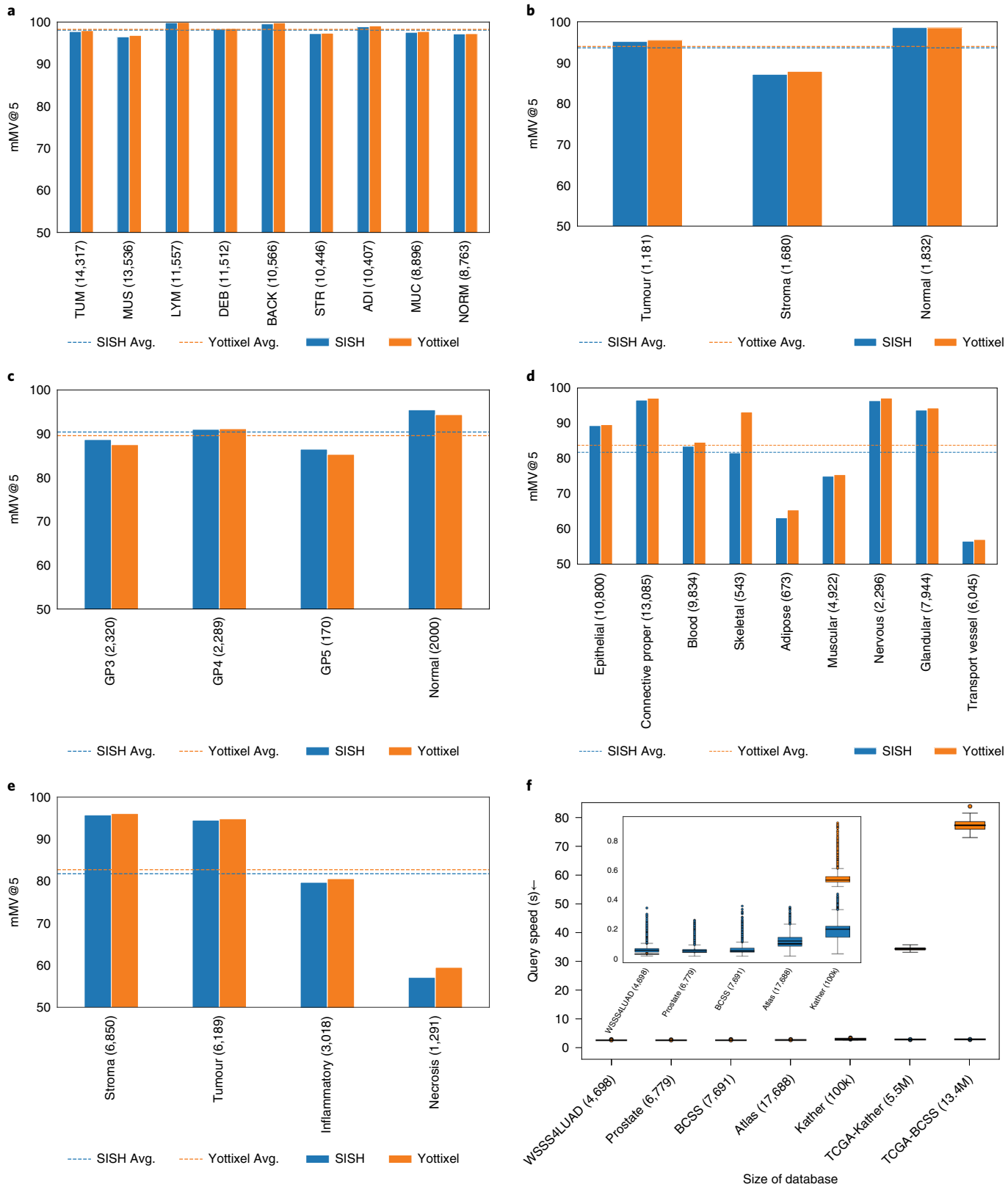


Fig. 7 | Patch-level retrieval. a–e, mMV@5 results for SISH and Yottixel for patch level retrieval on multiple patch level datasets: Kather100k, WSSS4LUAD, BWH prostate, Atlas and BCSS datasets. SISH performed similarly to Yottixel on all datasets. **f**, Query speeds of SISH and Yottixel. SISH achieved faster mean query speed by a factor of 3 to 230 as the size of the database grew from 100,000 to 13.4 million images. The results were averaged across query times of all data in the database, except for the TCGA-Kather and TCGA-BCSS databases because of their large size. For these latter two databases, the results were averaged across the query time of all data in Kather100k and BCSS for SISH and 100 random samples from Kather100k and BCSS for Yottixel due to slow performance. In summary, SISH has similar performance to Yottixel but faster search speed when the size of the database grows. Inset: a closer look at the first five patch dataset. The box extends from Q1 to Q3 of the data and the whiskers extend from the box by 1.5 × IQR.

Our approach has several other limitations that future studies may address. The discrete index used to look up similar candidate patches may have limited expressive power, especially if a query mosaic is large, as this may require visiting many neighbors in the vEB tree to cover all promising candidates. This can limit the efficiency of the search engine despite a theoretical constant search time. Additionally, by using a fixed set of patches to represent each WSI in the form of a mosaic, we could potentially miss other informative regions in the slide that are valuable for search but do not make it into the final mosaic of the WSI. Also, during retrieval, candidates for each patch in the query mosaic are mutually exclusive of one another; since all patches are extracted at a single fixed resolution, morphological patterns that span larger regions of interest may not be adequately represented during search. Other limitations include the large space (memory) complexity of the vEB tree; however, we found the memory utility to be reasonable and tractable on a consumer grade workstation for all experiments conducted in this study. Given the scale of the study, several hyper-parameters were not tuned and conventional values were used from prior literature; further tuning these hyper-parameters may lead to better results.

Finally, the current system has been developed only to search for images using a query image. In clinical practice, pathologists rely on other data such as the patient's medical record, other imaging modalities and molecular test results to guide diagnoses and clinical decision making. Therefore, we believe one important future direction is to develop a multimodal version of SISH by pairing each WSI with other data from the same patient so that our search system can present a holistic view for pathologists, given a query WSI. In a similar vein, extending SISH to accept multimodal queries, such as text or genomic data, would be a promising direction, provided an efficient way to compare and find semantic similarities between disparate and orthogonal data types can be developed. Other advances could include the development of similar fast and scalable search engines for multiplex immunofluorescence and spatial transcriptomics data. Overall, SISH represents an initial proof-of-concept for the utility of self-supervised learning for retrieval in massively large medical datasets and paves the way for future studies to explore the utility of larger datasets at scale, and additional modalities to eventually build a universal search engine for biomedicine.

Methods

SISH. SISH is a histology-image search pipeline that addresses the scalability issues of speed, storage and pixel-wise label scarcity. While image-level class labels or annotation for whether a pair or triplet of images are similar/dissimilar has often been leveraged to explicitly learn an embedding space that captures semantic similarity between images, or to identify key points before retrieval[50,51], it is difficult to directly apply such techniques to WSI image search due to the high dimensionality of large WSIs and the lack of patch-level annotation. Instead, SISH builds upon a set of preprocessed mosaics from WSIs without pixel-wise or ROI-level labels to reduce storage and labelling costs, by relying on indices learned via self-supervised learning and pretrained embeddings. SISH scales with a constant search speed, regardless of the size of the database, by taking advantage of the benefits of discrete latent codes from a VQ-VAE, and using guided search and ranking algorithms. We present these essential components of SISH in this section and provide a pictorial illustration for the methods described in Fig. 2. For clarity, we have summarized all symbols used in the following text in Supplementary Table 18.

Discrete latent code of VQ-VAE. VQ-VAE[41] is a variant of a Variational AutoEncoder (VAE) that introduces a training objective that allows for discrete latent codes. Let $\boldsymbol{e} \in \mathbb{R}^{K \times D}$ be the latent space (that is, codebook) where K is the number of discrete codewords and D is the dimension of the codewords. We set $K=128$ and $D=256$ in our experiments. To decide the codeword of the given input, an encoder q encodes input x as $z_e(x)$. The final codeword $z_q(x)$ of x and the training objective function are given by

$$\boldsymbol{z}_q(x) = \boldsymbol{e}_k, \text{where } k = \text{argmin}_j \left\| \boldsymbol{z}_e(\boldsymbol{x}) - \boldsymbol{e}_j \right\|, \tag{1}$$

$$\log p(\boldsymbol{x}|\boldsymbol{z}_q(\boldsymbol{x})) + \left\| \text{sg}[\boldsymbol{z}_e(\boldsymbol{x})] - \boldsymbol{e} \right\| + \alpha \left\| \boldsymbol{z}_e(\boldsymbol{x}) - \text{sg}[\boldsymbol{e}] \right\|, \tag{2}$$

where α is a hyperparameter and sg denotes the stop gradient operation. A stop gradient operation acts as the identity function during the forward pass while having zero gradient during the backward pass. The first term in the objective function optimizes the reconstruction of the encoder and decoder, the second term is used to update the codebook, and the third term is used to prevent the encoder's output from diverging too far from the latent space. The architecture of our VQ-VAE model is shown in detail in Supplementary Fig. 4. We re-ordered the codebook on the basis of the value of the first principal component and changed the latent code accordingly as we found that the re-ordered codebook can provide a more semantic representation of the original input image (see Supplementary Fig. 5).

Feature extraction, index generation and index encoding. We show how each patch i in the mosaic of a WSI can be represented by a tuple (p_i, h_i) composed of a patch index p_i and a patch texture feature h_i. To get p_i, we encode and re-map the latent code z_i from the encoder and re-ordered codebook from the VQ-VAE. The index p_i is determined by the following equations:

$$\boldsymbol{z}_{i,1} = \text{AVGPOOL}(2,2)(\boldsymbol{z}_i) \tag{3}$$

$$\boldsymbol{z}_{i,2} = \text{AVGPOOL}(2,2)(\boldsymbol{z}_{i,1}) \tag{4}$$

$$\boldsymbol{z}_{i,3} = \text{AVGPOOL}(2,2)(\boldsymbol{z}_{i,2}) \tag{5}$$

$$p_i = \text{SUM}(\boldsymbol{z}_{i,1}) + \text{SUM}(\boldsymbol{z}_{i,2}) \times 10^6 + \text{SUM}(\boldsymbol{z}_{i,3}) \times 10^{11} \tag{6}$$

$$\text{SUM}(\boldsymbol{z}_{i,1}) \in [0, 130048], \text{SUM}(\boldsymbol{z}_{i,2}) \in [0, 32512], \text{SUM}(\boldsymbol{z}_{i,3}) \in [0, 8128] \tag{7}$$

To convert the information in the latent code from higher to lower resolution, we apply a series of 2×2 average pooling. We then take the summation to aggregate the latent code in each resolution, as the summation operator has better expressive power than the mean or maximum[52]. We get the final integer index by taking the summation of the information aggregated in each resolution and multiplying it by 10^0, 10^6 and 10^{11}, respectively. The intuition behind choosing the power is to keep the information of the latent code in each resolution (that is, $z_{i,1}$, $z_{i,2}$ and $z_{i,3}$) separate. For example, multiplying $\text{SUM}(z_{i,2})$ by 10^6 separates the feature in the second layer from $\text{SUM}(z_{i,1})$ since the maximum of the latter is 130,048. Likewise, multiplying $\text{SUM}(z_{i,3})$ by 10^{11} separates the feature in the last layer from the previous two. We insert each p_i into the vEB tree for fast search. We apply this process to WSIs in the datasets to build our databases. Note that the time complexity of all operations in the vEB tree is $O(\log\log(M))$. On the basis of the properties of the vEB tree, M can be determined by

$$M = 2^x > \max(p_i), \tag{8}$$

where x is the minimum integer that satisfies the inequality. Since our codebook size ranges from 0 to 127, we can determine the maximum summation $\text{Sum}(z)$ in each level. Solving the inequality, we find that the minimum M that satisfies the inequality is $M=1{,}125{,}899{,}906{,}842{,}624$. Because M is a constant that only depends on the index generation pipeline, our search performance is $O(1)$. One limitation of using vEB is that it has a large space complexity $O(M)$ where M depends on the size of the codebook and the dynamic range of the index used for search. M remains fixed and does not scale with the number of data points (WSIs or patches) in the database. To get h_i, we use DenseNet121 to extract features from the $1{,}024\times1{,}024$ patch at ×20, then follow the algorithm proposed in ref. [36] to binarize it (that is, starting from ∞; if the next value is smaller than the current one, the current value 0 is assigned, and 1 is assigned otherwise).

In addition to creating the tuple to represent the patch, we also make a hash table H with p_i as key and the metadata μ of the patch as value. The metadata include the texture feature h_i, the name of the slide associated with the patch, the coordinates on the slide from which the patch is cropped, the file format of the slide and the diagnosis of the slide. Note that different patches could share the same key. In this case, the value is a list that stores the metadata for each patch. If the size of the search database is significantly large, which is expected to be the case for most practical real-world databases, the search speed would be greater than pre- and post-processing steps. When running a fixed number of iterations, the K-means clustering algorithm (Lloyd's algorithm) has time complexity $O(BKIC)$ where B is the number of patches in a WSI, K is the number of cluster centroids, I is the number of iterations and C is the dimension of each input data point. For fixed I, K and C, the initial clustering steps of mosaic construction is $O(B)$. To obtain the final mosaic, a fixed percentage (e.g. 5%) of patches are sampled from each cluster, and hence the resulting mosaic varies from slide to slide with size $B' = 0.05 \times B$. During retrieval, the number of total candidates proposed (before ranking) is $T \cdot (k_{\text{succ}} + k_{\text{pred}}) \cdot B'$ (see the next section for the definition of T, k_{succ} and k_{pred}). For ranking, the complexity is $O(B')$. Therefore, given fixed k_{succ}, k_{pred} and T, the time complexity of retrieval is $O(B')$. Note that since the size of a WSI is capped by the physical size of the glass slide and the tissue specimen, for a fixed patch size, we can always pick a reasonable constant B_{max} to upper bound the maximum B in the database and in incoming query slides. Therefore, the entire workflow

has a theoretical constant time complexity of $O(1)$. In real-world scenarios where we expect the size of the database to scale to hundreds of thousands or millions of slides, the time complexity of retrieval will dominate over other steps such as mosaic generation and ranking if we do not use an $O(1)$ search algorithm and it instead scales with $O(n)$ or $O(n\log n)$, where n is the size of the database. However, we note that while in most practical scenarios with increasingly large databases, the size of the WSI database (n) would be larger than the size of the number of patches in the query slide (B); in rare cases where the size of the database is very small, such that average B is not negligible compared to n, while the search operation will continue to have a constant $O(1)$ complexity, the speed of the overall pipeline may be limited by the mosaic generation $O(B_{max})$. Mosaic generation can also be completed before case review, further improving search speeds.

Guided search algorithm. For clarity, we use m_i to denote the patch index in the mosaic of the query slide to distinguish those in the database. Given a query slide I represented as $I=\{(m_1,h_1),(m_2,h_2),\ldots,(m_k,h_k)\}$ with k patches, where each tuple is composed of the index of the patch m_i and its texture features h_i, we apply GUIDED-SEARCH to each tuple and return the corresponding results r_i. The output takes the form of $R_I=\{r_1,r_2,\ldots,r_k\}$. Each $r_i=\{(p_{i1},\mu_{i1}),(p_{i2},\mu_{i2}),\ldots,(p_{in},\mu_{in})\}$, a set of tuples consisting of the indices of similar patches $(p_{i1},p_{i2},\ldots,p_{in})$ and their associated metadata $(\mu_{i1},\mu_{i2},\ldots,\mu_{in})$. μ_{ij} includes all metadata associated with the j-th patch plus the Hamming distance between h_i and h_j. A visual illustration is shown in Fig. 2.

The drawback to using only m_i for the query is that the current patch index is sensitive to minor changes in $z_{i,3}$. For example, a patch that differs from another by 1 incurs a 10^{11} difference in index, putting the two patches far from each other in the vEB tree. To address this issue, we create a set of candidate indices $m_{i,c+}$ and $m_{i,c-}$ along with the original m_i by adding and subtracting an integer C for T times from Sum($z_{i,3}$). We call helper functions FORWARD-SEARCH and BACKWARD-SEARCH to search the neighbour indices in $m_{i,c+}$ and $m_{i,c-}$, respectively. Both functions include only those neighbouring indices whose Hamming distance from the query h_i is smaller than a threshold, θ_h. The details of these algorithms are presented in Algorithms 1 through 3.

Algorithm 1 Guided Search Algorithm

$H \leftarrow$ *hash table* ▷ Hash table with patch index as key and metadata as value
$C, T \leftarrow 50\times10^{11}, 10$ ▷ Integer and number of times for addition and subtraction
$\theta_h \leftarrow 128$ ▷ Threshold of the Hamming distance between query patch index and the neighbor
$k_{succ}, k_{pred} \leftarrow 375$ ▷ Number of time to call *vEB.Successor()* *and vEB.Predecessor()*
Function GUIDED-SEARCH($m_i, h_i, C, T, \theta_h, k_{pred}, k_{succ}, H, vEB$)
  $m_{i,c+}, m_{i,c-}, results \leftarrow \{\},\{\},\{\}$
  $V \leftarrow \{\}$
  $m_{i,c+}.insert(m_i)$
  for $t \leftarrow 1,2,\ldots,T$ **do**
    $m_{tmp+}, m_{tmp-} \leftarrow m_i + t\times C, m_i - t\times C$
    $m_{i,c+}.insert(m_{tmp+})$
    $m_{i,c-}.insert(m_{tmp-})$
  $results_+, V \leftarrow$ FORWARD-SEARCH($m_{i,c+}, k_{succ}, \theta_h, V, H, vEB$)
  $results_- \leftarrow$ BACKWARD-SEARCH($m_{i,c-}, k_{pred}, \theta_h, V, H, vEB$)
  $results.insert(results_+)$
  $results.insert(results_-)$
  $results \leftarrow$ SORT-ASCENDING($results, key = results.hammingdistance$)
  return *results*

Results ranking algorithm. Our ranking function RANKING (Algorithm 4) takes the results $R_I=\{r_1,r_2,\ldots,r_k\}$ from GUIDED-SEARCH as input. The output is the top-k similar slides given the query slide I. We set k equal to 5 for all experiments, except for anatomic site retrieval where k is equal to 10. The intuition behind RANKING is to find the most promising patches in R_I on the basis of the uncertainty. It relies on three helper functions—WEIGHTED-UNCERTAINTY-CAL (Algorithm 5), CLEAN (Algorithm 6) and FILTERED-BY-PREDICTION (Algorithm 7).

WEIGHTED-UNCERTAINTY-CAL (Algorithm 5) takes R_I as input and calculates the uncertainty for each r_i by computing their entropy (that is, frequency of occurrence of slide labels). The lower the entropy, the less uncertain the patch and vice versa. The output is the entropy of r_i, along with records that summarize the diagnosis occurrences and Hamming distance of each element in r_i. The disadvantage of counting the occurrences naively in the entropy calculation is that the most frequent diagnosis in the anatomic site dominates the result and therefore downplays the importance of others. We introduce a weighted occurrence approach to address this issue. The approach counts the diagnosis occurrences by considering the percentage of the diagnosis in the given site and the diagnosis's position in the retrieval results. It calculates the weight of each diagnosis in the anatomic sites by the reciprocal of the number of diagnosis. We normalize the weights such that their summation is equal to a constant N. A diagnosis's final value in r_i is the normalized weight of the diagnosis multiplied by the inverse of position where the diagnosis appears in r_i. Therefore, the same diagnosis can have different weighted occurrences because of its position in r_i. As such, less frequent diagnoses and those with lower Hamming distance (that is, close to the beginning of the retrieval results) gain more importance in the ranking process. As illustrated in Fig. 2, we summarize R_I with three metadata values, S_{lb}, S_m and S_l, to facilitate the subsequent processes. Specifically, S_m is a list that stores tuples of form (index of r_i, entropy, Hamming distance info in μ_{ij}, length of r_i), S_l is an array that only stores the length of r_i and S_{lb} is a nested dictionary that stores the disease occurrences in r_i.

Algorithm 2 Forward Search Algorithm

function FORWARD-SEARCH($m_{i,c+}, k_{succ}, \theta_h, V, H, vEB$)
  $res_+ \leftarrow \{\}$
  for i_+ in $m_{i,c+}$ **do**
    $succ\_cnt, succ_{prev} \leftarrow 0, i_+$
    while $succ\_cnt < k_{succ}$ **do**
      $succ \leftarrow vEB.Successor(succ_{prev})$
      if $succ \in V$ or $succ$ is empty **then**
        break
      else if $H[succ].len() == 0$ **then**
        // The case when the patient is identical to query slide I
        $succ_{prev} \leftarrow succ$
        continue
      else
        // Find the patch with smallest Hamming distance in the same key
        $dist_j, j \leftarrow$ ARGMIN$_j$(HAMMING-DISTANCE($h_i, H[succ]$))
      if $dist_j < \theta_h$ **then**
        $V.insert(succ)$
        $meta \leftarrow H[succ][j]$
        $res_+.insert((dist_j, meta))$
      $succ\_cnt, succ_{prev} \leftarrow succ\_cnt + 1, succ$
  return res_+, V

Algorithm 3 Backward Search Algorithm

function BACKWARD-SEARCH($m_{i,c-}, k_{pred}, \theta_h, V, H, vEB$)
  $res_- \leftarrow \{\}$
  for i_- in $m_{i,c-}$ **do**
    $pred\_cnt, pred_{prev} \leftarrow 0, i_-$
    while $pred\_cnt < k_{pred}$ **do**
      $pred \leftarrow vEB.Predecessor(pred_{prev})$
      if $pred \in V$ or $pred$ is empty **then**
        break
      else if $H[pred].len() == 0$ **then**
        // The case when the patient is identical to query slide I
        $pred_{prev} \leftarrow pred$
        continue
      else
        // Find the mosaic with smallest Hamming distance in the same key
        $dist_j, j \leftarrow$ ARGMIN$_j$(HAMMING-DISTANCE($h_i, H[pred]$))
      if $dist_j < \theta_h$ **then**
        $V.insert(pred)$
        $meta \leftarrow H[pred][j]$
        $res_-.insert((dist_j, meta))$
      $pred\_cnt, pred_{prev} \leftarrow pred\_cnt + 1, pred$
  return res_-

Algorithm 4 Results Ranking Algorithm

function RANKING(R_s, D_inv, N, K)
  if R_s is empty **then return**
  $D\_inv \leftarrow$ NORMALIZE(D_inv, N) ▷ Normalize the reciprocal of diagnosis count so that the sum is equal to N. $N=10$ for the fixed site and $N=30$ for the anatomic site experiments, respectively.
  $S_{lb}, S_m \leftarrow \{\},\{\}$
  $S_l \leftarrow \{\}$
  for each patch's results r_i in R_S **do**
    if r_i is not empty **then**
      $Ent, label\_cnt, dist \leftarrow$ WEIGHTED-UNCERTAINTY-CAL(r_i, D_inv)
      $S_{lb}.insert(i, label\_cnt)$
      $S_m.insert((i, Ent, dist, r_i.len()))$
      $S_l.insert(r_i.len())$
    else
      continue
  $S_m \leftarrow$ CLEAN(S_m, S_l)
  $f \leftarrow$ FILTERED-BY-PREDICTION(S_m, S_{lb})
  $R_{ret}, V \leftarrow \{\},\{\}$
  for e in S_m **do**
    $uncertainty, i \leftarrow e.Ent, e.i$
    if i in f **then**
    continue
    else
      $r_i = R_S[i]$
      for p, μ in r_i **do**

if $uncertainty == 0$ and $\mu.slide\_name$ not in V **then**
$R_{ret}.insert(\mu)$
$V.insert(\mu.slide\_name)$
else if $uncertainty > 0$ and $\mu.Hamming\_dist \leq \theta_{h'}$ and $\mu.slide\_name$ not in V **then**
$R_{ret}.insert(\mu)$
$V.insert(\mu.slide\_name)$
$R_{ret} \leftarrow$ SORTING(R_{ret})
return $R_{ret}[0:K]$

Clean (Algorithm 6) aims to remove outliers and the patches that are less similar to the query in R_I. It takes summaries of patch S_m and S_l from the previous stage as input, removing r whose result length $|r|$ is less than the O_l or greater than the O_h quantiles. Additionally, we take the average of the mean Hamming distances in the top $k_{\theta'_h}$ patches for each $r \in R_I$ as a threshold denoted by $\theta_{h'}$, using this to filter out r whose mean Hamming distance in top $k_{\theta'_h}$ retrieval is greater than $\theta_{h'}$. After cleaning the results, we sort them on the basis of the uncertainty calculated from Weighted-Uncertainty-Cal in ascending order.

Algorithm 5 Uncertainty Calculation

function Weighted-Uncertainty-Cal(r_i, D_inv)
$label\_cnt, dist \leftarrow \{\}, \{\}$
for pos_index, μ in r_i **do**
$label\_cnt[\mu.diagnosis] \leftarrow label\_cnt[\mu.diagnosis] + D\_inv[\mu.diagnosis] \times \frac{1}{pos\_index}$
$dist.insert(\mu.hamming\_dist)$
for lb, cnt in $label\_cnt$ **do**
if $cnt < 1$ **then**
$label\_cnt[lb] \leftarrow 1$
$Ent =$ Entropy($label\_cnt$)
return $Ent, label\_cnt, dist$

Algorithm 6 Results Cleaning

function Clean(S_m, S_l)
//S_m:An array that stores tuples composed of the index i, the entropy, the Hamming distance of all patches, and the total number of patches for each r_i in R_I.
//S_l:An array that stores the total number of patches for each r_i in R_I.
$tmp \leftarrow \{\}$
$k_{\theta'_h} \leftarrow 5$
$O_l \leftarrow 5\%$
$O_h \leftarrow 95\%$
$l \leftarrow 3$
// When the unique results length is less than 3, we keep the original S_m.
if Unique(S_l)$\geq l$ **then**
for res in S_m **do**
if $res.r.len() \leq$ Quantile(S_l, O_l) or $res.r.len() \geq$ Quantile(S_l, O_h) **then**
del res
else
$tmp.insert$(Mean ($res.dist[0 : k_{\theta'_h}]$))
else
for res in S_m **do**
$tmp.insert$(Mean ($res.dist[0 : k_{\theta'_h}]$))
$\theta_{h'} \leftarrow$ Mean(tmp)
for res in S_m **do**
if Mean ($res.dist[0 : m]$) $> \theta'_h$ **then**
del res
$S_m \leftarrow$ Sort-Ascending(S_m, $key = Ent$)
return (S_m)

We can now return the slide from r_i at the beginning of the sorted S_m on the basis of the uncertainty. However, a drawback of this approach is that the low uncertainty of the first several $r \in S_m$ could be caused by the domination of the most frequent diagnoses in the given anatomic site. For example, the most frequent occurrences of the top 5 entries in S_m could be KIRC, BLCA, KIRP, KIRP and KIRP for the urinary site. In this case, the query slide would be diagnosed as KIRP on the basis of the majority vote. Therefore, the first and second entries that dominate the urinary site cases should not be considered during retrieval. We leverage the Filtered-By-Prediction (Algorithm 7) to mitigate this issue. This function takes the summation of the diagnosis occurrences from the top k_f certain elements in S_m. It first uses the diagnosis with the maximum score as a pseudo-ground truth diagnosis from the top k_f most certain elements. Afterwards, it removes all elements whose maximum occurrence diagnosis disagrees with the pseudo-ground truth. To return final results $R_{\text{ret},I}$ of slide query I, we take the slide name and its diagnosis in r_i pointed to by S_m one by one. If the uncertainty of r_i is zero, we take all (p_i, μ_i). Otherwise, we use $\theta_{h'}$ again to ignore (p_i, μ_i) whose Hamming distance is greater than the threshold. We sort $R_{\text{ret},I}$ first by uncertainty in ascending order then by Hamming distance in descending order if the uncertainty is a tie.

Training details of VQ-VAE. We randomly sampled 20 1,024×1,024 patches at ×20 magnification from 9,286 diagnostic slides across 29 TCGA projects. All patches were converted from RGB to Pytorch tensors, then normalized such that all values lie within $[-1, 1]$. The model was trained using the Adam optimizer with a learning rate of 10^{-3} without weight decay and without AMSgrad. We used default settings for other hyperparameters in Adam (β_1=0.9 and β_1=0.999, and $\epsilon = 10^{-8}$). We trained our model with a batch size of 4 for 10 epochs. We applied gradient clipping techniques by setting the gradient threshold to 1.0. The hyperparameter α in VQ-VAE was also set to 1 (equation 2). In the patch speed experiments, we followed the same training receipts above, except that we trained on a patch of size 256×256.

Ablation study. We conducted two ablation studies: one to show that using a pretrained VQ-VAE does not confer an advantage when compared with the Yottixel pipeline and a second to test the benefit of each function in our ranking module. For the first study, we conducted a top-5 nearest neighbour search on Kather100k by using the VQ-VAE integer index and the Yottixel index. For the ranking module, we compared the performance of the following four settings: (1) Naive: removing Clean and Filtered-By-Prediction and treating each diagnosis occurrence in the mosaic retrieval result equally (that is, replacing the assignment in line 4 in Algorithm 5 with 1). (2) +Weighed count: applying Uncertainty-Cal to the ranking module only. (3) +Clean: applying Uncertainty-Cal and Clean to the ranking module. (4) +Filter: applying all functions to the ranking module.

Algorithm 7 Results Filtering by Prediction

function Filtered_By_Prediction(S_m, S_{lb})
//S_m: An array that stores tuples composed of the index i, the entropy, the Hamming distance of all patches, and the total number of patches for each r_i in R_I.
//S_{lb}: A nested hash table that stores the index of r in R_I as the key and its weighted diagnosis occurrences table as value.
$cnt \leftarrow \{\}$
$k_f \leftarrow 5$
for s_m in $S_m[0:k_f]$ **do**
// Calculate the score of each diagnosis
for d in $S_{lb}[s_m.i]$ **do**
$cnt[d] \leftarrow cnt[d] + S_{lb}[s_m.i][d]$
$plb\_list \leftarrow$ Sort-Descending(cnt)
$p \leftarrow 0$
// A while loop is used here to avoid the case that the plb remove all s_m.
while do
$plb \leftarrow plb\_list[p]$
$removed \leftarrow \{\}$
for s_m in $S_m[0:k_f]$ **do**
$pred \leftarrow$ Max($S_{lb}[s_m.i]$)
if $pred \neq plb$ **then**
$removed.insert(s_m.i)$
if $removed.len() \neq k_f$ **then**
break
else
$p \leftarrow p + 1$
return $removed$

Visualization. We build confusion matrices for each site using each slide diagnosis as ground truth along the y axis and Mv(ret[:k]) as predicted diagnosis along the x axis. For the Hamming distance matrix, we inspect the Hamming distance between the query slide and each result in ret[:k] one by one, adding the Hamming distance to the associated diagnosis label and infinity to others. The infinity here is defined as Hamming distance threshold θ_h plus 1, as θ_h is the maximum distance we can have in our pipeline. The final Hamming distance matrix is obtained by dividing the total number of slides in the given anatomic site.

Evaluation metrics. For all experiments, we remove the slide with the same patient identification as the query slide in the database (that is, leave-one-patient-out evaluation). Since the ability to diagnose each subtype of cancer is the paramount task in search-based disease classification[53] and a false-negative case does harm to the patient's health, we evaluate the accuracy of each subtype by dividing the number of correct prediction by the total number of subtype in the site. The prediction of a query is determined by the majority vote of slide label in the top-k retrieval (that is, MV@k). Therefore, the overall mean accuracy mMV@k for a given subtype L is defined as:

$$\text{mMV@k} = \frac{1}{Q}\sum_{i=1}^{Q} \mathbb{1}[L = \text{Mv}(\text{ret}_i[:k])], \tag{9}$$

where Q is the number of slides for the given cancer subtype and MV(ret$_i$[:k]) is the predicted diagnosis of S_i taken from the majority vote of the top-k retrieval ret$_i$. We use k=5 for disease subtyping and k=10 for anatomic site prediction for comparative analysis with ref. [35]. Similar to the related works that focused on top retrieval[38,54] (that is, $k \leq 5$), we also report k=1 and k=3 to evaluate the performance of the top hit. In addition, we use the definition of mAP@k below to provide another perspective.

$$\mathrm{AvP@k} = \frac{\sum_{i=1}^{k} \mathrm{Prec}_i \cdot \mathrm{Rel}_i}{k}, \quad (10)$$

$$\mathrm{mAP@k} = \frac{\mathrm{AvP@k}}{|Q|} \quad (11)$$

where AvP@k, Prec_i and Rel_i respectively denote average precision up to k, precision at i and relevant indicator at i (that is, 1 if the item at the position i is relevant and 0 otherwise). For AvP@k, we adapted the original definition[55] by changing the denominator to the maximum number of slides whose labels match the query's. We did this adaption to ensure that the mAP@k in each subtype has the same scale. We used $k=5$ for the disease subtyping task.

Computational hardware and software. We stored all WSIs, patches, segmentation masks and mosaics across multiple disks with total size around 27TB. Segmentation, patching, mosaic extraction and search of WSIs were performed on a CPU (AMD Ryzen Threadripper 3970X 32-Core Processor). The VQ-VAE pretraining and feature extraction were performed on 4 NVIDIA 2080Ti GPU. The whole SISH pipeline was written in Python (version 3.7.0), with the following external package: h5py (2.10.0), matplotlib(3.3.0), numpy (1.19.1), opencv-python (4.3.0.38), pillow (7.2.0), pandas (1.1.0), scikit-learn (0.23.1), seaborn (0.10), scikit-image (0.17.2), torchvision (0.6.0), tensorboard (2.3.0) and tqdm (4.48.0). We used Pytorch (1.5.0) for deep learning. All plots were created by matplotlib (version 3.2.2) and seaborn (version 0.10.1). The pie charts were created using Adobe Illustrator.

WSI datasets. There are three datasets in our slide-level retrieval experiment: diagnostic slides in The Cancer Genome Atlas (TCGA), Clinical Proteomic Tumor Analysis Consortium (CPTAC) and BWH in-house data.

TCGA and CPTAC diagnostic slides. We downloaded all diagnostic slides from the TCGA genomic data commons and the CPTAC websites. To fairly compare with Yottixel, we used slides from the same 13 anatomic sites for anatomic site retrieval and the same 29 diagnoses for disease subtype retrieval in the TCGA. There are 503 CPTAC-Clear Cell Renal Cell Carcinoma slides from 216 patients, 544 CPTAC-UCEC slides from 240 patients, 679 CPTAC-LUSC slides from 210 patients, 669 CPTAC-LUAD slides from 224 patients and 283 CPTAC-SKCM slides from 93 patients. All slides were processed at ×20. Detailed slide and patient numbers are reported in Supplementary Table 4.

In-house diagnostic slides. From the WSI database at Brigham and Women's Hospital, we collected 8,035 diagnostic WSIs that span 9 anatomic sites with 37 primary cancer subtypes. Further details of the dataset are available in Supplementary Table 5. All slides were processed at ×20.

Patch datasets. There are five datasets in the patch retrieval experiments: colon tissues (Kather100k), BWH prostate, lung tissues (WSSS4LUAD), atlas of digital pathology (Atlas) and breast tissues (BCSS). We provide details for each dataset below.

Kather100k[45]. The data contain 100,000 224×224 tissue patches from colon at ×20 without colour normalization, from the National Center for Tumor Diseases (NCT) biobank and the University Medical Center Mannheim (UMM). The tissue patches are adipose (ADI), background (BACK), debris (DEB), lymphocytes (LYM), mucus (MUC), smooth muscle (MUS), normal colon mucosa (NORM), cancer-associated stroma (STR) and colorectal adenocarcinoma epithelium (TUM).

BWH in-house prostate. In this cohort, each whole-slide image is from a different patient. For the prostate data used in patch-level retrieval, we collected 23 slides at ×20, scanned the slides using Hamamatsu S210 and annotated regions in each slide by gleason score 3, 4, 5 or normal, where the number of patches for each category is 2,355, 2,289, 171 and 2,000, respectively. Detailed slide and patient numbers are reported in Supplementary Table 5.

WSSSS4LUAD[46]. We used the training part of the data that contains 10,091 annotated patches from 49 WSIs from the Guangdong Provincial People's Hospital (GDPH) and 14 WSIs from lung adenocarinoma in TCGA. Each patch ranges from 150 to 300×150 to 300 pixels and has one annotation from ['stroma', 'stroma + tumour', 'tumour', 'normal']. We only considered the patch with annotation stroma, tumour and normal, which resulted in 4,698 patches. The slides were scanned using Leica and Aperio-AT2 at ×10.

Atlas[47]. This dataset contains 17,668 272×272 patches from 100 glass slides scanned by Huron TissueScope LE1.2 in a private cohort mentioned in ref. [47] with 38 histology tissue types annotation organized in 3 hierarchies. The patches were cropped from 100 slides at ×40 from various organs. We only considered the label in the first hierarchy which are: epithelial (E), connective proper (C), blood (H), skeletal (S), adipose (A), muscular (M), nervous (N), glandular (G) and transport vessel (T). Multiple labels for a patch is common in this dataset.

BCSS[48]. This dataset contains the pixel-level annotation from 151 TCGA-breast WSI at ×40 and are grouped into five categories: tumour, stroma, inflammatory, necrosis and other. Specifically, the tumour class includes the predominant tumour, ductal carcinoma in situ (DCIS) and angioinvasion, and the inflammatory infiltrates include plasma cell, lymphocytes and other infiltrates. We only considered the first four classes since they are more clinically relevant. We cropped each patch into 256×256 and assigned the patch with all labels appearing in the region. The number of total patches is 7,691. Having multiple labels for a patch is possible in this dataset. We considered the search correct if most of the categories in the search result match one of the categories in the query patch.

WSI processing. *Segmentation.* We used the automatic segmentation tool in clustering-constrained attention multiple instance learning[9] (CLAM) to generate the segmentation mask for each slide. The tool first applies a binary threshold to a downsampled whole-slide image on the hue, saturation and value colour space to generate a binary mask and then refines the mask by median blurring and morphological closing to remove the artefacts. After getting the approximate contours of the tissue, the tool filters out the tissue contours and cavities on the basis of a certain area threshold.

Patching. After segmentation, we cropped the contours into 1,024×1,024 patches without overlapping at ×20. For ×40 whole slides, we first cropped them into 2,048×2,048 patches and then downsampled them to 1,024×1,024 to get the equivalent patches at ×20.

Mosaic generation. We followed the mosaic generation process proposed in the Yottixel paper[36]. The algorithm first applies K-mean clustering to the RGB features extracted from each patch, with number of cluster $K=9$. Within each cluster, we run K-means clustering again on the coordinate of each patch by setting the number of clusters to 5% of the cluster size. If the number of clusters is <1 in the second stage, we took all coordinates within that cluster. Except for the number of clusters, we used all default settings in Scikit-learn for K-means clustering. To get better quality of mosaics, we collected 101 patches for both debris/pen smudges and tissue to train a logistic regression on the basis of local binary pattern histogram feature to remove the unmeaningful regions. We used the default setting from the Scikit-learn package in logistic regression and used the rotate invariant binary pattern from Scikit-image package with $P=8$ and $R=1$. The bin number of the histogram was set to 128.

Artefacts removal. Rarely, we found that the generated mosaic might contain patches with nearly completely white background. We removed such patches from the mosaic if the white regions accounted for over 90% of pixels in each patch. We applied the binary threshold method in OpenCV with a threshold value of 235 to determine the area (in number of pixels) of white regions.

Ethics oversight. The retrospective analysis of archival pathology slides was approved by the Mass General Brigham (MGB) IRB office under protocol 2020P000233.

Reporting summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Data availability

The TCGA diagnostic whole-slide data and corresponding labels are available from the National Institutes of Health (NIH) genomic data commons (https://portal.gdc.cancer.gov). The CPTAC whole-slide data and the corresponding labels are available from the NIH cancer imaging archive (https://cancerimagingarchive.net/datascope/cptac). Supplementary Table 19 provides access links to publicly available patch-level datasets. Source data for the figures are provided with this paper. All reasonable requests for academic use of in-house raw and analysed in-house data can be addressed to the corresponding author. All requests will be promptly reviewed to determine whether the request is subject to any intellectual property or patient-confidentiality obligations, will be processed in concordance with institutional and departmental guidelines, and will require a material transfer agreement.

Code availability

All scripts, checkpoints, preprocessed mosaics and pre-built database to reproduce the findings are available at https://github.com/mahmoodlab/SISH. We implemented all our methods in Python and used Pytorch as the primary package for training VQ-VAE. The code base is made available for non-commercial and academic purposes.

Received: 19 July 2021; Accepted: 15 July 2022;
Published online: 10 October 2022

References

1. Snead, D. R. et al. Validation of digital pathology imaging for primary histopathological diagnosis. *Histopathology* **68**, 1063–1072 (2016).
2. Mukhopadhyay, S. et al. Whole slide imaging versus microscopy for primary diagnosis in surgical pathology: a multicenter blinded randomized noninferiority study of 1992 cases (pivotal study). *Am. J. Surgical Pathol.* **42**, 39–52 (2018).
3. Azam, A. S. et al. Diagnostic concordance and discordance in digital pathology: a systematic review and meta-analysis. *J. Clin. Pathol.* **74**, 448–455 (2021).
4. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
5. Esteva, A. et al. A guide to deep learning in healthcare. *Nat. Med.* **25**, 24–29 (2019).
6. Esteva, A. et al. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **542**, 115–118 (2017).
7. Bera, K., Schalper, K. A. & Madabhushi, A. Artificial intelligence in digital pathology–new tools for diagnosis and precision oncology. *Nat. Rev. Clin. Oncol.* **16**, 703–715 (2019).
8. Niazi, M. K. K., Parwani, A. V. & Gurcan, M. N. Digital pathology and artificial intelligence. *Lancet Oncol* **20**, e253–e261 (2019).
9. Lu, M. Y. et al. Data-efficient and weakly supervised computational pathology on whole-slide images. *Nat. Biomed. Eng.* **5**, 555–570 (2021).
10. Chen, R. J. et al. Pathomic fusion: an integrated framework for fusing histopathology and genomic features for cancer diagnosis and prognosis. *IEEE Trans. Med. Imaging* **41**, 757–770 (2020).
11. Lu, M. Y. et al. AI-based pathology predicts origins for cancers of unknown primary. *Nature* **594**, 106–110 (2021).
12. Lipkova, J. et al. Deep learning-enabled assessment of cardiac allograft rejection from endomyocardial biopsies. *Nat. Med.* **28**, 575–582 (2022).
13. Chen, R. J. et al. Scaling vision transformers to gigapixel images via hierarchical self-supervised learning. In *Proc. IEEE/CVF Conference on Computer Vison and Pattern Recognition* 16144–16155 (2022).
14. Chen, R. J. et al. Multimodal co-attention transformer for survival prediction in gigapixel whole slide images. In *Proc. IEEE/CVF Conference on Computer Vison and Pattern Recognition* 4015–4025 (2021).
15. Chen, R. J. et al. Whole slide images are 2D point clouds: context-aware survival prediction using patch-based graph convolutional networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* 339–349 (2021).
16. Coudray, N. et al. Classification and mutation prediction from non–small cell lung cancer histopathology images using deep learning. *Nat. Med.* **24**, 1559–1567 (2018).
17. Campanella, G. et al. Clinical-grade computational pathology using weakly supervised deep learning on whole slide images. *Nat. Med.* **25**, 1301–1309 (2019).
18. Komura, D. & Ishikawa, S. Machine learning methods for histopathological image analysis. *Comput. Struct. Biotechnol. J.* **16**, 34–42 (2018).
19. Qi, X. et al. Content-based histopathology image retrieval using CometCloud. *BMC Bioinformatics* **15**, 287 (2014).
20. Zhang, X. et al. Towards large-scale histopathological image analysis: hashing-based image retrieval. *IEEE Trans. Med. Imaging* **34**, 496–506 (2014).
21. Sridhar, A., Doyle, S. & Madabhushi, A. Content-based image retrieval of digitized histopathology in boosted spectrally embedded spaces. *J. Pathol. Inform.* **6**, 41 (2015).
22. Kwak, J. et al. Automated prostate tissue referencing for cancer detection and diagnosis. *BMC Bioinformatics* **17**, 227 (2016).
23. Sparks, R. & Madabhushi, A. Out-of-sample extrapolation utilizing semi-supervised manifold learning (OSE-SSL): content based image retrieval for histopathology images. *Sci. Rep.* **6**, 27306 (2016).
24. Jiang, M. et al. Scalable histopathological image analysis via supervised hashing with multiple features. *Med. Image Anal.* **34**, 3–12 (2016).
25. Shi, X. et al. Supervised graph hashing for histopathology image retrieval and classification. *Med. Image Anal.* **42**, 117–128 (2017).
26. Komura, D. et al. Luigi: large-scale histopathological image retrieval system using deep texture representations. Preprint at *bioRxiv* https://doi.org/10.1101/345785 (2018).
27. Schaer, R. et al. Deep learning-based retrieval system for gigapixel histopathology cases and the open access literature. *J. Pathol. Inform.* **10**, 19 (2019).
28. Ma, Y. et al. Breast histopathological image retrieval based on latent dirichlet allocation. *IEEE J. Biomed. Health Inform.* **21**, 1114–1123 (2016).
29. Zheng, Y. et al. Histopathological whole slide image analysis using context-based CBIR. *IEEE Trans. Med. Imaging* **37**, 1641–1652 (2018).
30. Akakin, H. C. & Gurcan, M. N. Content-based microscopic image retrieval system for multi-image queries. *IEEE Trans. Inf. Technol. Biomed.* **16**, 758–769 (2012).
31. Zheng, Y. et al. Encoding histopathological WSIs using GNN for scalable diagnostically relevant regions retrieval. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* 550–558 (2019).
32. Mehta, N. et al. Content based sub-image retrieval system for high resolution pathology images using salient interest points. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society* 3719–3722 (2009).
33. Schaumberg, A. J. et al. Interpretable multimodal deep learning for real-time pan-tissue pan-disease pathology search on social media. *Mod. Pathol.* **33**, 2169–2185 (2020).
34. Hegde, N. G. et al. Similar image search for histopathology: Smily. *npj Digit. Med.* **2**, 56 (2019).
35. Kalra, S. et al. Pan-cancer diagnostic consensus through searching archival histopathology images using artificial intelligence. *npj Dig. Med.* **3**, 31 (2020).
36. Kalra, S. et al. Yottixel–an image search engine for large archives of histopathology whole slide images. *Med. Image Anal.* **65**, 101757 (2020).
37. Hemati, S. et al. CNN and deep sets for end-to-end whole slide image representation learning. *Proc. Mach. Learn Res.* **143**, 301–311 (2021).
38. Riasatian, A. et al. Fine-tuning and training of densenet for histopathology image representation using TCGA diagnostic slides. *Med. Image Anal.* **70**, 102032 (2021).
39. Weinstein, J. N. et al. The Cancer Genome Atlas Pan-Cancer analysis project. *Nat. Genet.* **45**, 1113–1120 (2013).
40. Edwards, J. N. The CPTAC Data Portal: a resource for cancer proteomics research. *J. Proteome Res.* **14**, 2707–2713 (2015).
41. Oord, A. V. D., Vinyals, O. & Kavukcuoglu, K. Neural discrete representation learning. In *Proc. Advances in Neural Information Processing Systems* 30 (2017).
42. Bentley, J. L. Multidimensional binary search trees used for associative searching. *Commun. ACM* **18**, 509–517 (1975).
43. Kingma, D. P.& Welling, M. Auto-encoding variational Bayes. In *International Conference on Learning Representations* (2014).
44. Huang, G. et al. Densely connected convolutional networks. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition* 4700–4708 (2017).
45. Kather, J. N. et al. Predicting survival from colorectal cancer histology slides using deep learning: a retrospective multicenter study. *PLoS Med* **16**, e1002730 (2019).
46. Han, C. et al. Multi-layer pseudo-supervision for histopathology tissue semantic segmentation using patch-level classification labels. *Med. Image Anal.* **80**, 102487 (2022).
47. Hosseini, M. S. et al. Atlas of digital pathology: a generalized hierarchical histological tissue type-annotated database for deep learning. In *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition* 11747–11756 (2019).
48. Amgad, M. et al. Structured crowdsourcing enables convolutional segmentation of histology images. *Bioinformatics* **35**, 3461–3467 (2019).
49. Ghassemi, M. et al. The false hope of current approaches to explainable artificial intelligence in health care. *Lancet Digit. Health* **3**, e745–e750 (2021).
50. Liu et al. Deep Supervised Hashing for Fast Image Retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition* 2064–2072 (2016).
51. Noh et al. Large-scale image retrieval with attentive deep local features. In *Proceedings of the IEEE international conference on computer vision* 3456–3465 (2017).
52. Xu, K. et al. How powerful are graph neural networks? In *International Conference on Learning Representations* (2019).
53. Tizhoosh, H. R. et al. Searching images for consensus: can AI remove observer variability in pathology? *Am. J. Pathol.* **191**, 1702–1708 (2021).
54. Afshari, M. & Tizhoosh, H. R. A similarity measure of histopathology images by deep embeddings. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society* 3447–3450 (2021).
55. Manning, C. D., Raghavan, P. & Schütze, H. Introduction to information retrieval. *Nat. Lang. Eng.* **16**, 100–103 (2010).

Acknowledgements

We thank A. Bruce for scanning internal cohorts of patient histology slides at Brigham and Women's Hospital (BWH); K. Bronstein, L. Cirelli and E. Askeland for querying the BWH slide database and retrieving archival slides; C. Li for assistance with EMRs and the Research Patient Data Registry (RPDR); M. Bragg, T. Mellen, S. Zimmet and T. A. Mages for logistical support. This work was supported in part by National Institute of General Medical Sciences (NIGMS) R35GM138216 (to F.M.), Brigham President's Fund, BWH and MGH Pathology, BWH Precision Medicine Program, Google Cloud Research Grant and Nvidia GPU Grant Program. M.Y.L. was additionally supported by the Tau Beta Pi Fellowship and the Siebel Foundation. T.Y.C. was additionally supported by the NIH National Cancer Institute (NCI) Ruth L. Kirschstein National Service Award, T32CA251062. The content is solely the responsibility of the authors and does not reflect the official views of the NIGMS, NIH or the NCI.

Author contributions

C.C. and F.M. conceived the study and designed the experiments. C.C., A.J.S. and F.M. designed the algorithmic pipeline. C.C. performed the experiments. C.C., M.Y.L.,

D.F.K.W., T.Y.C., A.J.S. and F.M. analysed the results. All authors contributed to writing the manuscript. F.M. supervised the research.

Competing interests

F.M. receives research support from Leidos Biomedical Research, Inc. for projects unrelated to this study. The authors declare no other competing interests.

Additional information

Extended data is available for this paper at https://doi.org/10.1038/s41551-022-00929-8.

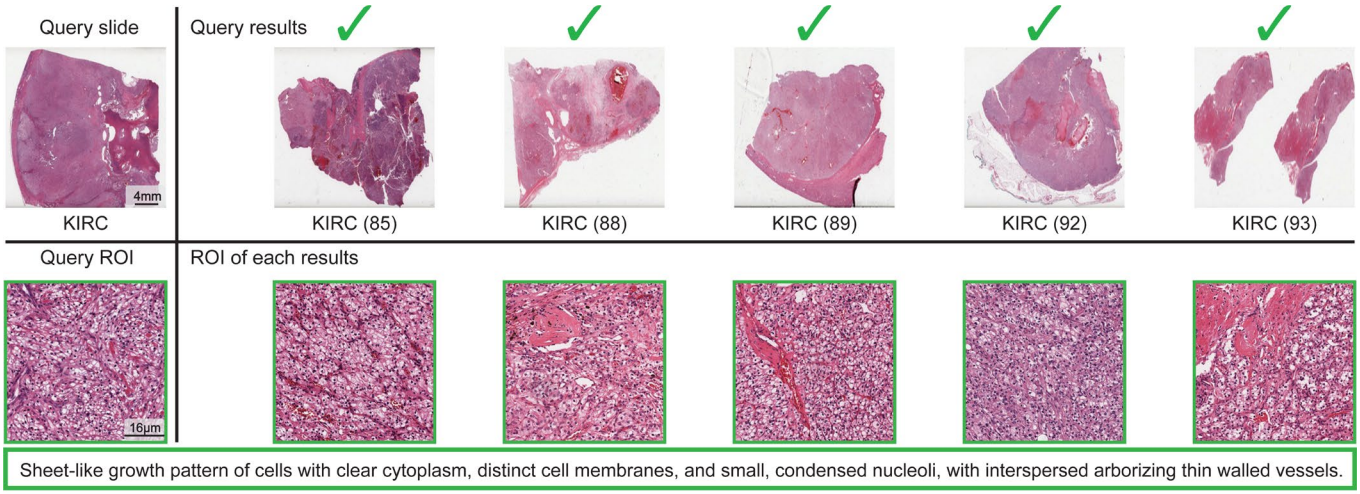
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41551-022-00929-8.

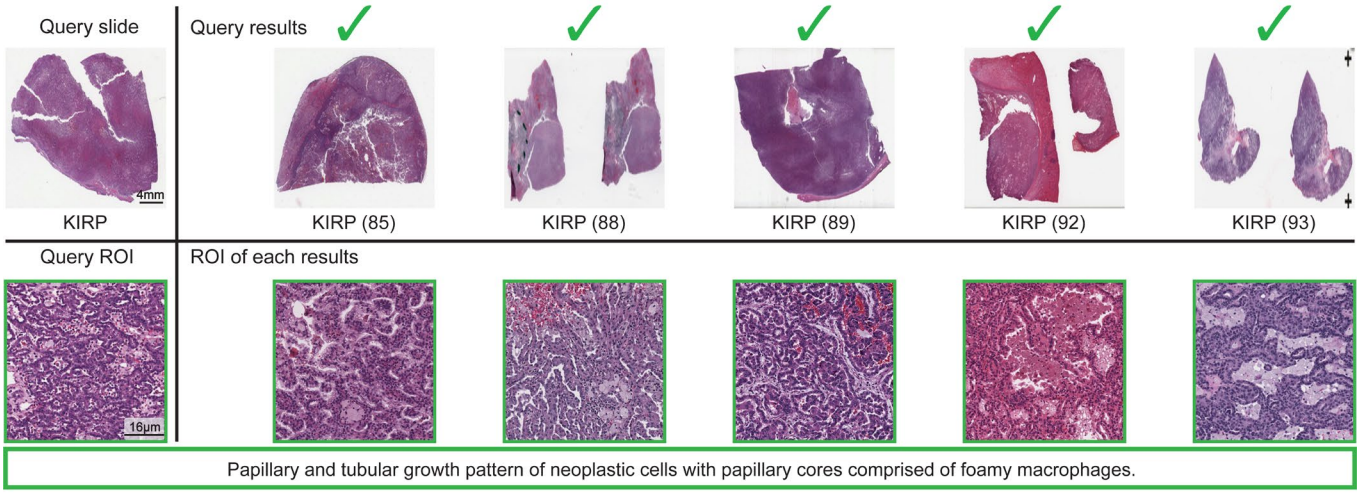
Correspondence and requests for materials should be addressed to Faisal Mahmood.

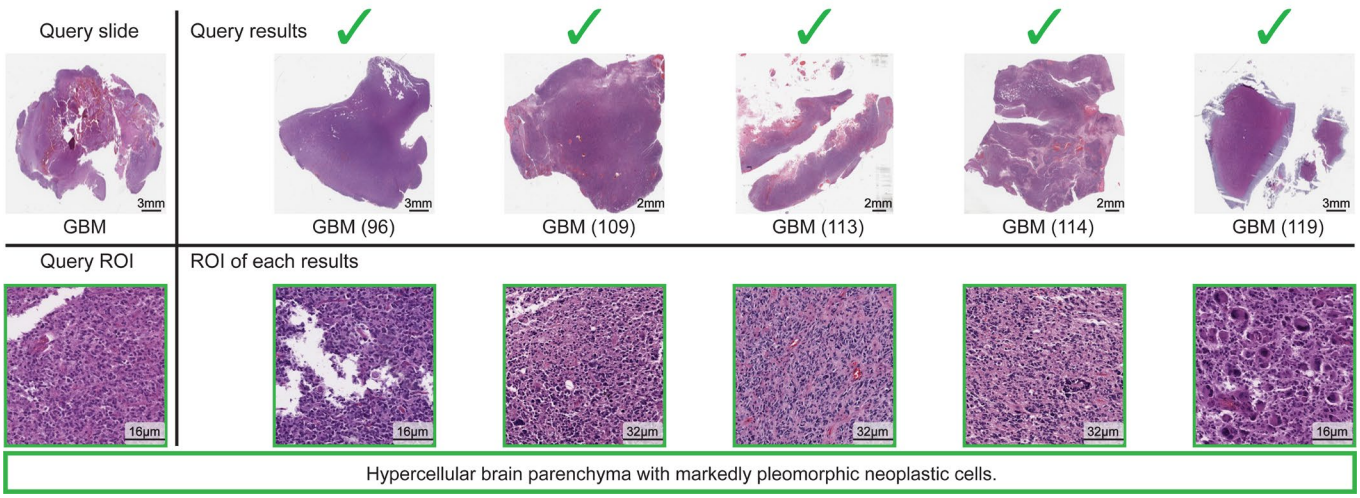
Peer review information *Nature Biomedical Engineering* thanks Richard Levenson and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permissions information is available at www.nature.com/reprints.

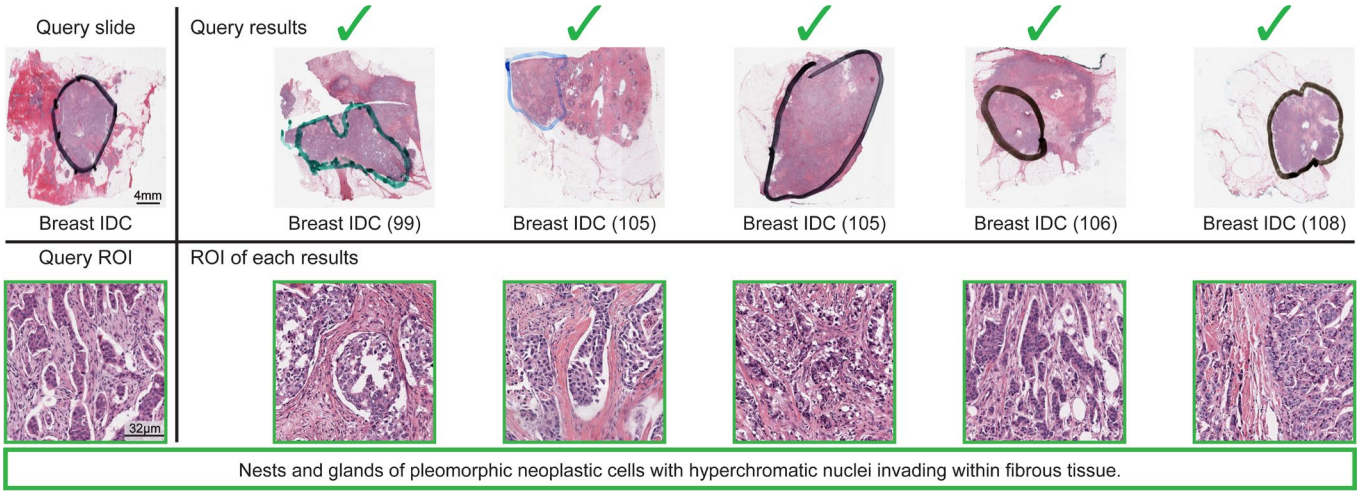
Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

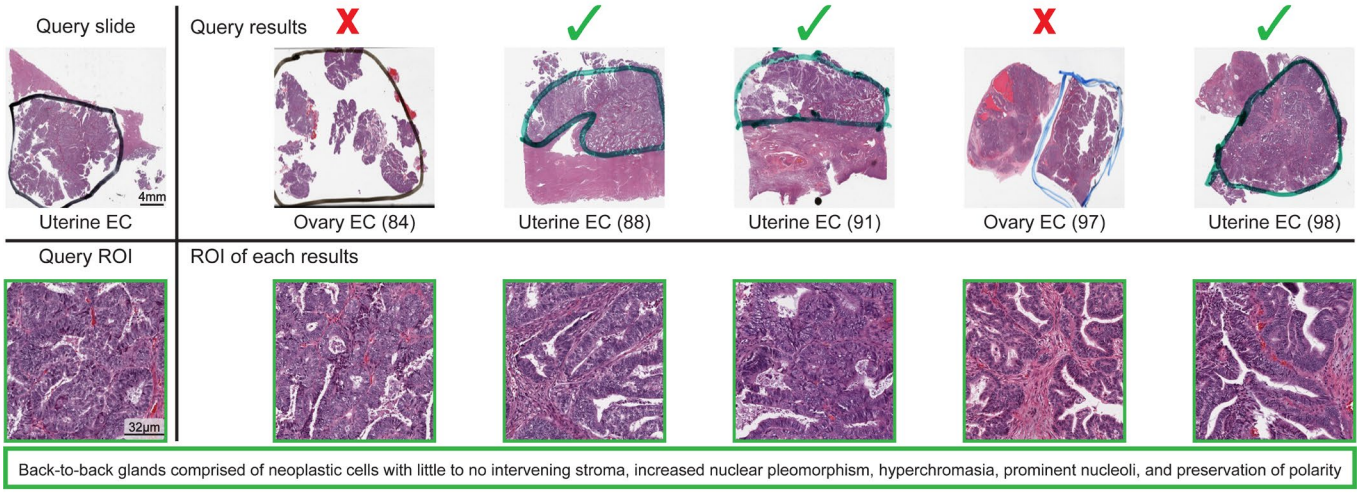


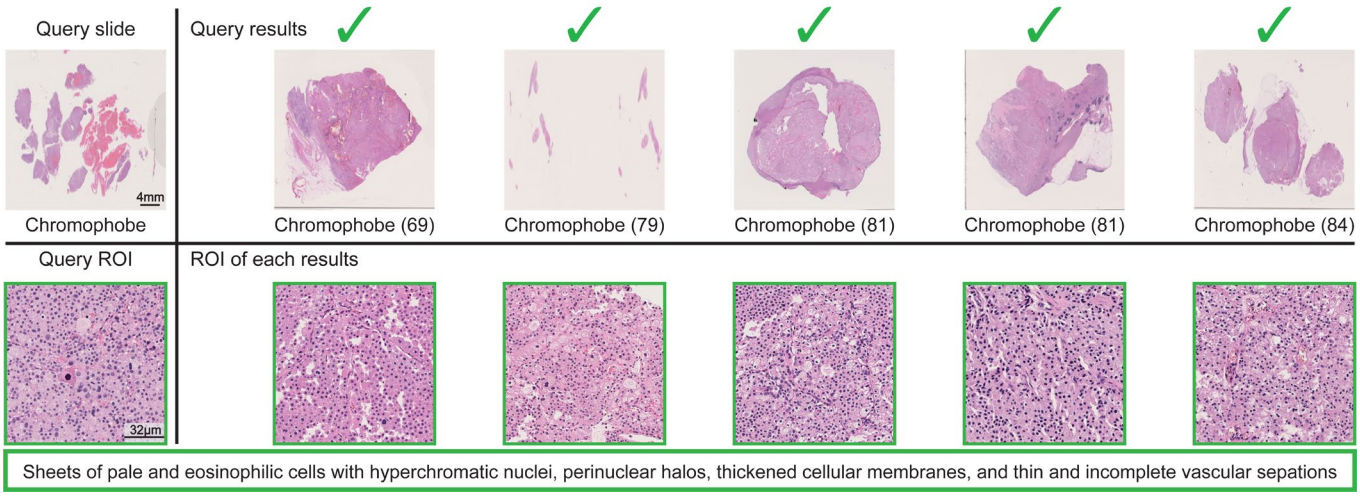




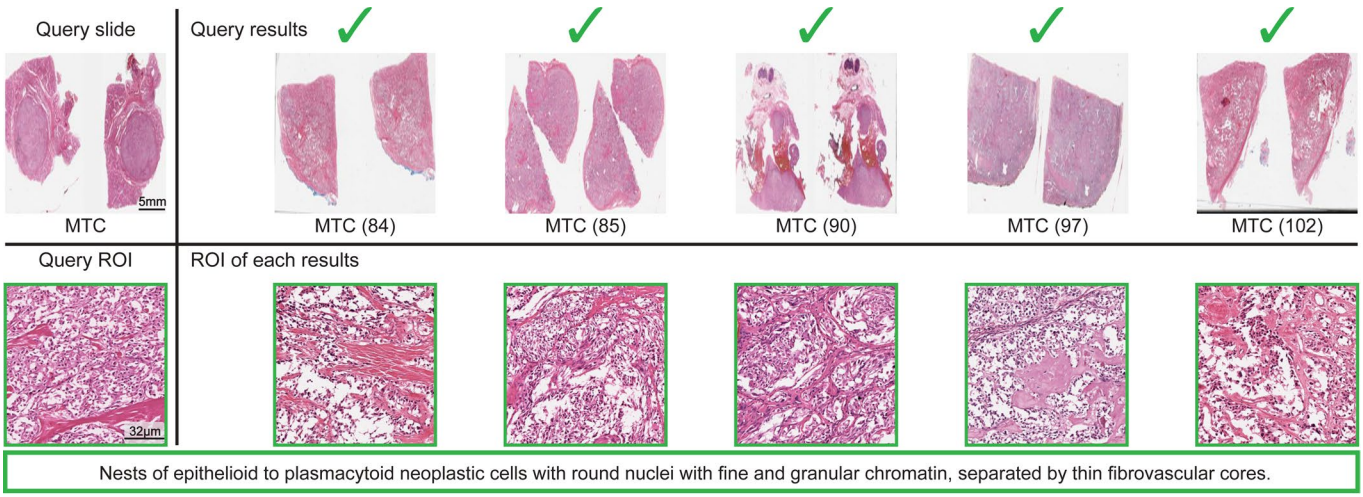
Extended Data Fig. 1 | Examples of fixed-site disease subtype retrieval in TCGA cohort. Examples of retrieval slides and corresponding ROI identified by SISH in TCGA-KIRC, TCGA-KIRP, and TCGA-GBM. The green border of ROIs denotes the selected regions match the histological features annotated by the pathologist. The number in parentheses is the Hamming distance between the query slide and each result, determined by the identified ROI in each WSI. Each row shares the same scale bar unless specified otherwise.

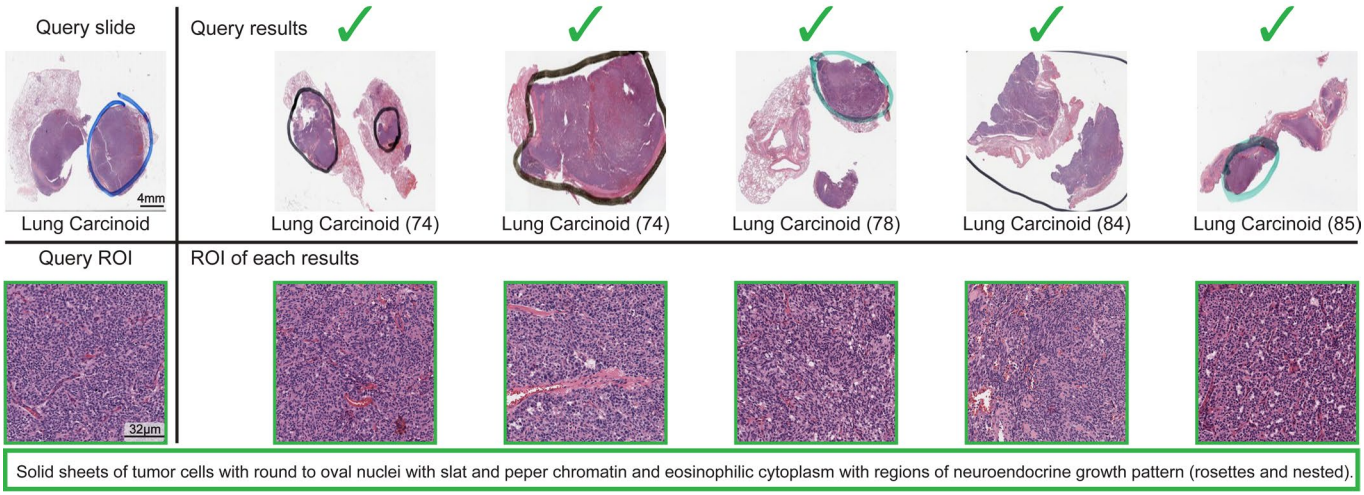


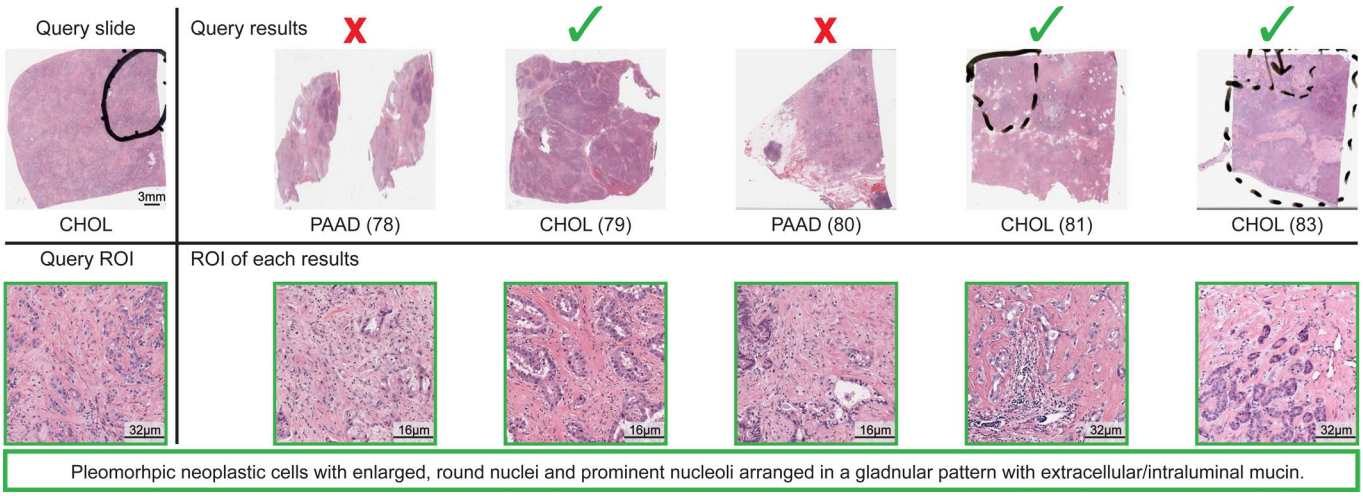




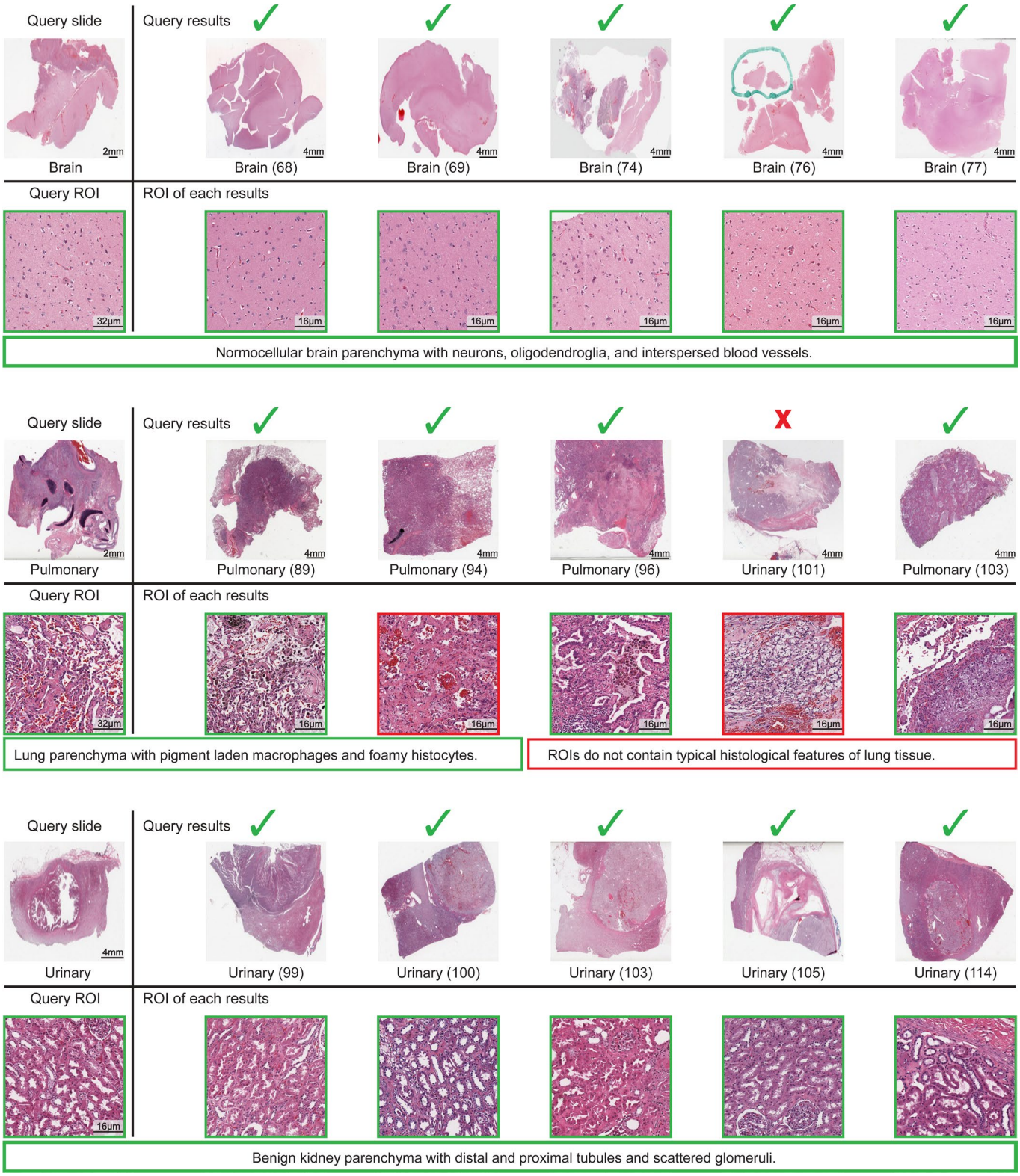
Extended Data Fig. 2 | Examples of fixed-site disease subtype retrieval in independent cohort. Examples of retrieval slides and corresponding ROI identified by SISH in Breast Invasive Ductial Carcinoma (Breast IDC), Uterine Endometriod Carcinoma (Uterine EC), and Kidney Chromophobe. The green border of ROIs denotes the selected regions match the histological features annotated by the pathologist. The number in parentheses is the Hamming distance between the query slide and each result, determined by the identified ROI in each WSI. Each row shares the same scale bar unless specified otherwise. We found that SISH is sometimes confused with Ovary EC and Uterine EC, which is reasonable as both diseases have similar morphology.



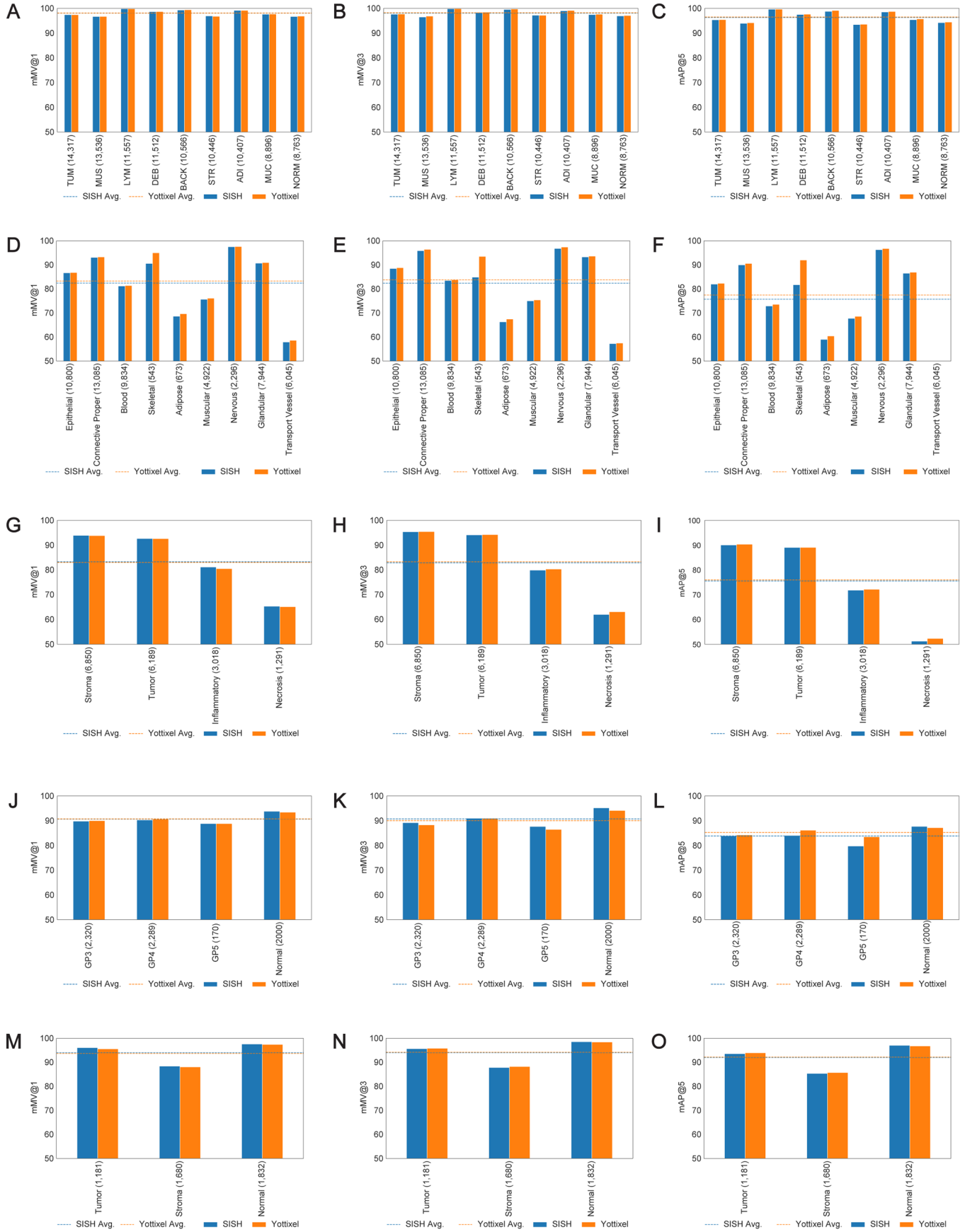




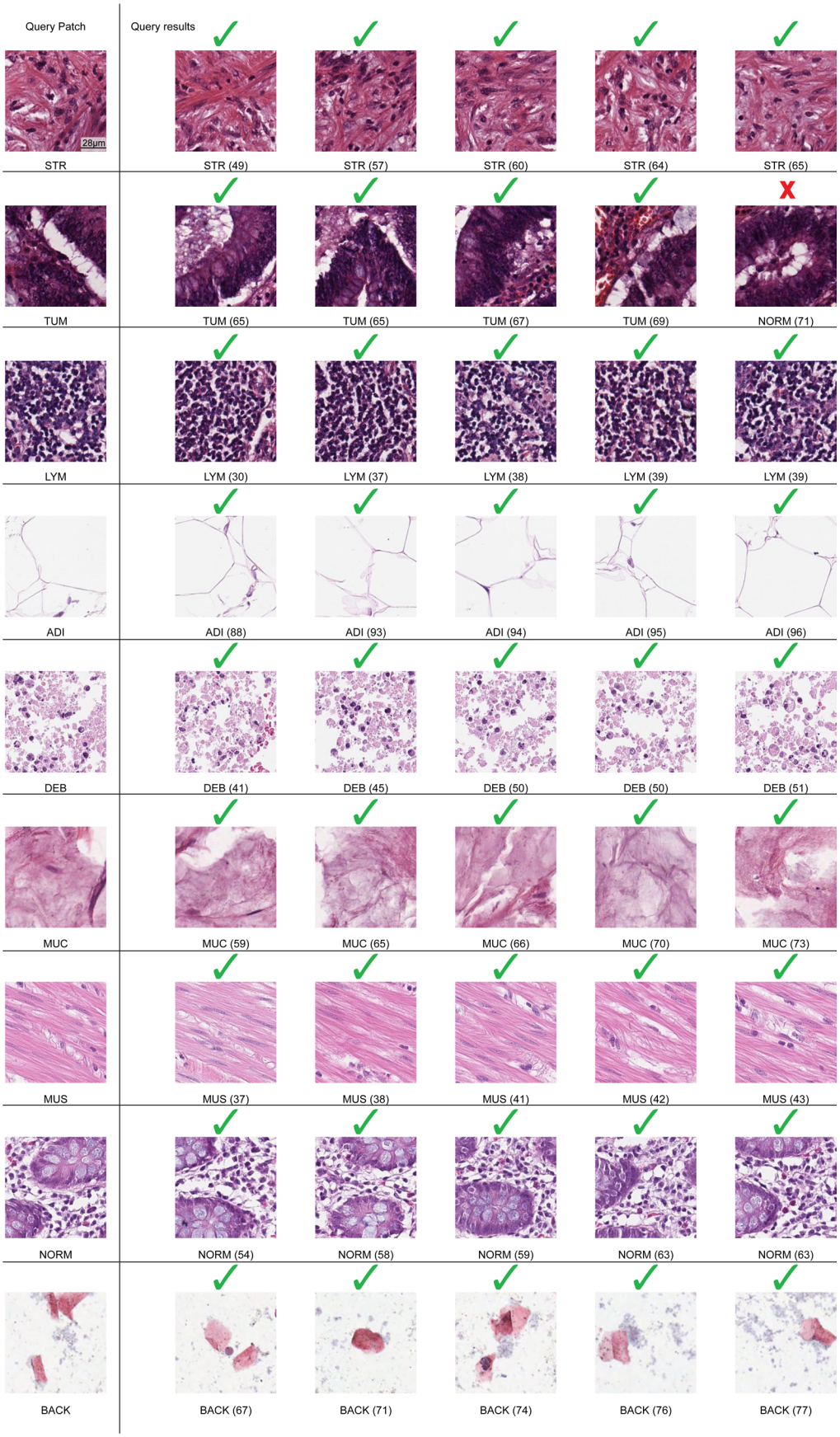
Extended Data Fig. 3 | Examples of fixed-site retrieval on rare cancer subtype. Examples of retrieval slides and corresponding ROI identified by SISH in Medullary Thyroid Carcinoma (MTC), Lung Carcinoid and Cholangiocarcinoma (CHOL). The green border of ROIs denotes the selected regions match the histological features annotated by the pathologist. The number in parentheses is the Hamming distance between the query slide and each result, determined by the identified ROI in each WSI. Each row shares the same scale bar unless specified otherwise. We found that SISH is sometimes confused with Cholangiocarcinoma and Pancreatic Adenocarcinoma (PAAD), which is reasonable as both diseases have similar morphology.



Extended Data Fig. 4 | Examples of anatomic site retrieval in TCGA cohort. Examples of retrieval slides and corresponding ROI identified by SISH in Brain, Pulmonary and Kidney. The visualization showed that SISH can also identified regions that contain typical histological features for a site. The green border denotes the regions that contain typical features while the red borders denote the failure cases. The number in parentheses is the Hamming distance between the query slide and each result, determined by the identified ROI in each WSI. Each row shares the same scale bar unless specified otherwise.

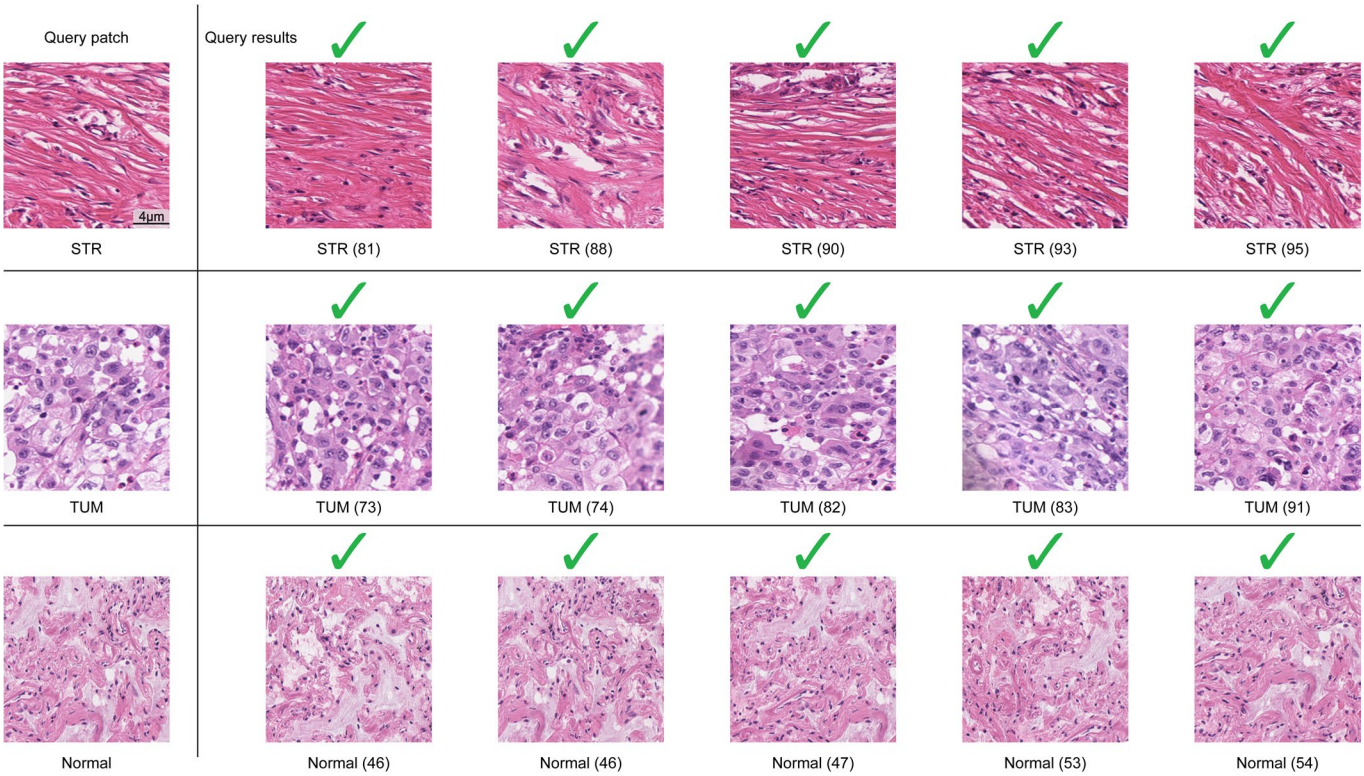


Extended Data Fig. 5 | mMV@1, mMV@3 and mAP@5 results on patch data. A-C: Kather100k, D-F Atlas, G-I Breast, J-L BWH prostate and M-O WSSS4LUAD. On all datasets and metrics, the performance of SISH was on par with Yottixel.

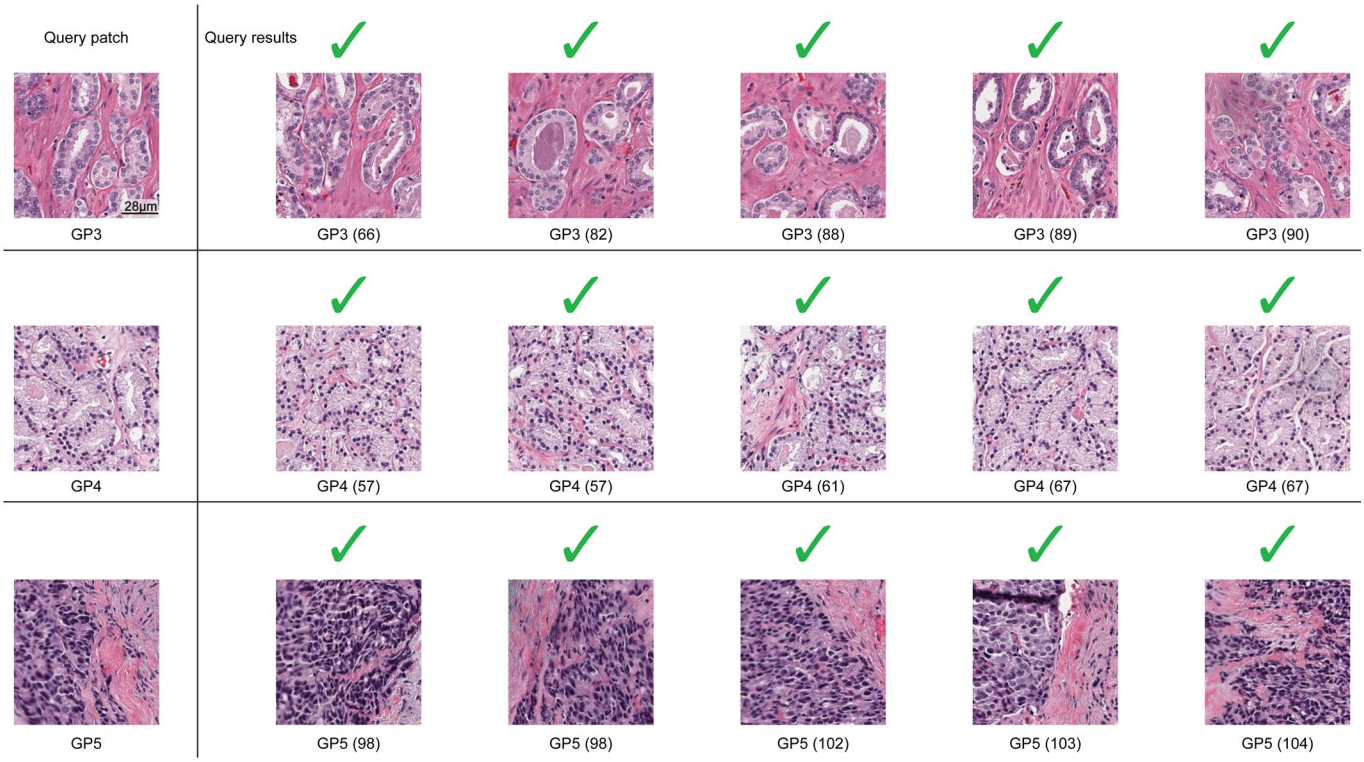


Extended Data Fig. 6 | See next page for caption.

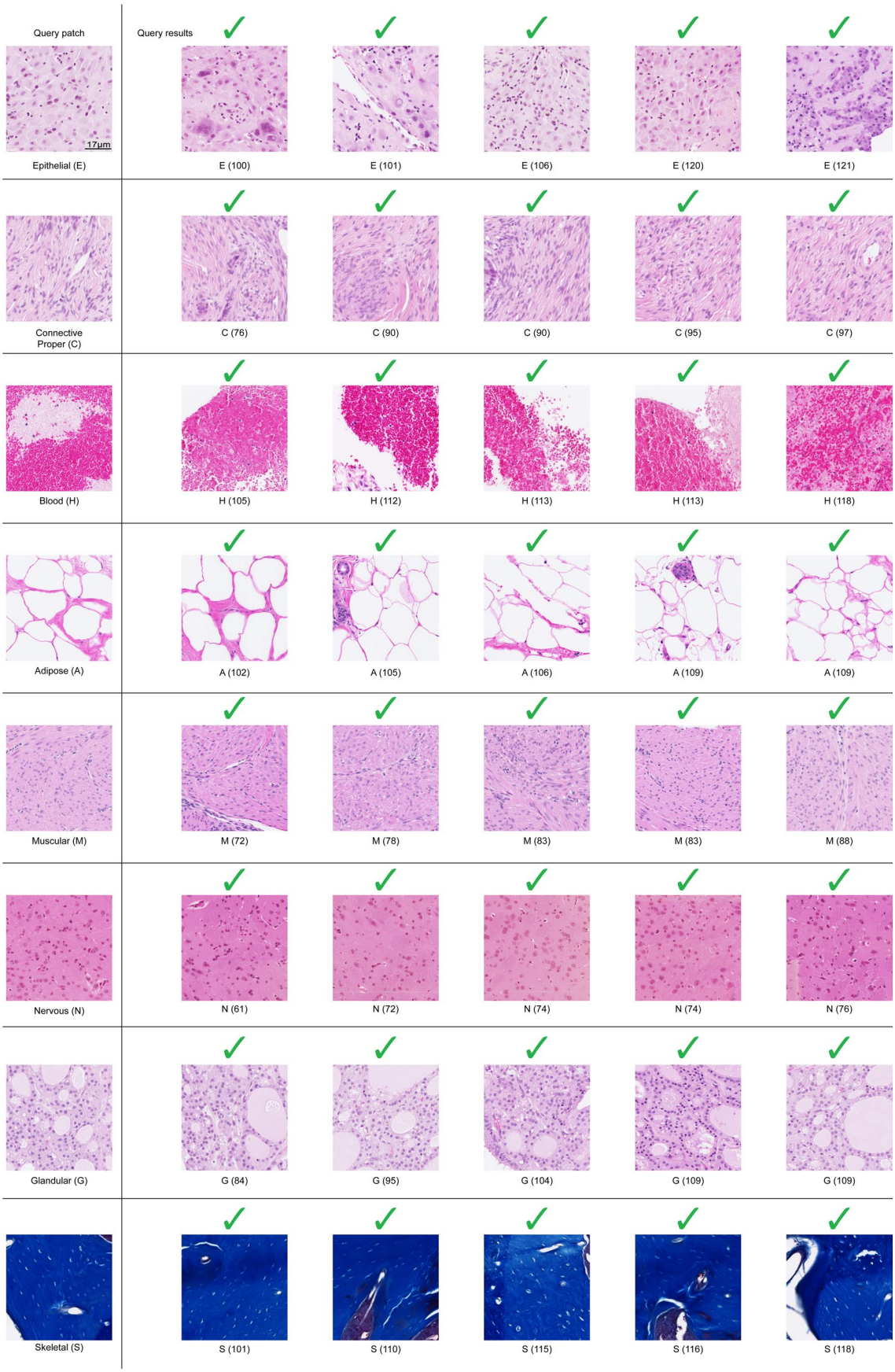
Extended Data Fig. 6 | Examples of patch retrieval on Kather100k colon data. The patches of cancer associated stroma (STR), colorectal adenocarcinoma epithelium (TUM), lymphocytes (LYM), adipose (ADI), debris (DEB), mucus (MUC), muscle (MUS), normal tissue (NORM), and background (BACK) are presented in the figure. The number in parentheses is the Hamming distance between the query patch and each result. All patches share the same scale bar.



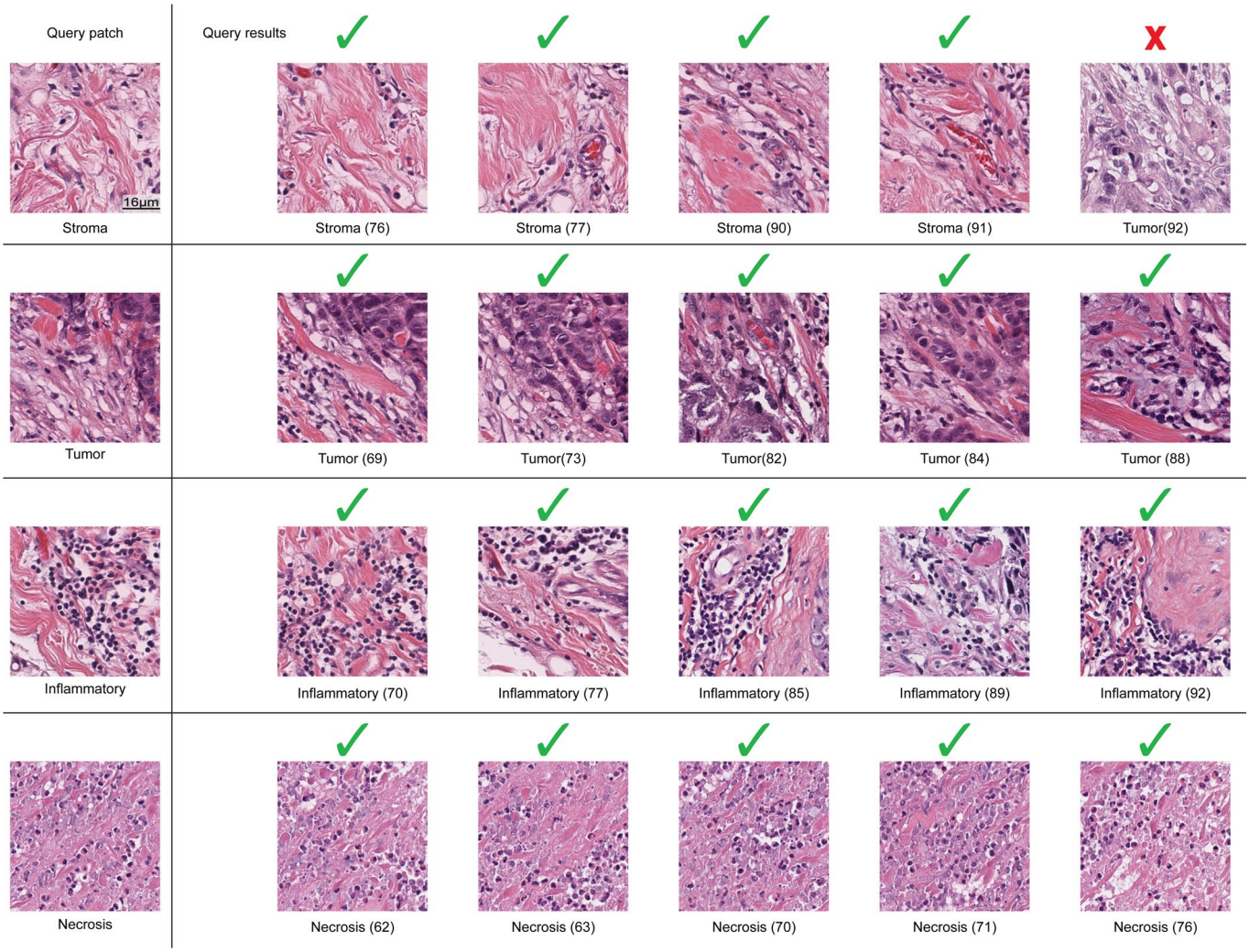
Extended Data Fig. 7 | Examples of patch retrieval on WSSS4LUAD lung data. We considered three types of tissues in WSSS4LUAD data (i.e., Stroma (STR), Tumor (TUM) and Normal (non-neoplastic)). The number in parentheses is the Hamming distance between the query patch and each result. All patches share the same scale bar.



Extended Data Fig. 8 | Examples of patch retrieval on in-house BWH prostate data. The patches of Gleason pattern (GP) 3, 4 and 5 are presented in the figure. The number in parentheses is the Hamming distance between the query patch and each result. All patches share the same scale bar.



Extended Data Fig. 9 | Examples of patch retrieval on Atlas data. The patches of Epithelial (E), Connective Proper (C), Blood (H), Adipose (A), Muscular (M), Nervous (N), Glandular (G) and Skeletal (S) are presented in the figure. The tissue patches are collected from various unknown organs[47]. The number in parentheses is the Hamming distance between the query patch and each result. All patches share the same scale bar.



Extended Data Fig. 10 | Examples of patch retrieval on BCSS breast data. The patches of Stroma (STR), Tumor (TUM), Inflammatory, and Necrosis are presented in the figure. The number in parentheses is the Hamming distance between the query patch and each result. All patches share the same scale bar.

nature portfolio

Corresponding author(s): Faisal Mahmood

Last updated by author(s): Jul 13, 2022

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☒	☐	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☐	☒	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☒	☐	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection	All in-house slides were scanned by Aperio and Hamamatsu S210, and were accessed through openslide (3.4,1). Code for data and image processing was implemented in Python (3.7.0), and is available at https://github.com/mahmoodlab/SISH.
Data analysis	The implementation of the pipeline for model development and evaluation are available from https://github.com/mahmoodlab/SISH. All codes were written in Python (3.7.0) and use Pytorch (1.5.0) for deep learning. These additional Python libraries were used: h5py (2.10.0), matplotlib(3.3.0), numpy (1.19.1), opencv-python (4.3.0.38), pillow (7.2.0), pandas (1.1.0), scikit-learn (0.23.1), seaborn (0.10),scikit-image (0.17.2), torchvision (0.6.0) tensorboard (2.3.0) and tqdm (4.48.0). We used Adobe Illustrator to create pie charts and to arrange all figures.

For manuscripts utilizing custom algorithms or software that are central to the research but not yet described in published literature, software must be made available to editors and reviewers. We strongly encourage code deposition in a community repository (e.g. GitHub). See the Nature Portfolio guidelines for submitting code & software for further information.

Data

Policy information about availability of data

All manuscripts must include a data availability statement. This statement should provide the following information, where applicable:

- Accession codes, unique identifiers, or web links for publicly available datasets
- A description of any restrictions on data availability
- For clinical datasets or third party data, please ensure that the statement adheres to our policy

The TCGA diagnostic whole-slide data and corresponding labels are available from the National Institutes of Health (NIH) genomic data commons (https://portal.gdc.cancer.gov). The CPTAC whole-slide data and the corresponding labels are available from the NIH cancer imaging archive (https://cancerimagingarchive.net/datascope/cptac). Supplementary Table 19 provides access links to publicly available patch-level datasets. All reasonable requests for

academic use of in-house raw and analysed in-house data can be addressed to the corresponding author. All requests will be promptly reviewed to determine whether the request is subject to any intellectual property or patient-confidentiality obligations, will be processed in concordance with institutional and departmental guidelines, and will require a material transfer agreement.

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

☒ Life sciences ☐ Behavioural & social sciences ☐ Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	No statistical tests were used to determine the sample size. We used all available data for model development. Public datasets: After excluding poor-quality slides, we used 11,561 slides from the TCGA and 2,678 slides from the CPTAC. For the kather100k, BCSS and Atlas, we use all data in the released datasets. For WSSS4LUAD, we used the data in the challenge training set with stroma, tumour and normal annotation (see Methods for details). BWH independent test cohort: We used all available whole-slide images from our in-house repository corresponding to rare and common diagnoses. Based on the availability of slides and after excluding poor-quality slides, we used a total of 8,169 in house slides.
Data exclusions	Pre-established exclusion criteria include slides with significant marking covering the tissue area, damaged slides and missing slides. Slides with markings that do not predominantly cover tissue regions were not excluded.
Replication	We provide the data and instructions to reproduce the results at https://github.com/mahmoodlab/SISH. Replication was successful when all requirements were met.
Randomization	No patient recruitment was performed for this study, and randomization was not necessary.
Blinding	Blinding was not needed for this retrospective image-analysis study.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
☒	☐ Antibodies
☒	☐ Eukaryotic cell lines
☒	☐ Palaeontology and archaeology
☒	☐ Animals and other organisms
☐	☒ Human research participants
☒	☐ Clinical data
☒	☐ Dual use research of concern

Methods

n/a	Involved in the study
☒	☐ ChIP-seq
☒	☐ Flow cytometry
☒	☐ MRI-based neuroimaging

Human research participants

Policy information about studies involving human research participants

Population characteristics	Public Data: TCGA,CPTAC, Kather100k, BCSS, WSSS4LUAD and Atlas contain data from a diverse population representing multiple hospitals. In-house data: All patient cases between 2002–2020 were queried from the pathology database.
Recruitment	No patient recruitment was necessary for the use of histology whole-slide images retrospectively.
Ethics oversight	The Mass General Brigham IRB committee approved the retrospective image analysis of pathology slides.

Note that full information on the approval of the study protocol must also be provided in the manuscript.