



Deep learning for cancer cell detection: do we need dedicated models?

Michał Karol¹ · Martin Tabakov¹ · Urszula Markowska-Kaczmar¹ ·
Łukasz Fulawka²

Accepted: 31 December 2023 / Published online: 14 February 2024
© The Author(s) 2024

Abstract

This article proposes a novel concept for a two-step Ki-67/lymphocytes classification cell detection pipeline on Ki-67 stained histopathological slides utilizing commonly available and undedicated, in terms of the medical problem considered, deep learning models. Models used vary in implementation, complexity, and applications, allowing for the use of a dedicated architecture depending on the physician's needs. Moreover, generic models' performance was compared with the problem-dedicated one. Experiments highlight that with relatively small training datasets, commonly used architectures for instance segmentation and object detection are competitive with a dedicated model. To ensure generalization and minimize biased sampling, experiments were performed on data derived from two unrelated histopathology laboratories.

Keywords Histopathology images · Ki-67 index · Cell detection · Machine learning · Deep learning · Virtual pathology

1 Introduction

The Ki-67 index is a commonly used predictive factor for treatment decisions in breast cancer patients. It plays a significant role in the prediction of whether and what scheme of therapy is needed. This fact is due to the role of Ki-67 nuclear protein as a marker of cellular proliferation activity, which is defined by the ratio of Ki-67-positive cells (given

✉ Michał Karol
michal.karol@pwr.edu.pl

Martin Tabakov
martin.tabakow@pwr.edu.pl

Urszula Markowska-Kaczmar
urszula.markowska-kaczmar@pwr.edu.pl

Łukasz Fulawka
lukasz.fulawka@cellgen.pl

¹ Department of Artificial Intelligence, Wrocław University of Science and Technology, Wyb. Wyspińskiego 2, 50-370 Wrocław, Poland

² Molecular Pathology Center Cellgen, Ul. Piwna 13, 50-353 Wrocław, Poland

as a percentage), commonly known as proliferation index (PI) (Inwald et al. 2013; Nielsen et al. 2020). Proliferation Index is also vital in non-breast cancers like prostate cancer (Kammerer-Jacquet et al. 2019) and lung cancer (Wei et al. 2018), making it a common cancer metric.

The expression of Ki-67 nuclear protein is visualized by immunohistochemistry (IHC), which is conducted in each newly diagnosed breast cancer during the routine histopathological examination. As a result of the IHC reaction, the Ki-67-positive nuclei are stained brown with diaminobenzidine (DAB). The remaining nuclei as well as the other tissue elements are colored blue with hematoxylin.

The routinely applied method of PI assessment in diagnostics is eyeballing. However, this approach is prone to significant interobserver variability reaching up to 57%, in one of the previous studies (Fulawka and Halon 2017). The most precise way of PI evaluation is one-by-one nuclei counting, which in practice can only be conducted utilizing software equipped with a pointer function. According to the International Ki-67 in Breast Cancer Working Group recommendations, a minimum of 500 cells should be counted to assess the proliferation index (Nielsen et al. 2020). These recommendations also suggest that the evaluation of PI may be aided by automated scoring using digital image analysis.

The former approaches to solving this problem were mainly based on classic machine learning techniques such as clustering and fuzzy logic, Mungle et al. (2017). Now, it is an unquestioned belief that we owe considerable progress in the computer vision domain and image analysis to developing deep neural networks (DNNs). Using DNN removes the need to select features manually. Instead, the model can learn features from the image, process it, and give output. DNN also demonstrated a potential for many fields in medicine (Esteva et al. 2021; Iqbal et al. 2021; Lu et al. 2022). In this paper, we focus on pathological analysis, which is very subjective in the visual inspection of tissue samples under a microscope. It can cause inconsistencies in diagnostic and prognostic opinions. Therefore, Deep Learning (DL) can support critical medical tasks. We can mention diagnostics, prognostication of outcomes and treatment response, pathology segmentation, and disease monitoring.

Many of the approaches described in the literature are evaluated on one dataset with limited variability. Therefore, it is difficult to determine if the methods would generalize well with new, unseen data.

This observation was the starting point for the research. The primary assumption of the study is the application of well-known deep learning models, for instance, segmentation and object detection, to verify if such a process is robust enough in terms of data dependents. Many research papers report high classification Ki-67 accuracy results, but often they deal with medical data prepared in a specific medical facility, which is available during the research. Despite maintaining the typical, well-known standards of histopathology preparations and acquisition of corresponding microscopic images, it does not solve the problem of inter-laboratory variance. Therefore, if processed in the same way in the same facility, the test data might be considered a biased sample, which can affect the robustness of the proposed classifiers. Therefore, our research focuses on testing the Ki-67 index calculation procedure on an entirely different dataset concerning tissue preparation and image acquisition from the training dataset.

The second goal of the study was to check what Ki-67 proliferation index calculation results can be obtained using generic models for image segmentation or detection compared to dedicated models such as PathoNet.

2 Related work

Deep Neural Networks (DNNs), especially Convolution Neural Networks (CNNs) (LeCun et al. 1998), have a relatively long record in artificial intelligence. CNN's have shown great potential in the field of image processing. Based on these models, we can observe research development in the Ki-67 index automatic assessment. Several CNNs exist. The most common are the following AlexNet (Krizhevsky et al. 2012), GoogleNet (Szegedy et al. 2014), ResNet (He et al. 2015; Li et al. 2022), DenseNet (Huang et al. 2016). They are frequently used in image classification, but segmentation and detection tasks are worth mentioning especially histopathological image recognition. Besides these generic architectures, models dedicated to histopathological image recognition are also considered.

Smith et al. (2020) consider the pipeline construction needed to use Deep Learning (DL) approach to support physiologists' work. The first problem in modeling Whole-slide-images (WSIs) is the scale, which is impossible to process by existing hardware. This discrepancy requires tiling and reshaping the tiles. Usually, regions of interest (ROIs) are annotated by sampling tiles to prepare learning patterns for models. The way of sampling is crucial for the final result. It can be done systematically (grid search) using overlapping tiles (a choice in overlap amount is critical to balance redundancy and computational efficiency) or in random sampling, which sampling is impractical to predict the whole region of interest. The final decision is based on prediction, which can be made on the tile level or slide level (Dimitriou et al. 2019). It means we have to decide whether we are interested in drawing a conclusion based on a small piece of the tissue or the slide as a whole. In each case, we must prepare the training set appropriately and fit the loss function for training a model. In the case of tile-level predictions, even a single positive prediction is essential, but sometimes it may require a broader context (information from other tiles or slides). More typically used measures were the number of detected metastases, mean detection size and standard deviation, mean detection likelihood, and standard deviation. Reasoning based on a whole slide considers it as cancer-positive, while only local regions of that slide are recognized as positive.

Many previous works, for example, Xing et al. (2014) assumed manual selection of hot spots (ROIs) to assess the Ki-67 index. In new research, automatic detection of hot spots is popular. Govind et al. 2020 describe the whole pipeline, which classifies each hot-spot-sized tile in a WSI into one of four classes: background, non-tumor, G1 tumor, and G2 tumor. A Ki-67 index of <3% is grade 1 (G1), between 3 and 20% is grade 2 (G2), and >20% is grade 3 (G3). The developed approach termed Synaptophysin-Ki-67 Index Estimator (SKIE) automatically detects hot spots and calculates the Ki-67 index from those hot spots.

One of the first uses of DNN for Ki-67 stained hotspot detection and proliferation rate scoring is the paper (Saha et al. 2017), where authors used a five-layered convolutional network (CNN). They assessed the obtained results based on their dataset containing 450 samples and compared the results to the traditional approaches.

Niazi et al. (2018) used two CNN networks (AlexNet and Inception Ver 3) trained initially on ImageNet. They retrain Inception v3 via transfer learning to classify 64x64 pixel tiles extracted from Ki-67 stained neuroendocrine tumor biopsies. AlexNet was fine-tuned using the same dataset. It served as a baseline for comparison. Feng et al. (2020) proposed the method based on GoogLeNet Inception V1 (Szegedy et al. 2014) to calculate the Ki67 index automatically. The GoogleNet model was used to locate invasive ductal carcinoma in the form of a box with a heatmap. Then, they use another algorithm to extract the structure,

morphology, color, and other characteristics automatically of positive/negative cells, to train the random forest classifier.

In Liu et al. (2020) a modified ResNet18 convolutional network was applied as a classifier to locate positive or negative cells in large-scale images of H & E (hematoxylin & eosin) stained slides. The authors applied a special transformation method to the trained CNN to convert fully connected layers into convolutional layers. Thanks to this procedure, the classification of Region of Interests (ROIs) sized (7, 556x3, 864) was possible.

In the context of this survey, the U-Net architecture is also worth mentioning. Initially, the U-Net architecture was designed for segmenting biomedical images. The model used a per-pixel classification approach (semantic segmentation) to segment cells, followed by a watershed to separate overlapping cells in the prediction. The U-NET is also based on a convolutional neural network (CNN) with an encoding part that compresses the images into a compact representation and a decoding part that reconstructs the prediction.

Negahbani et al. in Negahbani et al. (2021) describe the whole pipeline to assess the Ki67 index. They also present PathoNet architecture designed explicitly for Ki-67 proliferation index calculation. First, the PathoNet model estimates a density map for each class. Then, binary images are produced based on these maps. Next, region centers with low and borders with high values are scored using inverse distance transformation. Finally, the watershed algorithm predicts cell center coordinates. The PathoNet model is based on the U-Net-like (Ronneberger et al. 2015a) backbone, where convolutional layers are replaced by a new inception module called residual dilated inception module (RDIM). The PathoNet network produces the density map of Ki-67 immunopositive, immunonegative, or lymphocyte class. The authors collected their SHIDC-B-Ki-67 dataset (from Shiraz Histopathological Imaging Data Center) that served as the evaluation of the proposed method.

The KiNet model proposed by Xing et al. (2019) is also based on the U-Net model with residual connections for nucleus recognition. To handle scale variation of nuclei, they apply a multi-context aggregation to the combination of multi-level features. The new element is also ROI prediction (with weak supervision), which assists the nucleus identification task.

U-Net is also the central part of piNET (Geread et al. 2021). Instead of the watershed algorithm, this model uses a Gaussian-defined proximity map as ground truths for individual nuclei and a regression-based loss function on a per-pixel basis to identify central regions of the tumor nuclei. In the case of this research, the method was evaluated using five different datasets. The proliferation index was divided into three ranges (Low<25, Medium 25–75, High>75). The model predicted the class of the image. The proliferation index accuracy was from 80% to 88.3%, depending on the dataset.

Li et al. (2021) proposed another model for Ki-67 index predictions in gliomas in a non-invasive way. The authors also named it KiNet. It differs from the one described by Xing et al. (2019). The model uses multimodal information. It contains two independent auxiliary branches and one main branch. Each branch is responsible for four feature extraction stages, and each stage is composed of several blocks. The blocks have structures similar to the ResNet block. The whole network performs binary classification (Ki-67 indexes less than 10% were labeled as 0, and the rest are called high Ki-67 indexes labeled as 1).

A deep CNN ensemble model for the Nottingham histological grade (NHG) into three classes is proposed in Wang et al. (2022). The ensemble is composed of 20 CNN models.

Considering Ki-67 proliferation index calculation, deep models can solve various tasks. As the first one, we can mention binary classification (Li et al. 2021) (one class assigns the Ki-67 index less than 10% and the second contains index values higher than 10%). Multiclass classification is considered for instance in Geread et al. (2021) where ranges are

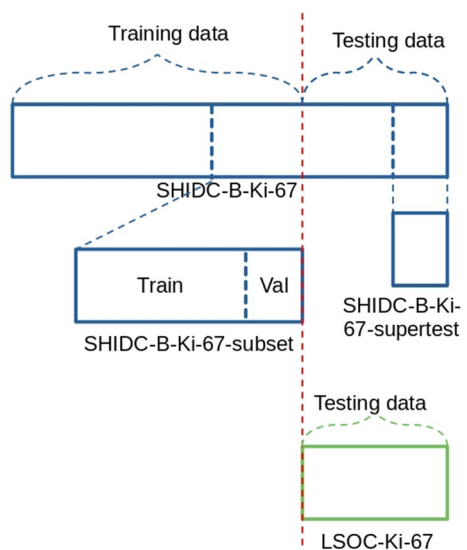
defined as Low<25, Medium 25–75, High>75. The research focused on individual nucleus recognition, first needs nuclei identification and classification. In such a case, the index can be calculated manually, or other algorithms are used. Sometimes before index calculation, the task first is to segment all immunopositive nuclei or detect them by surrounding these nuclei with bounding boxes. The index calculation is based on another algorithm.

Summing up this survey, it is worth citing the authors of Srinidhi et al. (2021) who noticed that there is no generic rule about the choice of architectures with the type of disease prediction task. This observation also gave motivation to our research.

3 Materials and methods

We prepared a special experimental procedure to test both of the problems. First, color correction and padding were applied to the images. Next, deep neural networks that enable automatic detection, classification, and counting of cells on images were used. All these steps are presented in detail in Sect. 3.4. As a dedicated model for Ki-67 calculation, the PathoNet was used in two variants—with base weights and retrained. As for generic models, Faster R-CNN, SSD, RetinaNet, YOLO ver4, and Mask R-CNN were picked. The need for an additional lymphocyte classifier emerged after initial experiments due to the high error rate in classifying lymphocyte and Ki-67 negative cells. Therefore, to improve results, we developed a new lymphocyte classifier. Experiments were performed using two datasets: a subset of the SHIDC-B-Ki-67 dataset (Negahbani et al. 2021) and LSOC-Ki-67 (Fulawka et al. 2022) acquired in a different laboratory. The SHIDC-B-Ki-67 dataset was split into a training dataset and a validation dataset. Generic networks need the appropriate image annotation (bounding boxes and segmented cells). Therefore, before experiments, a subset of the SHIDC-B-Ki-67 dataset was taken and annotated appropriately to the needs of the networks. Next, this subset was divided into training and validation datasets. To test the results, the original SHIDC-B-Ki-67 test set was used as well as the LSOC-Ki-67 dataset. Figure 1 shows the general overview of the dataset structure and names used.

Fig. 1 Overview of the datasets used in the experiments



3.1 Datasets

In this section, both datasets used will be described alongside the preprocessing steps needed for experiments.

3.1.1 SHIDC-B-Ki-67

The SHIDC-B-Ki-67 dataset contains microscopic biopsy images of invasive ductal carcinoma type of breast cancer. The entire SHIDC-B-Ki-67 dataset contains 1656 training and 701 test data. The histopathological images were derived from Ki-67-IHC slides prepared from tru-cut biopsies. Stained images were annotated by expert pathologists concerning three categories: Ki-67 positive tumor cells, Ki-67 negative tumor cells, and tumor cells and tumor-infiltrating lymphocytes (Fig. 2: *Data Stage*). Each image contains 69 cells on average, and there are 162,998 cells. All further details of the SHIDC-B-Ki-67 dataset preparation procedure can be found in Negahbani et al. (2021).

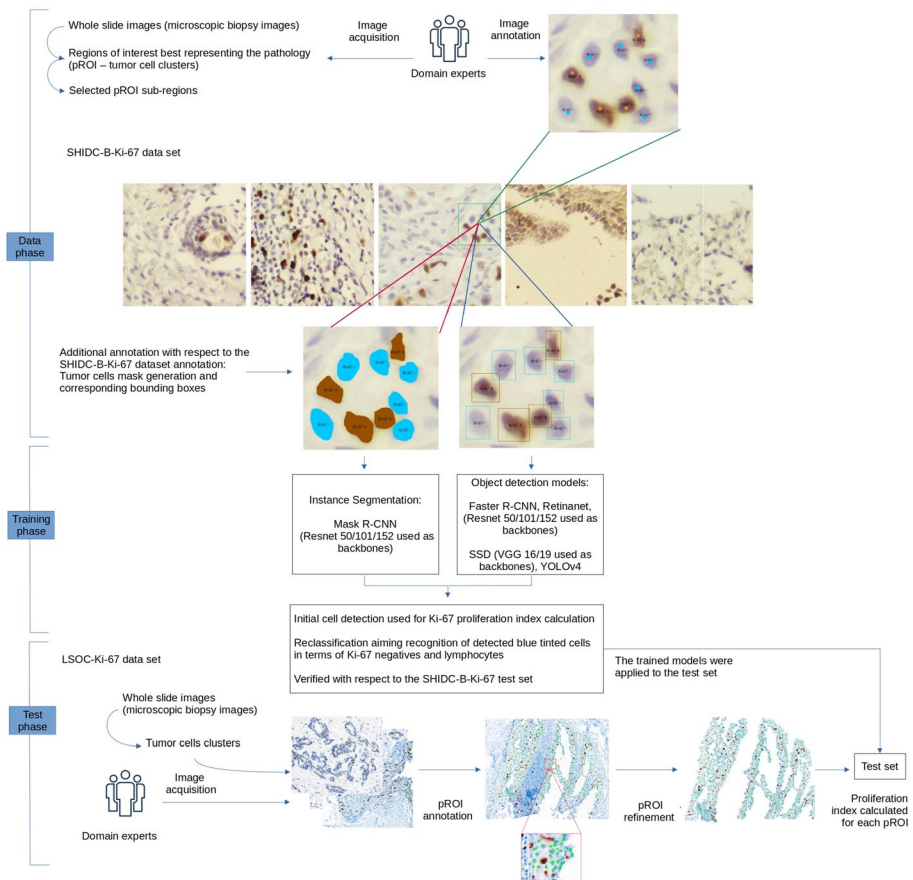


Fig. 2 The pipeline used in the research

In the raw SHIDC-B-Ki-67 dataset, only the cell center points are assigned. To conduct the study, cell masks and bounding boxes were added but only in the training dataset. The rest (validation and test data sets) remained unchanged. And from SHIDC-B-Ki-67 train set, a subset of 94 images, was extracted. The training dataset was annotated by the annotation team with masks and bounding boxes based on experts' center point annotations to prepare correct labeling for the task of segmentation and object detection. Thus allowed us for training such models (Fig. 2: *Data Stage*). The resulting annotated dataset contains 6884 masks and bounding boxes. Images were then split into 512x512 pixel tiles, and the resulting dataset is further referred to as SHIDC-B-Ki-67-subset. This dataset was split in an 80:20 ratio to training and validation images (623 tiles for training and 145 tiles for validation).

After running the first round of experiments, it was decided to create a unique SHIDC-B-Ki-67. This subset, later called the SHIDC-B-Ki-67-supertest dataset, contains 50 random images from the SHIDC-B-Ki-67 test dataset. Images included in this super-test dataset were annotated by the expert pathologist. Annotations were made based only on the raw image data, without any preliminary annotations marked on the image. The new test dataset preparation was motivated by measuring the impact of the false positive errors due to missing annotation and the overall effect on the result.

3.1.2 LSOC-Ki-67 dataset description

The LSOC-Ki-67 (Lower Silesian Oncology Center) dataset applied as the test set in the research consists of 95 whole slide images (WSI) of ductal carcinoma in situ (DCIS). The diagnostic tissue material was obtained by minimally invasive vacuum-assisted percutaneous breast biopsy in the Breast Unit, Lower Silesian Oncology Center (Wrocław, Poland), in the period 2003–2011. The biopsied tissue was processed with standard diagnostic procedures applicable in the Department of Pathology. The tissue slides for the current project, retrieved from archival FFPEs, were subjected to an immunohistochemical (IHC) reaction against the human Ki-67 antigen. The 4 μm -thick paraffin sections were cut onto SuperFrost Slides (Thermo Fisher Scientific, Gerhard Menzel GmbH, Braunschweig, Germany). The rabbit monoclonal ready-to-use (2 $\mu\text{g}/\text{mL}$) antibody (Clone 30-9) and ultra-View Universal DAB Detection Kit (Ventana, Tucson, AZ, USA) were applied. The procedure was performed automatically using Benchmark XT (Ventana, Tucson, AZ, USA), according to the producer's manual.

The process was conducted following local guidelines and regulations. The study was approved by the local ethics committee (Bioethics Committee Wrocław Medical University, Poland). The use of the archival material has been approved by the Director of the Lower Silesian Oncology Center. Additionally, archival hematoxylin-eosin slides for each case were retrieved to be assessed as a 'second look' by the two board-certified pathologists experienced in breast cancer. The IHC slides were scanned using the two-slide scanners: ScanScope CS (Aperio ePathology, Leica Biosystems Imaging, Vista, CA, USA) and Panoramic 250 (3DHitech Ltd., Budapest, Hungary). A board-certified pathologist marked hot spots (areas of elevated Ki-67 protein expression). One hot spot was selected for each WSI and defined as a pathological region of interest (pROI) (Fig. 2: level test phase). Following selection criteria for pROI were used: representativeness in morphology, the highest image sharpness, possible lack of artifacts, absence of unspecified cytoplasmic staining pattern and extracellular reaction, and possible avoidance of tissue edges. Of course, it does not mean that selected pROIs may not contain such areas. The selection

procedure aimed to choose regions with a high number of tumor cells for further annotation by the domain expert. The size of pROIs varies from 736×1216 pixels (0.895 Mpix) to 9488×4832 pixels (45.846 Mpix), with an average size of 9.705 Mpix. The pROIs images were saved as RGB tiff/png files with lossless compression. Each Ki-67 slide comprises 2 stains: diaminobenzidine (DAB) for positive nuclei and hematoxylin for all nuclei and background. It is worth underlying that the staining is not limited to cancer cell nuclei only, since cancer tissue by itself is composed of other elements, such as fibroblasts, collagen fibers, immune cells (overwhelmingly lymphocytes), and blood vessels. The annotation process was done by a domain expert using the ImageJ software with the Cell Counter plug-in (Schneider et al. 2012); Vos (Retrieved on 2022-07-12) (Rueden et al. 2017). The detailed methodology is described in Fulawka and Halon (2016). Diaminobenzidine (DAB) and hematoxylin-stained neoplastic nuclei were annotated in the area of pROIs (Fig. 2). Complete agreement of reviewing expert was required to include annotation from annotating expert in dataset.

Following the recommendations of the International Ki-67 in Breast Cancer Working Group 6, the nuclei were considered Ki-67-positive despite the staining intensity. Artificially changed nuclei were not marked. A total number of 150,592 nuclei were selected, including 26,093 DAB-stained ones. On average, 1585 (median 1220) nuclei per image were annotated. The annotations were revised and accepted by the second board-certified pathologist. The ground truth (gold standard) is the labeling index, i.e. proliferation index (PI), calculated by dividing the number of DAB-stained neoplastic nuclei (Ki-67-positive) by the total number of neoplastic nuclei (the sum of DAB and hematoxylin stained nuclei) (Eq. 1).

$$PI = \frac{N_{Ki-67(+)}}{N_{Ki-67(-)} + N_{Ki-67(+)}} \quad (1)$$

This data set was used and described in detail in Fulawka et al. (2022). Neoplastic nuclei were annotated in the pROIs.

To test models, additional refinement of the selected pROI was utilized. As models of instance segmentation or object detection target all viable cells, the objective comparison with other cell recognition procedures is hard to be applied. The reason is that many authors present test results considering only annotated image ROIs. This implies that if a cell is recognized correctly but is not located in the annotated ROI, wrongly would be recognized as a false positive. Considering the above, the LSOC-Ki-67 histopathology images were limited in the area of annotated ROIs for comparative analyses.

The general pipeline of the research experiment is shown in Fig. 2. It consists of the following steps:

- preparation of the annotated SHIDC-B-Ki-67-subset
- training models on the training set of the SHIDC-B-Ki-67-subset and validation and early-stopping based on a validation subset of SHIDC-B-Ki-67-subset.
- application of the SHIDC-B-Ki-67 original test set for base model comparison,
- application of the LSOC-Ki-67 dataset as a test set for final model comparison.

3.1.3 Dataset preprocessing

A custom pipeline for the SHIDC-B-Ki-67-subset was developed to achieve the best results from such a limited number of data. The research pipeline consists of:

1. *Color correction by white balancing the images* This operation allowed us to work on images with better visualization possibilities, and white balancing allowed us to run models on other white-balanced datasets (like LSOC-Ki-67). White balancing was performed by clipping the top and the bottom 5% of pixel values for each of the channels, and then scaling the remaining 90% of values to the full pixel value range(0–255).
2. *Padding and tiling with a tile size of 512x512 pixels and overlap of 10%* Such a small tile size is due to GPU memory limitations. Using smaller tile sizes made it possible to train models without the need to downscale the images. To use as much as possible out of training data in the SHIDC-B-Ki-67-subset, 10% overlap was used. That was training cells that would occur on tile-splitting lines and would have to be discarded and could be used in research.
3. *Transformations using the following operators: vertical flip, horizontal flip, hue/brightness/saturation change, and Gaussian blur* Operators are randomly selected, where the probability of choosing a single operator is 50%, and operator choice probabilities are independent of each other.

The resulting images and annotations are saved in COCO dataset format for later use. Due to specific needs, the dataset for YOLOv4 and PathoNet had to be transformed from COCO-style JSON format annotations to architecture-specific text files.

3.2 Deep neural networks models

In this subsection, all models applied in the research will be shortly described. Mostly generic models intended for object detection will be mentioned, but there is also one model for instance segmentation. At the end of the section, a model explicitly designed for proliferation index calculation will be outlined.

3.2.1 Faster R-CNN

faster R-CNN (Ren et al. 2016) was developed to improve the detection results of Fast R-CNN's early architecture (Girshick 2015). Adding a fully convolutional region proposal network made it possible to obtain possible ROI with the probability score quickly. Using the Non-Max-Suppression algorithm (NMS) (Neubeck and Van Gool 2006) causes a significant proposal reduction, resulting in a faster network overall speed. The network became the state-of-the-art solution for object detection after surpassing other solutions like MultiBox.

3.2.2 Mask R-CNN

Mask R-CNN (He et al. 2017) is an extended and improved version of Faster R-CNN with a particular parallel branch dedicated to segmentation mask generation. With this simple addition of a fully convolutional mask prediction branch, extraordinary results were achieved for object segmentation, surpassing state-of-the-art solutions. Architecture enhancement of detection ROI alignment step improved bounding box detections alongside mask predictions.

3.2.3 SSD

Single Shot MultiBox Detector (SSD) (Liu et al. 2016) is one of the earliest general-purpose one-shot detection architectures, beating state-of-the-art Faster R-CNN. By changing completely how detections and bounding boxes are calculated in contrast to the R-CNN architectures, already pretrained parts of VGG/ResNet convolutional neural networks can be used as feature maps. Multiscale object detection with built-in NMS simplifies working with the architecture, as different scale detections are cleared and not overlapping. SSD was and still is an inspiration for more complex object detection architectures.

3.2.4 RetinaNet

RetinaNet (Lin et al. 2017) is one of the best one-shot object detection architectures. Using a feature pyramid network (FPN), class+box subnetworks with a highly specialized loss function called focal loss instead of cross-entropy loss allowed for improvements in mAP scores. Focal loss favors the hard-to-learn detection cases in the presence of an overwhelming number of easy background cases.

3.2.5 YOLOv4

YOLOv4 (Bochkovskiy et al. 2020) further improves the already excellent YOLOv3 (Redmon and Farhadi 2018) architecture based on CSPDarkNet53 (Wang et al. 2019). YOLO architectures are specifically designed for maximum performance for live object detection. By using simple but effective changes like mosaic data annotation, actively randomizing training image size, switching variants of IoU loss, and adding a learning rate scheduler, the authors improved the resulting mAP. But except for those simple changes, other specific actions were taken, e.g., changing layer activation to Mish activation function, cross-stage partial connections, and many others. The resulting architecture traded off slight performance loss for huge mAP improvements in object detection tasks.

3.2.6 PathoNet

PathoNet (Negahbani et al. 2021) is currently one of the best choices for automatic proliferation index calculation. The solution uses well-established U-Net (Ronneberger et al. 2015b) architecture as a backbone. It also takes advantage of the watershed algorithm, giving precise center point detections as a final result. Furthermore, the open dataset used by the authors of the PathoNet allows us to apply it for comparison. In contrast to general-purpose object detection models, it is tailored to the proliferation index calculation task, increasing its quality and performance.

3.3 Lymphocyte classification

One of the obstacles that emerged during the experiments was the issue of lymphocytes looking like Ki-67-negative cells. To accurately calculate the proliferation index, better differentiation of the lymphocytes from the Ki-67-negative cells is needed. This section describes two approaches to handle higher lymphocyte class presence in test datasets than in train datasets. The first one relies on binary classification (lymphocytes vs. the Ki-67-negative cells). The second one performs neighborhood analysis.

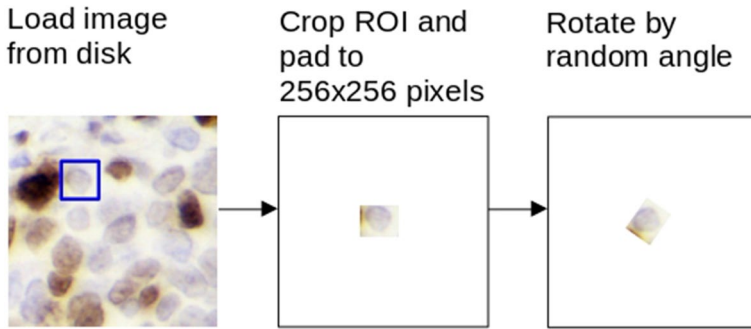


Fig. 3 Pipeline for reclassification model dataset

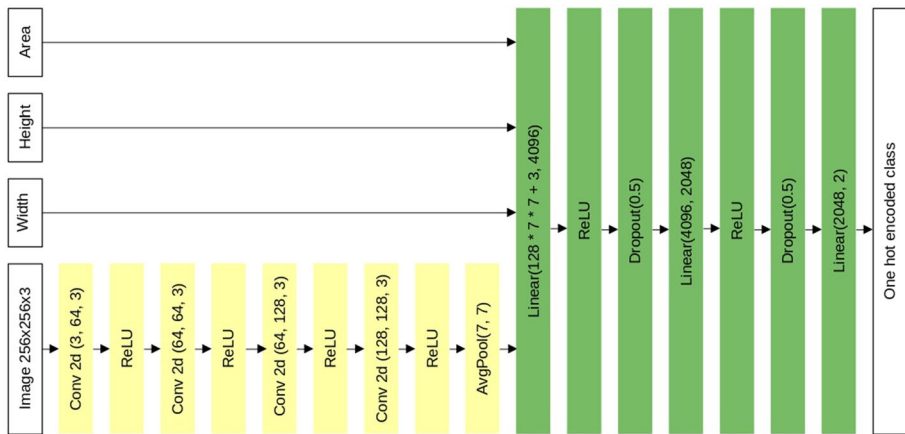


Fig. 4 Architecture of reclassification network

3.3.1 Reclassification–binary classification

During model training and validation, lymphocytes were often improperly classified as Ki-67-negative cells. The main issue is that lymphocytes in the SHDC-B-Ki-67-subset dataset are underrepresented, and models struggle to distinguish between those two classes. To improve results quality, the post-processing step reclassifying Ki-67 negative detections using a small convolutional neural network was added.

ROI areas of Ki-67-negative cells and lymphocytes were extracted from the SHDC-B-Ki-67-subset dataset by cropping an image to the cell bounding box and padding it up to the 256×256 pixel input image to train the reclassifier model. Lymphocyte samples were cloned to balance the classes in the resulting dataset. It could have posed a threat of overfitting on specific images, so data augmentation in image rotation by random angle was used. All steps are depicted in Fig. 3.

Initially, the authors' efforts focused on training ResNet50 or ResNet101 architectures to do this reclassification task, but soon it was clear that using deep architecture was an excessive move. Results obtained from training ResNet50 and ResNet101 were of low quality (F1 below 0,85). Finally, the small multimodal convolution architecture shown in Fig. 4

was chosen, and as a result, finally, F1 scores were acceptable (F1 over 0,95). All models where lymphocytes were reclassified are marked with 'lym-reclass' postfix.

Algorithm 1 Lymphocyte reclassification based on the neighborhood analysis

```

Input : detections
Output: reclassifiedDetections
1 PASSES  $\leftarrow$  30
2 tree  $\leftarrow$  KDTree(GetCenterPoints(detections))
3 numberOfPoints  $\leftarrow$  min(10, detections.len)
4 kiNegativeDetections  $\leftarrow$  GetKiNegativeDetections(detections)
5 neighbourhood  $\leftarrow$  tree.get(kiNegativeDetections, numberOfPoints)
6 for iteration in range(PASSES) do
7     oldNuberOfLymphocytes  $\leftarrow$ 
        GetLymphocyteDetections(detections).len
8     for kiNegativeDetectionIndex in range(kiNegativeDetections.len) do
9         detectionNeighbourhood  $\leftarrow$ 
            neighbourhood[kiNegativeDetectionIndex]
10        distances  $\leftarrow$  GetDistances(detectionNeighbourhood)
11        kiNegativeneighbors  $\leftarrow$ 
            GetKiNegativeDetections(detectionNeighbourhood)
12        lymphocytenighbors  $\leftarrow$ 
            GetLymphocyteDetections(detectionNeighbourhood)
13        kiNegativeScore  $\leftarrow$  Sum(1/distances[kiNegativeneighbors])
14        lymphocyteScore  $\leftarrow$  Sum(1/distances[lymphocytenighbors])
15        if lymphocyteScore > kiNegativeScore then
16            detectionIndex  $\leftarrow$ 
                mapKiNegativeIndex(kiNegativeDetectionIndex)
17            detections[detectionIndex].cls  $\leftarrow$  LYM
18        newNumberOfLymphocytes  $\leftarrow$ 
            GetLymphocyteDetections(detections).len
19        if oldNuberOfLymphocytes = newNumberOfLymphocytes then
20            break
21 reclassifiedDetections  $\leftarrow$  detections

```

3.3.2 Neighborhood analysis

The second approach adds the neighborhood analysis step just before the ground truth point matching step to improve proper lymphocyte classification. The whole procedure, described by Algorithm 1, is to find Ki-67 negatives, which are surrounded by lymphocytes with a high probability of improperly classified lymphocytes, as suggested by an expert. At first, centers of all detections are calculated (line 2). The points selected for analysis are calculated as the minimum value from the set composed of the number 10,

which is the threshold, and the number of detections to handle cases when the number of detections is below the 10 point threshold. (line 3). Next, detections are filtered to get only Ki-67 negative detections (line 4) and the neighborhood is queried for those detections (line 5).

Next, the algorithm is run multiple times (line 6) to converge to the best solution. In the single algorithm step, Ki-67 negative detections are iterated over (line 8), and for each detection, the neighborhood is defined for it (line 9) as well as the distance between centers of detection, and neighborhood detection is calculated (line 10). For the detection, neighborhood indexes of Ki-67 negative neighbors (line 11) and lymphocyte neighbors (line 12) are calculated that are used to calculate scores for both Ki-67 negative and lymphocyte classes (lines 13,14). The score is calculated as $\frac{1}{\text{distance_between_centers}}$. In case of a score of the lymphocyte class is higher than for Ki-67 negative class (line 15), true index of the detection is calculated (line 16) and class is changed to lymphocyte (line 17). After going through all Ki-67 negative detections, it is assessed (line 19) whether the number of lymphocytes changed based on the values calculated at the beginning and the end of the step (lines 7 and 18 respectively). When the number of lymphocytes on the image has stabilized, the algorithm will finish (line 20). Models where neighborhood analysis was performed are marked with 'neigh' postfix.

3.4 Experimental procedure

This section describes in detail all steps of the experimental procedure that is visualized in Fig. 5.

3.4.1 Image preprocessing

The baseline for comparison was a pretrained model PathoNet shared by authors on their GitHub repository. PathoNet detection thresholds as stated in the article's (Negahbani et al. 2021) supplementary materials were used. In the case of the pretrained PathoNet model for SHIDC-B-Ki-67, padding and tiling were omitted, as test image sizes are the same as the model's image input size. For the LSOC-Ki-67 dataset, images were padded to fit 1228x1228 pixel tiles with no overlap and tiled. For pretrained PathoNet LSOC-Ki-67 dataset was color corrected, by applying reverse process to white balancing. This means that normally white-balanced images match SHIDC-B-Ki-67 train dataset channel mean values. The resulting point detections, in order to fit the bounding box output of other models, were transformed to 60x60 pixel artificial bounding boxes with the detection in the center.

For architectures other than PathoNet, test dataset images were padded with white background, white-balanced and split into 512×512 pixel tiles with 10% overlap, as in the training transformation pipeline.

Additionally, PathoNet on 512×512 pixel tiles from the SHIDC-B-Ki-67-subset dataset was trained. The primary reason for training such a model was to compare the quality of PathoNet architecture when the dataset is substantially reduced and to use it as a common ground between generic models and PathoNet. Such a model is later referenced as 'PathoNet-trained'. Similarly, as in pretrained PathoNet, artificial bounding box detection was used to match the output of other models.

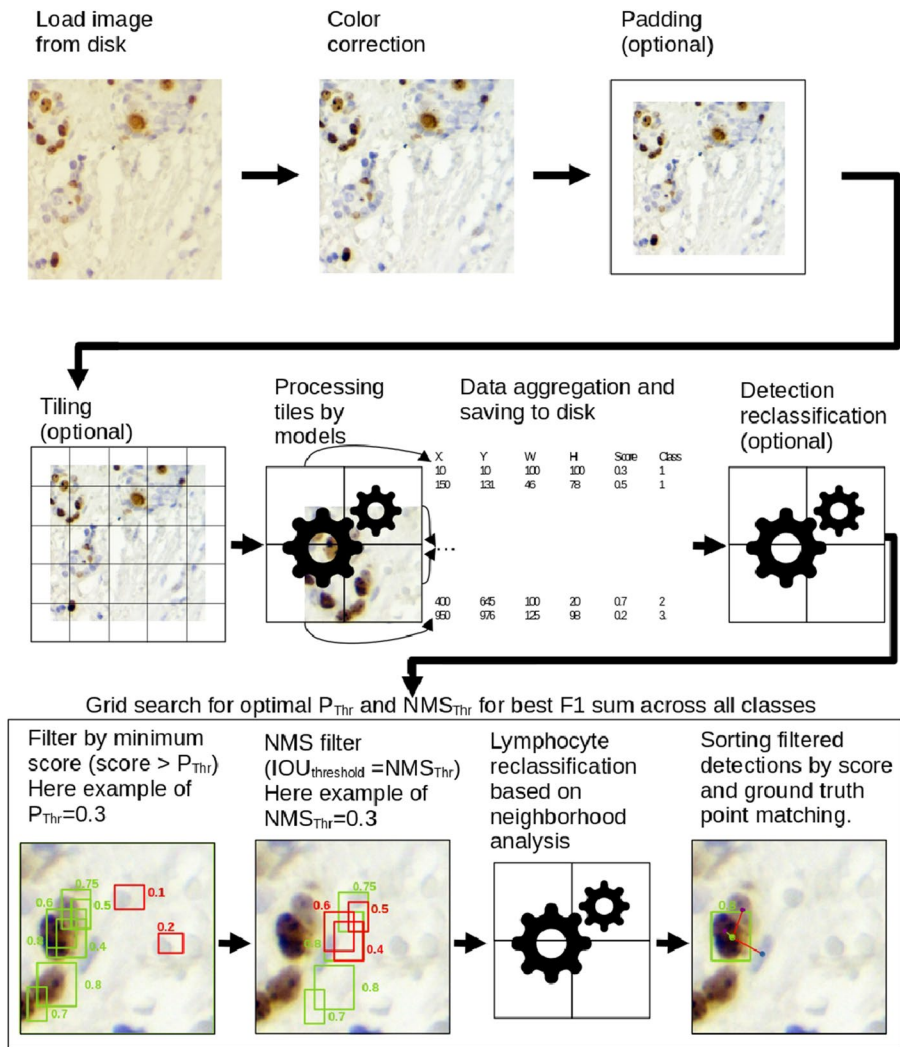


Fig. 5 Experimental procedure for both SHIDC-B-Ki-67 and LSOC-Ki-67 datasets. In step "Data aggregation and saving to disk" X, Y are coordinates of the top-left point of the detection; W, and H are the width and height of a detection bounding box, and the score shows how sure models are with the detection and its recognized class

3.4.2 Training models

This study uses implementations provided by library *torchvision* (Paszke et al. 2019) developed by PyTorch Contributors, YOLOv4 implementation sourced from Alexey Bochkovskiy GitHub repository, and PathoNet implementation provided by the authors of the PathoNet article. Architectures used in this article were state-of-the-art for generic object detection or segmentation and used as a base for further developments in the field. Such flexibility is perfect for medical image analysis, as the variety in the color of the

background field could potentially result in lower detection quality. Training started by using transfer learning out of the models pretrained on the COCO dataset, and later training was conducted on the SHIDC-B-Ki-67-subset training set. Early stopping was used for PyTorch models based on the value of mAP to avoid overfitting models to data. The value of mAP was calculated on the SHIDC-B-Ki-67-subset validation set. From this step on, SHIDC-B-Ki-67-subset is no longer used, and only test datasets: SHIDC-B-Ki-67 and LSOC-Ki-67 will be used.

Hyperparameters The main goal was to show the flexibility of generic models, and as so, the experiment hyperparameter policy uses default implementation values from *lightning_bolts* library. For generic models implemented in Pytorch, the default learning rate value of 0.0001 and SGD (Robbins and Monro 1951) optimizer with a momentum of 0.9 and weight decay of 0.005 were chosen as in default implementations in *lightning_bolts*. For YOLOv4 architecture, we made the appropriate adjustments to the default configuration to accommodate the change in the number of classes. The learning rate (0.001) and SGD parameters (momentum: 0.949, decay: 0.0005) were kept as default. For PathoNet, the selected optimizer was Adam (Kingma and Ba 2017), and thresholds were adjusted to [60, 55, 40] as indicated in Appendix A to the PathoNet Article (Negahbani et al. 2021). Table 1 summarizes the values of the assumed hyperparameters for all models.

3.4.3 Result extraction

The procedure shown in Fig. 5 was used to produce experimental results for all models. First, all steps are performed—loading, color correction, padding, and tiling. The models then process the resulting images, and the detections are aggregated and grouped by the input image. Detections are in the form of confidence score and detected class as well as bounding box described as coordinates of the upper-left point (X, Y on the image) and width and height (W and H, respectively).

3.4.4 Result post-processing and lymphocyte reclassification

Before the final step, it is applied one more post-processing step—detection reclassification. This step is only used for PyTorch models and YOLO as those models have correct information about detection bounding boxes. In the case of PathoNet, bounding boxes are generated artificially. During this step, the previously mentioned lymphocyte classifier is used and only detections of Ki-67-negative cells are processed again, so class labels are adjusted.

As a final step for all architectures, we performed a grid search on SHIDC-B-Ki-67-train dataset for the best F1 score over post-processing parameters: P_{Thr} for minimum score filtering and IOU_{Thr} for NMS. Selected post-processing parameters P_{Thr} and IOU_{Thr} will be utilized for post-processing on SHIDC-B-Ki-67-test, SHIDC-B-Ki-67-supertest and LSOC-Ki-67 datasets. The reasoning behind performing grid search is that each architecture assigns unique confidence values, and models use different strategies to draw bounding boxes. Therefore, no universal post-processing parameters can be chosen. Parameter P_{Thr} was searched in range 0.0 to 0.9 with step 0.1, and IOU_{Thr} was searched in range 0.1–1.0 with step 0.1. and the whole parameters space was checked to find the best set of parameters. P_{Thr} is a parameter of the minimum detection score threshold, and during minimum score filtering, all detections that model scored below P_{Thr} are filtered out. This way, detections that are most probably false positives are removed. IOU_{Thr} is the parameter of the NMS algorithm that indicates the minimum IOU value of two boxes to be considered

Table 1 Hyperparameter values for all models

Model	Library	Optimizer	Batch size	Loss function	Image size
Faster R-CNN	Pytorch	SGD(lr=0.0001, momentum=0.9, weight_decay=0.005)	2	Combination of classification and localization loss	512
Mask R-CNN	Pytorch	SGD(lr=0.0001, momentum=0.9, weight_decay=0.005)	2	Combination of classification, localization and segmentation mask loss	512
SSD	Pytorch	SGD(lr=0.0001, momentum=0.9, weight_decay=0.005)	2	Combination of localization loss and confidence loss	512
RetinaNet	Pytorch	SGD(lr=0.0001, momentum=0.9, weight_decay=0.005)	2	Focal loss	512
YOLOv4	Darknet	SGD(lr=0.001, momentum=0.949, decay=0.0005)	64	YOLO loss function	512
PathoNet	Tensorflow	ADAM(lr=0.0001, betas=(0.9, 0.999)) with learning rate decrease of 0.1 per 10 epochs	16	MSE loss function	1228 resized to 256 in base version, 512 resized to 256 for PathoNet-trained

for the detection of the same object. With this parameter, NMS identifies groups of detections covering the same object (based on the IOU value and IOU_{Thr}) and keeps only the detection with the best score.

After filtering out low-quality detections, an additional step of lymphocyte reclassification is performed to find Ki-67 negative detections surrounded by the lymphocytes. This simple neighborhood analysis improves the pipeline's ability to work with images containing lymphocyte clusters and correctly classify them.

The post-processing pipeline for an image consists of:

1. Filtering by the minimum score—detection score must be over the P_{Thr} .
2. NMS filter with IOU threshold equal to IOU_{Thr}
3. Lymphocyte reclassification based on neighborhood analysis
4. Sorting filtered detections by score and detection to ground truth point match.

The last step of detection to ground truth match was performed by following algorithm Algorithm 2.

Algorithm 2 Matching detections with ground truth points

Input : detections, groundTruthPoints
Output: detectionToGroundTruthPointsMatch

```

1 sortedDetections ← SortByScore(detections)
2 tree ← KDTree(groundTruthPoints)
3 results ← List()
4 usedGroundTruthPoints ← Set()
5 numberOfPoints ← Min(3, groundTruthPoints.len)
6 for detection in sortedDetections do
7   detectionCenterPoint ← GetCenterPoint(detection)
8   topGroundTruthPoints ←
9     tree.get(detectionCenterPoint, numberOfPoints)
10  topNotUsedGroundTruthPoints ←
11    (topGroundTruthPoints – usedGroundTruthPoints)
12  for groundTruthPoint in topNotUsedGroundTruthPoints do
13    if IsPointInsideDetection(groundTruthPoint, detection) then
14      usedGroundTruthPoints ←
15        usedGroundTruthPoints + {groundTruthPoint}
16      results ← results + [(detection.cls, groundTruthPoint.cls)]
17      break
18  results ← results + [(detection.cls, 0)]
19 pointsLeft ← (groundTruthPoints – usedGroundTruthPoints)
20 for groundTruthPoint in groundTruthPoints do
21   results ← results + [(0, groundTruthPoint.cls)]
22 detectionToGroundTruthPointsMatch ← results

```

As depicted in Algorithm 2, at the beginning detections are sorted by score (line 1) and the k-d tree (Bentley 1975) is constructed from ground truth points to quickly find nearest neighbors (line 2). Then empty containers for results (line 3) and used points (line 4) are prepared. The number of selected points is calculated as the minimum number out of 3 and the number of ground truth points to handle cases when the number of ground truth points is low (line 5). Then for each detection, a center point is found (line 6) and find the closest *pointNb* neighbors to this center point (line 8). Neighboring points (dots) are filtered at first by checking whether they were already used (line 9) and if the point is inside the detection bounding box (line 11). Then, if any point is left, it is marked as used (line 12) and the match of detection class and point class is added to the results (line 13); otherwise it is assigned the detection class of the 'background' class (line 15). After setting the detections to ground truth annotations, the reminding ground truth points set is constructed (line 16) and classified as detected with class 'background' (lines 17–18). Such a way of handling not detected points and false detections allows us to use already implemented metrics from the *sklearn* (Pedregosa et al. 2011) module.

3.4.5 Metrics

The proliferation index is calculated for each test image. The quality assessment of model predictions is based on the R^2 , MSE, and MAE values. As the quality of bounding boxes is irrelevant, the standard object detection metric—mean average precision—is unsuitable here. Therefore, four classification metrics (accuracy, precision, recall, and F1) were calculated based on the following values:

- True Positive (TP) is the number of correct positive predictions,
- False Positives (FP) is the number of incorrect positive predictions,
- False Negatives (FN) is the number of incorrect negative predictions,
- True Negatives (TN) is the number of correct negative predictions.

These four metrics applied in the experimental part are the following: *Accuracy* (Eq. 2) allows calculating rate of true class detections to all detections.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

Precision (Eq. 3) gives the proportion of true detection of a specific class to all these class detections. Precision is beneficial to getting information about the percentage of relevant objects among all class detections.

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

Recall (Eq. 4) depicting percentage of relevant objects of certain class detected by the model.

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

F1 (Eq. 5) is a harmonic mean of precision. *F1* metric is reacting stronger to changes in each of the components. A high *F1* score is only possible when precision and recall are high, making it a universal metric.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (5)$$

Additionally, model are compared with manual annotations by the expert using regression metrics: R^2 (Eq. 6), Mean Squared Error (Eq. 7), and Mean Absolute Error (Eq. 8) metrics.

$$R^2 = \frac{n(\sum_{i=0}^n y_i \hat{y}_i) - (\sum_{i=0}^n y_i)(\sum_{i=0}^n \hat{y}_i)}{\sqrt{(n \sum_{i=0}^n y_i^2 - (\sum_{i=0}^n y_i)^2)(n \sum_{i=0}^n \hat{y}_i^2 - (\sum_{i=0}^n \hat{y}_i)^2)}} \quad (6)$$

In those metrics, y_i equals to real proliferation index for i image calculated based on ground truth annotations, and \hat{y}_i , on the other hand, is a value of proliferation index calculated based on the model detections. R^2 is a percentage of data variance described by a model. It is one of the best statistical metrics for describing a model, but it must be accompanied by some metric describing residual error due to its limitations.

Mean Square Error is an excellent metric, strongly reacting to the outlier errors, consisting of the arithmetic mean of squared errors between observation and predicted value.

$$MSE = \frac{1}{n} \sum_{i=0}^n (y_i - \hat{y}_i)^2 \quad (7)$$

Mean Absolute Error is a metric better representing the value of error as the unit of measurement and is not squared. Therefore, calculating the simple arithmetic mean of the absolute error value between observations and the predicted value allows for a better understanding of how severe the model error is. The metrics were calculated using a widely used *sklearn.metrics* module to limit possible implementation errors.

$$MAE = \frac{1}{n} \sum_{i=0}^n \|y_i - \hat{y}_i\| \quad (8)$$

4 Results

This section will describe the model evaluation results on both the SHDC-B-Ki-67 test dataset and the LSOC-Ki-67 dataset. All experiments were conducted in a single computing environment: Ubuntu 22.04, CPU AMD Ryzen 9 5900X, GPU Nvidia RTX3080 Ti. To provide reproducibility, the *DVC* (Kuprieiev et al. 2023) pipeline was prepared, and each step, depending on a random value, was fixed on the selected seed.

4.1 Experiment 1: assessment of the models based on SHDC-B-Ki-67 testing dataset

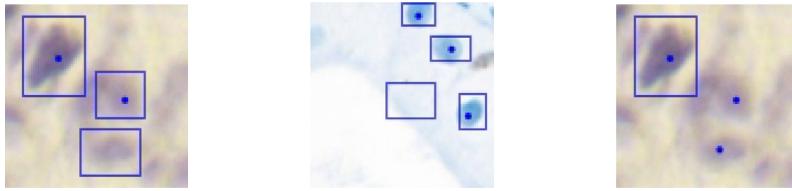
The first experiment tests the quality of models on PathoNet's test dataset SHDC-B-Ki-67. Results of the evaluation are presented in Table 2. Ki67- and Ki67+ labels in Table 2 correspond with Ki-67-negative and Ki-67-positive classes, for which the metric is calculated. For Faster R-CNN, RetinaNet, and Mask R-CNN, ResNet (He et al. 2015) backbone was used, and for SSD, VGG (Simonyan and Zisserman 2014) was used. The number next to the backbone name indicates the number of convolutional layers used in that backbone variant. YOLOv4 is using CSPDarkNet53 (Wang et al. 2019) backbone, known to be rather specific for YOLO architecture. PathoNet, on the other hand, utilizes U-Net architecture (Ronneberger et al. 2015b).

The results of experiments on SHDC-B-Ki-67, visualized in Table 2 show that models using some generic architecture for object detection or instance segmentation are slightly inferior compared to the state-of-the-art model PathoNet. SSD architecture is not the best choice for this task, based on classification metrics like Precision and F1. Other network architectures are lagging a little bit behind, but still with excellent results of F1 metric and R^2 values. RetinaNet and YOLO are third-best in the case of classification quality (Precision, Recall, and F1 taken into account), surpassed by Mask R-CNN and Faster R-CNN, and, of course, PathoNet. There are, unfortunately, cases where some of the errors depicted in Fig. 6a prove that models can still be improved. But looking at the quality of PathoNet-trained results of generic models are even more impressive.

The SHDC-B-Ki-67-supertest dataset was used to estimate the significance of the missing annotation error. Results from Table 3 show that the models' performance is reduced, primarily due to a significant decrease in Ki-67 positive cells' detection quality. In most cases, it is caused by additional expert annotations of Ki-67 positive cells in groups and cells that are not that saturated and contrasting with the background. Generic models were unaffected, as the white balancing preprocessing step helped increase the contrast between background and low-saturated cells. More commonly used models detect more cells in cell groups; therefore, new cell annotations inside the groups are not influencing the result that much. Because the SHDC-B-Ki-67-supertest dataset is relatively small, it has to be considered whether additional annotations would be significant given the whole SHDC-B-Ki-67-test. Nonetheless, this dataset allows for understanding and finding weak points of models and the training process. It is also clearly visible that additional post-processing is causing decrease in detection quality for all the architectures.

4.2 Experiment 2: assessment of the models based on LSOC-Ki-67 dataset

Results of running the experiments on the LSOC-Ki-67 dataset, visualized in Table 4 in the same fashion as in Experiment 1 identify Faster R-CNN as a possible architecture for proliferation index calculation. All 3 models achieve high F1 scores for both Ki-67-positive and Ki-67-negative cells. RetinaNet and Mask R-CNN architectures struggle with correct Ki-67-negative cell detections, while SSD and YOLOv4 models are having real issues with Ki-67-negative cell detection. YOLO architecture severely overestimates the number of cells, as can be seen on Table 5 classification metrics. PathoNet-trained is still lagging behind generic models, but on the other hand,



(a) False positive due to missing annotation (b) False positive due to invalid detection on back-ground (c) False negative due to missing detection

Fig. 6 Bounding boxes with false-positive and false-negative cases. Dots of various colors represent the ground truth annotations, and boxes represent the model prediction. Red dots and boxes depict the Ki-67-positive class; similarly, the blue color is used for the Ki-67-negative class. All images are example results of Faster R-CNN ResNet50 and examples for Fig. 6a and c are from SHIDC-B-Ki-67 test dataset and Fig. 6b is taken from LSOC-Ki-67 dataset

PathoNet trained on the full SHIDC-B-Ki-67 train dataset is achieving results comparable with inferior common models. For this step, post-processing in the form of lymphocyte reclassification is giving inconclusive improvements in detection quality. Neighborhood analysis algorithm is providing negative effect on detection quality.

4.3 Model errors overview

All the networks based on ResNet50 are rather basic solutions to the cell counting problem. Networks work well with ideally visible cells and little cell groups but, on the other hand, are not very reliable in case of false positives: empty spaces (e.g., blood vessels or artifacts) are often detected as Ki-67-negative cells. Networks based on ResNet101 are often better at handling basic problems, but they, unfortunately, developed an issue of skipping detections in cell groups. Otherwise, this backbone performs well, e.g., for the Mask R-CNN+ResNet101 model. ResNet152 is the biggest and better backbone for networks. It is better at splitting and detecting cell groups and an overall number of detections. The problem with not annotated cells did not skew the results of the models used in detections. Due to the simpler network, Faster R-CNN worked well with SHIDC-B-Ki-67-test and LSOC-Ki-67. Overall, Mask R-CNN with bigger backbones performed well in detecting cells on the images from the same laboratory as for training images. It is mainly the cause of much better behavior in splitting up of cell groups and the skipping of empty space and colored artifacts. RetinaNet, regardless of the backbone, is an excellent network for images with multiple cell crowds. The network worked relatively well and split cell groups into individual cells. One downside of such an approach was the reported several false positive cells, many of which were not correctly annotated. SSD with VGG16/19 backbone is struggling with detecting cells blended into the background, but even worse, obvious cells. We observed a lot of false negatives due to the fact many cells were just ignored. YOLOv4 was performing great with large cell's detection and failed on smaller ones. Unfortunately, the network is using much bigger windows than the SSD or RetinaNet therefore, it requires different network anchors size. PathoNet-trained could not perform even the basic task of detecting ideally visible

Table 2 Model quality metrics for detections for SHDC-B-Ki-67 test dataset

Architecture, backbone, post-processing	Accuracy $\uparrow\uparrow$		Precision $\uparrow\uparrow$		Recall $\uparrow\uparrow$		F1 $\uparrow\uparrow$		R^2 $\uparrow\uparrow$	MSE $\downarrow\downarrow$	MAE $\downarrow\downarrow$
	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+			
Faster R-CNN, ResNet50	0.707	0.901	0.738	0.782	0.696	0.855	0.716	0.817	0.620	0.024	0.085
Faster R-CNN, ResNet101	0.682	0.903	0.746	0.800	0.616	0.834	0.675	0.817	0.329	0.041	0.107
Faster R-CNN, ResNet152	0.699	0.905	0.701	0.799	0.739	0.833	0.719	0.815	0.716	0.018	0.079
RetinaNet, ResNet50	0.669	0.891	0.721	0.804	0.644	0.777	0.680	0.790	0.679	0.020	0.090
RetinaNet, ResNet101	0.678	0.898	0.708	0.815	0.684	0.783	0.696	0.799	0.726	0.017	0.078
RetinaNet, ResNet152	0.679	0.899	0.706	0.841	0.698	0.756	0.702	0.796	0.722	0.017	0.083
Mask R-CNN, ResNet50	0.685	0.903	0.684	0.800	0.734	0.820	0.709	0.810	0.738	0.016	0.079
Mask R-CNN, ResNet101	0.680	0.904	0.732	0.801	0.632	0.832	0.678	0.816	0.465	0.033	0.098
Mask R-CNN, ResNet152	0.682	0.906	0.714	0.785	0.638	0.858	0.674	0.820	0.259	0.045	0.118
SSD, VGG16	0.626	0.876	0.592	0.672	0.649	0.883	0.619	0.764	0.517	0.030	0.115
SSD, VGG19	0.623	0.878	0.589	0.675	0.646	0.888	0.616	0.767	0.545	0.028	0.115
YOLOv4, Darknet	0.679	0.882	0.688	0.760	0.695	0.768	0.691	0.764	0.709	0.018	0.081
Faster R-CNN, ResNet50, lym-reclass	0.703	0.903	0.726	0.773	0.684	0.864	0.704	0.816	0.725	0.017	0.078
Faster R-CNN, ResNet101, lym-reclass	0.693	0.904	0.710	0.781	0.684	0.853	0.697	0.816	0.747	0.016	0.076
Faster R-CNN, ResNet152, lym-reclass	0.691	0.905	0.714	0.802	0.688	0.833	0.700	0.817	0.742	0.016	0.078
RetinaNet, ResNet50, lym-reclass	0.662	0.892	0.732	0.807	0.607	0.777	0.664	0.791	0.683	0.019	0.090
RetinaNet, ResNet101, lym-reclass	0.674	0.899	0.689	0.792	0.666	0.807	0.677	0.799	0.746	0.016	0.080
RetinaNet, ResNet152, lym-reclass	0.673	0.900	0.716	0.844	0.657	0.756	0.685	0.798	0.745	0.016	0.081
Mask R-CNN, ResNet50, lym-reclass	0.681	0.906	0.676	0.788	0.698	0.836	0.687	0.811	0.733	0.016	0.081
Mask R-CNN, ResNet101, lym-reclass	0.692	0.905	0.702	0.785	0.697	0.851	0.699	0.817	0.747	0.016	0.075
Mask R-CNN, ResNet152, lym-reclass	0.692	0.906	0.708	0.789	0.689	0.854	0.698	0.820	0.716	0.017	0.083
SSD, VGG16, lym-reclass	0.624	0.877	0.599	0.676	0.606	0.883	0.603	0.766	0.542	0.028	0.115
SSD, VGG19, lym-reclass	0.622	0.879	0.599	0.679	0.601	0.888	0.600	0.769	0.558	0.027	0.115
YOLOv4, Darknet, lym-reclass	0.680	0.882	0.717	0.762	0.631	0.768	0.671	0.765	0.686	0.019	0.084
Faster R-CNN, ResNet50, lym-reclass+neigh	0.687	0.902	0.758	0.784	0.610	0.855	0.676	0.818	0.230	0.047	0.117

Table 2 (continued)

Architecture, backbone, post-processing	Accuracy ↑↑		Precision ↑↑		Recall ↑↑		F1 ↑↑		R^2 ↑↑	MSE ↓↓	MAE ↓↓
	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+			
Faster R-CNN, ResNet101, lym-reclass+neigh	0.682	0.903	0.746	0.800	0.616	0.834	0.675	0.817	0.329	0.041	0.107
Faster R-CNN, ResNet152, lym-reclass+neigh	0.683	0.905	0.720	0.802	0.648	0.833	0.682	0.817	0.455	0.033	0.102
RetinaNet, ResNet50, lym-reclass+neigh	0.655	0.892	0.734	0.807	0.584	0.777	0.650	0.791	0.446	0.034	0.108
RetinaNet, ResNet101, lym-reclass+neigh	0.662	0.898	0.727	0.818	0.598	0.783	0.656	0.800	0.380	0.038	0.104
RetinaNet, ResNet152, lym-reclass+neigh	0.665	0.900	0.718	0.844	0.629	0.756	0.670	0.798	0.473	0.032	0.102
Mask R-CNN, ResNet50, lym-reclass+neigh	0.672	0.904	0.717	0.802	0.615	0.820	0.662	0.811	0.189	0.050	0.117
Mask R-CNN, ResNet101, lym-reclass+neigh	0.680	0.904	0.732	0.801	0.632	0.832	0.678	0.816	0.465	0.033	0.098
Mask R-CNN, ResNet152, lym-reclass+neigh	0.682	0.906	0.714	0.785	0.638	0.858	0.674	0.820	0.259	0.045	0.118
SSD, VGG16, lym-reclass+neigh	0.619	0.877	0.599	0.676	0.577	0.883	0.588	0.766	0.317	0.042	0.133
SSD, VGG19, lym-reclass+neigh	0.617	0.879	0.598	0.679	0.568	0.888	0.583	0.769	0.293	0.043	0.135
YOLOv4, Darknet, lym-reclass+neigh	0.667	0.882	0.730	0.762	0.568	0.768	0.639	0.765	-0.058	0.065	0.139
PathoNet-trained, UNet	0.584	0.846	0.735	0.849	0.449	0.552	0.558	0.669	0.581	0.026	0.102
PathoNet, UNet	0.729	0.922	0.699	0.832	0.814	0.852	0.752	0.842	0.655	0.022	0.076

Best value for each metric is highlighted

Table 3 Model quality metrics for detections for SHDC-B-Ki-67-supertest dataset

Architecture, Backbone, Post-processing	Accuracy ↑↑		Precision ↑↑		Recall ↑↑		F1 ↑↑		R^2 ↑↑	MSE ↓↓	MAE ↓↓
	Ki67- Ki67+	Ki67+ Ki67+	Ki67- Ki67+	Ki67+ Ki67+	Ki67- Ki67+	Ki67+ Ki67+	Ki67- Ki67+	Ki67+ Ki67+			
Faster R-CNN, ResNet50	0.706	0.884	0.805	0.902	0.646	0.624	0.717	0.737	0.894	0.008	0.064
Faster R-CNN, ResNet101	0.665	0.878	0.796	0.892	0.560	0.602	0.657	0.719	0.860	0.011	0.076
Faster R-CNN, ResNet152	0.704	0.876	0.773	0.889	0.677	0.590	0.722	0.709	0.797	0.016	0.092
RetinaNet, ResNet50	0.668	0.868	0.779	0.892	0.591	0.560	0.672	0.688	0.868	0.010	0.077
RetinaNet, ResNet101	0.669	0.866	0.772	0.902	0.601	0.545	0.676	0.679	0.822	0.014	0.084
RetinaNet, ResNet152	0.673	0.867	0.762	0.920	0.624	0.534	0.686	0.676	0.793	0.016	0.095
Mask R-CNN, ResNet50	0.680	0.876	0.756	0.904	0.650	0.582	0.699	0.708	0.796	0.016	0.092
Mask R-CNN, ResNet101	0.667	0.875	0.780	0.891	0.579	0.588	0.665	0.709	0.837	0.013	0.083
Mask R-CNN, ResNet152	0.667	0.878	0.765	0.864	0.588	0.616	0.665	0.719	0.499	0.040	0.122
SSD, VGG16	0.609	0.862	0.642	0.721	0.558	0.675	0.597	0.697	0.629	0.029	0.122
SSD, VGG19	0.611	0.870	0.641	0.738	0.569	0.689	0.603	0.713	0.666	0.026	0.123
YOLOv4, Darknet	0.667	0.866	0.719	0.860	0.641	0.547	0.677	0.668	0.851	0.012	0.077
Faster R-CNN, ResNet50, lym-reclass	0.687	0.886	0.786	0.892	0.618	0.633	0.692	0.741	0.862	0.011	0.076
Faster R-CNN, ResNet101, lym-reclass	0.679	0.882	0.770	0.881	0.614	0.623	0.683	0.730	0.844	0.012	0.080
Faster R-CNN, ResNet152, lym-reclass	0.684	0.876	0.772	0.889	0.628	0.590	0.693	0.709	0.802	0.016	0.090
RetinaNet, ResNet50, lym-reclass	0.656	0.868	0.784	0.892	0.553	0.560	0.649	0.688	0.874	0.010	0.075
RetinaNet, ResNet101, lym-reclass	0.656	0.874	0.742	0.880	0.591	0.580	0.658	0.699	0.811	0.015	0.089
RetinaNet, ResNet152, lym-reclass	0.661	0.867	0.767	0.920	0.586	0.534	0.664	0.676	0.801	0.016	0.092
Mask R-CNN, ResNet50, lym-reclass	0.669	0.880	0.741	0.888	0.626	0.601	0.679	0.717	0.787	0.017	0.098
Mask R-CNN, ResNet101, lym-reclass	0.680	0.881	0.755	0.876	0.632	0.619	0.688	0.725	0.821	0.014	0.088
Mask R-CNN, ResNet152, lym-reclass	0.676	0.877	0.764	0.869	0.616	0.613	0.682	0.719	0.748	0.020	0.103
SSD, VGG16, lym-reclass	0.598	0.862	0.639	0.721	0.518	0.675	0.573	0.697	0.620	0.030	0.125
SSD, VGG19, lym-reclass	0.600	0.870	0.638	0.738	0.527	0.689	0.577	0.713	0.662	0.027	0.125
YOLOv4, Darknet, lym-reclass	0.652	0.866	0.730	0.860	0.576	0.547	0.644	0.668	0.862	0.011	0.074
Faster R-CNN, ResNet50, lym-reclass+neigh	0.666	0.884	0.809	0.902	0.550	0.624	0.655	0.737	0.874	0.010	0.073

Table 3 (continued)

Architecture, Backbone, Post-processing	Accuracy ↑↑		Precision ↑↑		Recall ↑↑		F1 ↑↑		R^2 ↑↑	MSE ↓↓	MAE ↓↓
	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+			
Faster R-CNN, ResNet101, lym-reclass+neigh	0.665	0.878	0.796	0.892	0.560	0.602	0.657	0.719	0.860	0.011	0.076
Faster R-CNN, ResNet152, lym-reclass+neigh	0.677	0.876	0.775	0.889	0.606	0.590	0.680	0.709	0.806	0.015	0.088
RetinaNet, ResNet50, lym-reclass+neigh	0.653	0.868	0.789	0.892	0.540	0.560	0.641	0.688	0.876	0.010	0.075
RetinaNet, ResNet101, lym-reclass+neigh	0.645	0.866	0.777	0.902	0.535	0.545	0.634	0.679	0.826	0.014	0.084
RetinaNet, ResNet152, lym-reclass+neigh	0.658	0.867	0.770	0.920	0.576	0.534	0.659	0.676	0.800	0.016	0.093
Mask R-CNN, ResNet50, lym-reclass+neigh	0.653	0.876	0.770	0.904	0.558	0.582	0.647	0.708	0.816	0.015	0.088
Mask R-CNN, ResNet101, lym-reclass+neigh	0.667	0.875	0.780	0.891	0.579	0.588	0.665	0.709	0.837	0.013	0.083
Mask R-CNN, ResNet152, lym-reclass+neigh	0.667	0.878	0.765	0.864	0.588	0.616	0.665	0.719	0.499	0.040	0.122
SSD, VGG16, lym-reclass+neigh	0.592	0.862	0.636	0.721	0.501	0.675	0.560	0.697	0.580	0.033	0.130
SSD, VGG19, lym-reclass+neigh	0.592	0.870	0.634	0.738	0.506	0.689	0.563	0.713	0.629	0.029	0.132
YOLOv4, Darknet, lym-reclass+neigh	0.639	0.866	0.736	0.860	0.529	0.547	0.615	0.668	0.853	0.012	0.079
PathoNet-trained, UNet	0.561	0.820	0.798	0.940	0.356	0.357	0.492	0.517	0.597	0.032	0.092
PathoNet, UNet	0.577	0.849	0.793	0.939	0.389	0.469	0.522	0.625	0.836	0.013	0.079

Best value for each metric is highlighted

Table 4 Model quality metrics for detections for LSOC-Ki-67 dataset

Architecture, Backbone, Post-processing	Accuracy ↑↑		Precision ↑↑		Recall ↑↑		F1 ↑↑		R^2 ↑↑	MSE ↓↓	MAE ↓↓
	Ki67- Ki67+	Ki67+ Ki67+	Ki67- Ki67+	Ki67+ Ki67+	Ki67- Ki67+	Ki67+ Ki67+	Ki67- Ki67+	Ki67+ Ki67+			
Faster R-CNN, ResNet50	0.630	0.936	0.882	0.880	0.602	0.701	0.715	0.780	0.832	0.004	0.041
Faster R-CNN, ResNet101	0.628	0.933	0.875	0.881	0.600	0.677	0.712	0.766	0.626	0.008	0.050
Faster R-CNN, ResNet152	0.637	0.932	0.885	0.888	0.612	0.670	0.724	0.764	0.213	0.018	0.060
RetinaNet, ResNet50	0.580	0.912	0.900	0.928	0.531	0.512	0.668	0.660	0.902	0.002	0.031
RetinaNet, ResNet101	0.583	0.923	0.885	0.889	0.536	0.607	0.667	0.721	0.850	0.003	0.037
RetinaNet, ResNet152	0.625	0.928	0.888	0.909	0.594	0.622	0.712	0.738	0.905	0.002	0.028
Mask R-CNN, ResNet50	0.608	0.919	0.883	0.862	0.575	0.602	0.696	0.709	0.837	0.004	0.040
Mask R-CNN, ResNet101	0.614	0.927	0.869	0.894	0.585	0.623	0.699	0.734	0.418	0.013	0.052
Mask R-CNN, ResNet152	0.620	0.937	0.866	0.898	0.595	0.683	0.705	0.776	0.482	0.012	0.042
SSD, VGG16	0.639	0.934	0.797	0.843	0.669	0.693	0.727	0.761	0.951	0.001	0.023
SSD, VGG19	0.593	0.934	0.834	0.851	0.572	0.707	0.678	0.772	0.907	0.002	0.035
YOLOv4, Darknet	0.562	0.930	0.649	0.872	0.687	0.566	0.668	0.686	0.616	0.009	0.064
Faster R-CNN, ResNet50, lym-reclass	0.624	0.937	0.875	0.875	0.595	0.710	0.708	0.784	0.821	0.004	0.043
Faster R-CNN, ResNet101, lym-reclass	0.648	0.936	0.860	0.871	0.638	0.698	0.733	0.775	0.748	0.006	0.043
Faster R-CNN, ResNet152, lym-reclass	0.623	0.932	0.887	0.888	0.589	0.670	0.708	0.764	0.191	0.018	0.063
RetinaNet, ResNet50, lym-reclass	0.574	0.912	0.900	0.928	0.522	0.512	0.661	0.660	0.899	0.002	0.032
RetinaNet, ResNet101, lym-reclass	0.594	0.926	0.875	0.868	0.552	0.643	0.677	0.739	0.825	0.004	0.042
RetinaNet, ResNet152, lym-reclass	0.616	0.928	0.889	0.909	0.580	0.622	0.702	0.738	0.899	0.002	0.029
Mask R-CNN, ResNet50, lym-reclass	0.613	0.921	0.872	0.856	0.585	0.620	0.701	0.719	0.841	0.004	0.039
Mask R-CNN, ResNet101, lym-reclass	0.634	0.930	0.855	0.886	0.623	0.647	0.721	0.748	0.843	0.004	0.040
Mask R-CNN, ResNet152, lym-reclass	0.627	0.937	0.866	0.904	0.607	0.680	0.713	0.776	0.923	0.002	0.028
SSD, VGG16, lym-reclass	0.629	0.934	0.797	0.843	0.649	0.693	0.715	0.761	0.949	0.001	0.025
SSD, VGG19, lym-reclass	0.585	0.934	0.833	0.851	0.558	0.707	0.669	0.772	0.891	0.002	0.038
YOLOv4, Darknet, lym-reclass	0.561	0.930	0.654	0.872	0.665	0.566	0.659	0.686	0.637	0.008	0.061
Faster R-CNN, ResNet50, lym-reclass+neigh	0.601	0.936	0.884	0.880	0.557	0.701	0.683	0.780	0.711	0.007	0.055

Table 4 (continued)

Architecture, Backbone, Post-processing	Accuracy ↑↑		Precision ↑↑		Recall ↑↑		F1 ↑↑		R^2 ↑↑	MSE ↓↓	MAE ↓↓
	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+			
Faster R-CNN, ResNet101, lym-reclass+neigh	0.628	0.933	0.875	0.881	0.600	0.677	0.712	0.766	0.626	0.008	0.050
Faster R-CNN, ResNet152, lym-reclass+neigh	0.619	0.932	0.887	0.888	0.584	0.670	0.704	0.764	0.171	0.019	0.065
RetinaNet, ResNet50, lym-reclass+neigh	0.573	0.912	0.901	0.928	0.521	0.512	0.660	0.660	0.899	0.002	0.032
RetinaNet, ResNet101, lym-reclass+neigh	0.566	0.923	0.891	0.889	0.507	0.607	0.646	0.721	0.611	0.009	0.056
RetinaNet, ResNet152, lym-reclass+neigh	0.615	0.928	0.889	0.909	0.579	0.622	0.701	0.738	0.898	0.002	0.030
Mask R-CNN, ResNet50, lym-reclass+neigh	0.588	0.919	0.884	0.862	0.544	0.602	0.674	0.709	0.799	0.005	0.045
Mask R-CNN, ResNet101, lym-reclass+neigh	0.614	0.927	0.869	0.894	0.585	0.623	0.699	0.734	0.418	0.013	0.052
Mask R-CNN, ResNet152, lym-reclass+neigh	0.620	0.937	0.866	0.898	0.595	0.683	0.705	0.776	0.482	0.012	0.042
SSD, VGG16, lym-reclass+neigh	0.627	0.934	0.797	0.843	0.645	0.693	0.713	0.761	0.943	0.001	0.026
SSD, VGG19, lym-reclass+neigh	0.584	0.934	0.834	0.851	0.556	0.707	0.667	0.772	0.864	0.003	0.039
YOLOv4, Darknet, lym-reclass+neigh	0.558	0.930	0.654	0.872	0.655	0.566	0.654	0.686	0.506	0.011	0.067
PathoNet-trained, UNet	0.452	0.911	0.816	0.901	0.372	0.507	0.511	0.649	0.315	0.015	0.080
PathoNet, UNet	0.413	0.905	0.848	0.876	0.308	0.492	0.451	0.630	0.380	0.014	0.084

Best value for each metric is highlighted

Table 5 Cell counts reported for models for each of the testing datasets compared with ground truth annotations

Architecture,Backbone,Post-processing	SHIDC-B-Ki-67		SHIDC-B-Ki-67		LSOC-Ki-67	
	test		supertest			
	Ki67-	Ki67+	Ki67-	Ki67+	Ki67-	Ki67+
Ground truth	32643	15755	2604	1180	124344	26180
Faster R-CNN,ResNet50	30763	17241	2090	816	84826	20848
Faster R-CNN,ResNet101	26934	16424	1831	796	85206	20101
Faster R-CNN,ResNet152	34379	16426	2281	783	86010	19743
RetinaNet,ResNet50	29174	15226	1975	741	73329	14435
RetinaNet,ResNet101	31500	15137	2028	713	75323	17871
RetinaNet,ResNet152	32270	14171	2132	685	83152	17910
Mask R-CNN,ResNet50	35025	16163	2240	760	80936	18290
Mask R-CNN,ResNet101	28202	16363	1934	779	83665	18249
Mask R-CNN,ResNet152	29180	17225	2003	841	85462	19924
SSD,VGG16	35816	20698	2265	1106	104304	21528
SSD,VGG19	35760	20722	2311	1101	85288	21736
YOLOv4,Darknet	32969	15923	2321	750	131711	17000
Faster R-CNN,ResNet50,lym-reclass	30726	17609	2046	837	84535	21248
Faster R-CNN,ResNet101,lym-reclass	31457	17203	2077	834	92270	20977
Faster R-CNN,ResNet152,lym-reclass	31439	16368	2118	783	82653	19743
RetinaNet,ResNet50,lym-reclass	27085	15176	1837	741	72144	14435
RetinaNet,ResNet101,lym-reclass	31563	16061	2073	777	78400	19403
RetinaNet,ResNet152,lym-reclass	29923	14120	1989	685	81196	17910
Mask R-CNN,ResNet50,lym-reclass	33721	16706	2197	798	83442	18980
Mask R-CNN,ResNet101,lym-reclass	32420	17092	2181	833	90586	19106
Mask R-CNN,ResNet152,lym-reclass	31755	17055	2099	832	87133	19687
SSD,VGG16,lym-reclass	33016	20590	2112	1106	101281	21528
SSD,VGG19,lym-reclass	32777	20609	2149	1101	83318	21736
YOLOv4,Darknet,lym-reclass	28700	15871	2053	750	126383	17000
Faster R-CNN,ResNet50,lym-reclass+neigh	26260	17176	1768	816	78341	20848
Faster R-CNN,ResNet101,lym-reclass+neigh	26934	16424	1831	796	85206	20101
Faster R-CNN,ResNet152,lym-reclass+neigh	29402	16368	2038	783	81795	19743
RetinaNet,ResNet50,lym-reclass+neigh	25957	15176	1782	741	72000	14435
RetinaNet,ResNet101,lym-reclass+neigh	26848	15082	1794	713	70751	17871
RetinaNet,ResNet152,lym-reclass+neigh	28570	14120	1947	685	80931	17910
Mask R-CNN,ResNet50,lym-reclass+neigh	27990	16105	1888	760	76569	18290
Mask R-CNN,ResNet101,lym-reclass+neigh	28202	16363	1934	779	83665	18249
Mask R-CNN,ResNet152,lym-reclass+neigh	29180	17225	2003	841	85462	19924
SSD,VGG16,lym-reclass+neigh	31411	20590	2050	1106	100720	21528
SSD,VGG19,lym-reclass+neigh	31009	20609	2077	1101	82949	21736
YOLOv4,Darknet,lym-reclass+neigh	25382	15871	1871	750	124472	17000
PathoNet-trained,UNet	19961	10256	1162	448	56721	14725
PathoNet,UNet	38025	16146	1278	589	45111	14695

cells. We observed results filled with false negative detections due not improper detection. Base PathoNet was unbeatable on the test set from SHIDC-B-Ki-67. However, the LSOC-Ki-67 test set caused many false negatives, as many cell groups were calculated as single cells. Overall, PathoNet was not handling the change in the dataset well, as the number of errors reported increased dramatically.

4.4 Side-by-side comparison

In this section, models are compared with each other in the context of all experiments.

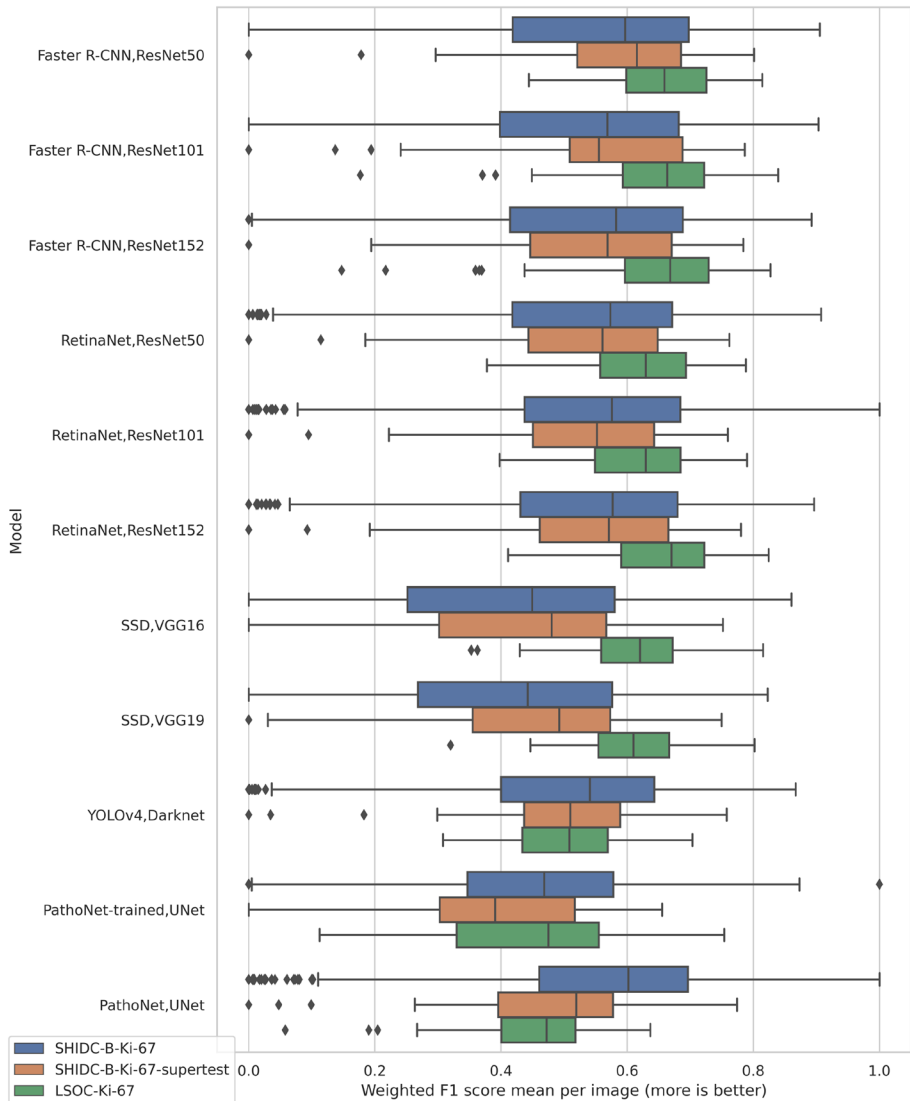
Figure 7 is a box and whiskers plot visualizing the distributions of the variables. The plot for single variables consists of the interquartile range (IQR) box containing the second and third quartiles of the distribution. In this range, 50% of the observations are located. The IQR box also includes the line indicating the position of the median value, which is the middle value of the observations. The position of the median line in the IQR box signifies whether the distribution is left or right-skewed. Whiskers show the range of the first and fourth quartiles, visualizing the spread of the values on the axis. Finally, on the plot, there are also outliers indicated using diamonds - observations outside the $(Q1 - 1.5 * IQR; Q3 + 1.5 * IQR)$ range.

Data for Fig. 7 is the weighted F1 score computed as a weighted mean of F1 scores per class on a single image. This is a different approach from the results tables (Tables 2, 3, and 4), as F1 is calculated for each of the classes and based on values from all images. Therefore, this approach allows us to better visualize the model's performance as it indicates possible outlying images. Moreover, such information allows us to plot the distribution and range of the resulting F1 values.

Based on Fig. 7 it can be concluded that PathoNet-trained, YOLOv4, and SSD-based models are underperforming compared with Faster R-CNN, RetinaNet, Mask R-CNN architectures and PathoNet trained on full SHIDC-B-Ki-67 train dataset as signified by the distributions. All 3 generic architectures left (Faster R-CNN, RetinaNet, and Mask R-CNN) are close to the PathoNet comparing the distributions for SHIDC-B-Ki-67 and even closer on the SHIDC-B-Ki-67 superset. For LSOC-Ki-67, the plot indicates the superiority of generic architectures over task-specific PathoNet architecture. Moreover, Faster R-CNN seems to achieve better IQR range width and Q1-Q4 width results over RetinaNet and Mask R-CNN.

5 Discussion

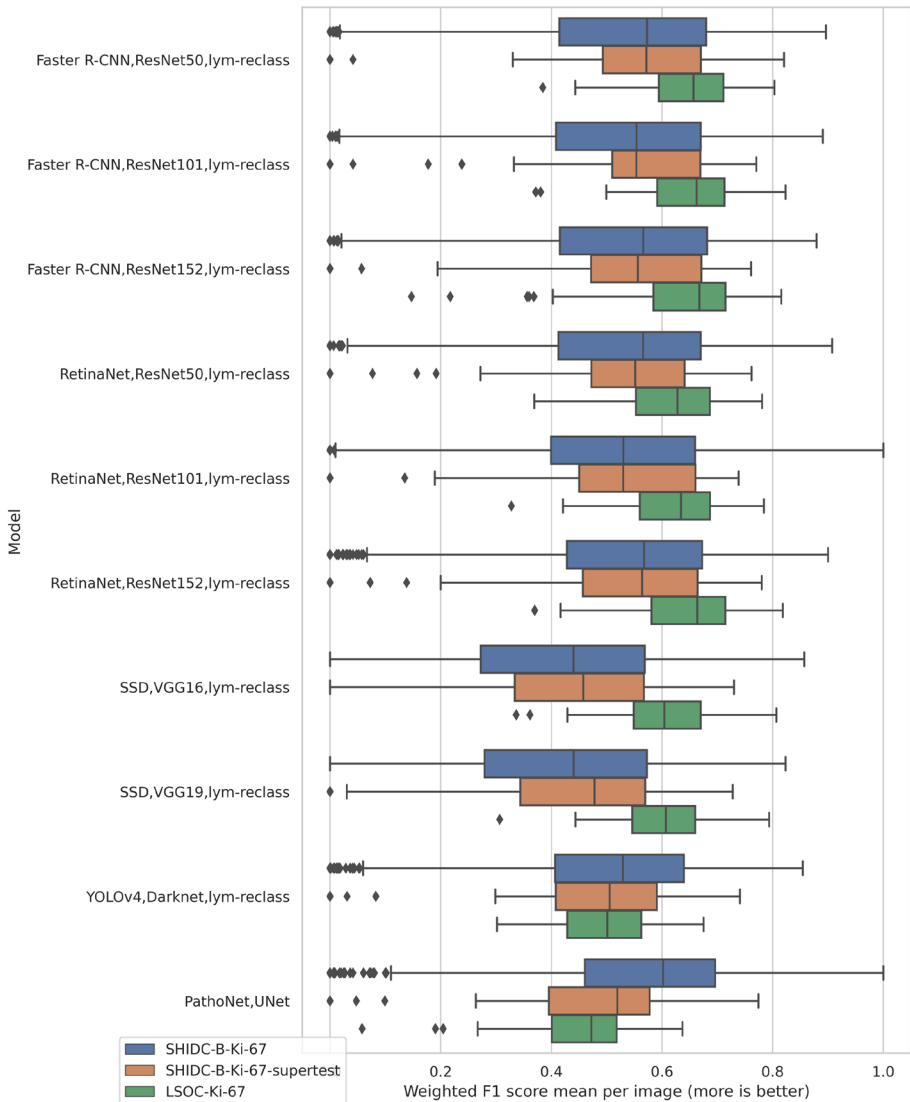
The primary goal of the research, which was to test whether specialized methods for counting Ki-67 stained cells are significantly better than generic models without hyperparameter tuning and pretrained on a task different from cell detection, was achieved. By analyzing the results and taking a solid look into the PathoNet-trained, it can be presumed that generic models with the training set of the size of a complete training set of PathoNet could even outperform PathoNet in the task. It is worth underlying that a very effective algorithmic pipeline was developed and by combining object detection/instance segmentation models with another classifier and unsupervised method for additional reclassification steps based on the neighboring cells results are greatly improved.



(a) Weighted F1 without postprocessing

Fig. 7 Weighted F1 score per image. The box on the chart represents the range where 50% of the observations are located. Whiskers depict the first and last quartiles of the interquartile range, and outliers are represented as diamonds. They are visible on the left side of the plot

The main benefit of using other architectures than those relying on U-Net is a better model selection depending on the model use case. With its lightning speed of detection, YOLO architecture could be used for real-time cell detection and fast proliferation index estimation during tissue scanning. Mask R-CNN, with its rather unique output of object mask, could be used

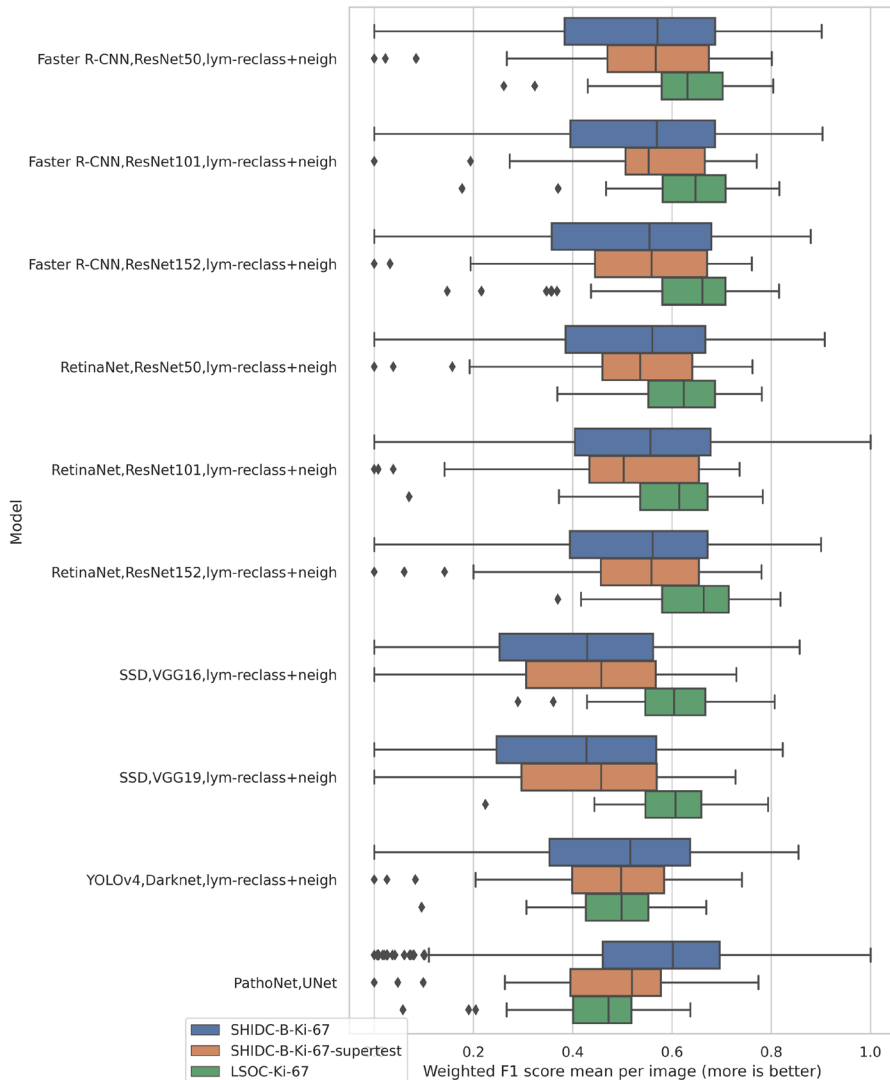


(b) Weighted F1 with lymphocyte reclassification postprocessing

Fig. 7 (continued)

to clearly show the margins of the cell to the pathology doctor. Faster R-CNN and RetinaNet could be used as object detection models in automatic proliferation index calculators.

What is more, it is observed that in complex cases like, e.g., a cluster of cells, generic models are handling the input better, giving clear margins between the object in contrast to the PathoNet that is relying on U-Net convolutional neural network.



(c) Weighted F1 with lymphocyte reclassification and neighbourhood analysis postprocessing

Fig. 7 (continued)

On the other hand, models unfortunately were unable to be fully utilized on a separate dataset. Despite preprocessing the input data to minimize the effects of color variation, and post-processing to counter class imbalance in the training set, the results are far from perfect. One explanation could be a too big difference in the histograms of images in both datasets. The SHIDC-B-Ki-67 dataset is colored yellow, and even with white balancing, the color distribution is different from the LSOC-Ki-67. Other factors that could have impacted the experiment

are blurry training images and class imbalance in the training dataset compared to the test dataset.

Moreover, it is identified that comparing models based on the calculated proliferation index from its detections is not a real possibility. As seen in examples of YOLOv4 or PathoNet-trained R^2 , MSE and MAE metrics could be showing great regression results despite low-quality detection metrics. It might be caused by simply keeping the same ratio of class detections; therefore, if a model is not performing well enough in detecting both classes, the proliferation index will remain the same. Thus, in the work, the results of R^2 , MAE, and MSE metrics were not analyzed as those metrics could be misleading.

During the research, two significant factors affecting model performance were detected. The first one is the number of lymphocytes on the images. As LSOC-Ki-67 had not selected hot spots, the balance of lymphocytes to Ki-67-negative cells was different from the SHIDC-B-Ki-67-subset train dataset or even the SHIDC-B-Ki-67-train dataset. With the novel approach to pre- and post-processing of data, this factor impact can be reduced significantly. Such actions give the option to use pipelines without the need to mask the pROI. Another critical component was scaling datasets to training dataset resolution. The process of adjusting scales was very cumbersome and required an expert to calibrate images accordingly based on cues from the specific cells (lymphocytes, erythrocytes).

Most impediments in the research could be tracked to the training data—the quality of data, the quality of annotations, and the number of annotated samples. Authors look forward to the new generative neural network research, allowing artificially balanced datasets with high-quality image output and automated annotation generation. Such a solution with the method presented in a paper could be a key to training generic architectures, resulting in overall higher metrics.

6 Conclusion

This paper presents that the results of generic object detection and instance segmentation models are comparable to those of the cell detection dedicated model. Using pre- and post-processing steps, those models could match the dedicated model's results despite a reduced train dataset and even outperform PathoNet on a custom dataset sourced from an unrelated laboratory—LSOC-Ki-67. Moreover, although there are significant differences between the two datasets, the novel cell detection pipeline allowing for using the models on such different datasets was developed, showing the significance of pre- and post-processing in computer vision tasks.

Furthermore, by adding post-processing steps, an original approach to the lymphocyte classification problem is proposed. It applies a deep learning solution and a classical one to solve it. The deep learning solution is based on a multimodal convolutional neural network, and the classical one uses neighborhood analysis. The performance of lymphocyte reclassification was not significant, and neighborhood analysis method was causing a drop in quality.

The performed experiments show that generic object detection and image segmentation architectures, even without hyperparameter optimization, are achieving similar results or even outperforming dedicated architectures. Moreover, depending on the pathologist's needs and tasks, generic object detection and image segmentation architectures allow full pipeline utilization and achieve high-quality results.

In future works, it is planned to utilize image generation to balance datasets and extend train and test datasets with data from multiple independent laboratories. The primary purpose of those works is to improve source material quality on which models can train. Furthermore, the use of more advanced image normalization algorithms to handle better image data of various characteristics is considered.

The datasets generated during and/or analyzed during the current study are available from the corresponding author upon reasonable request. Source code is available at https://github.com/MichalKarol/do_we_need_dedicated_models on MIT license.

Acknowledgements We would like to thank our dataset annotation team: Mateusz Gwizdź, Natalia Krzysztofik, Wojciech Kubera, and Hanna Maruchniak. Michał, Karol would also like to thank Agata Skibińska for her support, comments, and corrections.

Author Contributions Michał, Karol: Conceptualization, Methodology, Software, Formal analysis, Investigation, Data Curation, Writing—Original Draft, Visualization. Martin Tabakov: Supervision over study and manuscript preparation, Manuscript editing, Final approval of the article. Urszula Markowska-Kaczmar: Formal analysis, Supervision, Writing—Original Draft, Manuscript editing, Final approval of the article. Łukasz Fulawka: Resources, Data Curation, Supervision.

Funding The authors received no financial support for this article's research, authorship, or publication.

Declarations

Conflict of interest We declare that we have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethical approval The research was conducted in accordance with local guidelines and regulations. The study was approved by the local ethics committee (Bioethics Committee Wrocław Medical University).

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Bentley JL (1975) Multidimensional binary search trees used for associative searching. *Commun ACM* 18(9):509–517. <https://doi.org/10.1145/361002.361007>
- Bochkovskiy A, Wang CY, Liao HYM (2020) Yolov4: optimal speed and accuracy of object detection. *arXiv:2004.10934*
- Dimitriou N, Arandjelović O, Caie PD (2019) Deep learning for whole slide image analysis: an overview. *Front Med*. <https://doi.org/10.3389/fmed.2019.00264>
- Esteva KA, Chou Yeung S, Naik N et al (2021) Deep learning-enabled medical computer vision. *NPJ Digit Med*. <https://doi.org/10.1038/s41746-020-00376-2>
- Feng M, Deng Y, Yang L et al (2020) Automated quantitative analysis of Ki-67 staining and he images recognition and registration based on whole tissue sections in breast carcinoma. *Diagn Pathol* 15(1):1–12
- Fulawka L, Halon A (2016) Proliferation index evaluation in breast cancer using imageJ and immunoratio applications. *Anticancer Res* 36:3965–72
- Fulawka L, Halon A (2017) Ki-67 evaluation in breast cancer: the daily diagnostic practice. *Indian J Pathol Microbiol* 60(2):177–184. https://doi.org/10.4103/IJPM.IJPM_732_15

- Fulawka L, Blaszczyk J, Tabakov M et al (2022) Assessment of Ki-67 proliferation index with deep learning in DCIS (ductal carcinoma in situ). *Sci Rep*. <https://doi.org/10.1038/s41598-022-06555-3>
- Geread RS, Sivanandarajah A, Brouwer ER et al (2021) Pinet-an automated proliferation index calculator framework for Ki67 breast cancer images. *Cancers*. <https://doi.org/10.3390/cancers13010011>
- Girshick R (2015) Fast R-CNN. In: Proceedings of the 2015 IEEE international conference on computer vision (ICCV). IEEE Computer Society, USA, ICCV '15, pp 1440–1448. <https://doi.org/10.1109/ICCV.2015.169>
- Govind D, Jen KY, Matsukuma K et al (2020) Improving the accuracy of gastrointestinal neuroendocrine tumor grading with deep learning. *Sci Rep* 10(1):2045–2322. <https://doi.org/10.1038/s41598-020-67880-z>
- He K, Zhang X, Ren S et al (2015) Deep residual learning for image recognition. *CoRR abs/1512.03385*. <https://arxiv.org/abs/arXiv:1512.03385>
- He K, Gkioxari G, Dollár P et al (2017) Mask R-CNN. In: 2017 IEEE international conference on computer vision (ICCV), pp 2980–2988
- Huang G, Liu Z, Weinberger KQ (2016) Densely connected convolutional networks. *CoRR abs/1608.06993*. <https://arxiv.org/abs/arXiv:1608.06993>
- Inwald EC, Klinkhammer-Schalke M, Hofstaedter F et al (2013) Ki-67 is a prognostic parameter in breast cancer patients: results of a large population-based cohort of a cancer registry. *Breast Cancer Res Treat* 139:539–552
- Iqbal I, Younus M, Walayat K et al (2021) Automated multi-class classification of skin lesions through deep convolutional neural network with dermoscopic images. *Comput Med Imaging Graph* 88:101,843. <https://doi.org/10.1016/j.compmedimag.2020.101843>
- Kammerer-Jacquet SF, Ahmad A, Iler H et al (2019) Ki-67 is an independent predictor of prostate cancer death in routine needle biopsy samples: proving utility for routine assessments. *Mod Pathol* 32(9):1303–1309
- Kingma DP, Ba J (2017) Adam: a method for stochastic optimization. [arXiv:1412.6980](https://arxiv.org/abs/1412.6980)
- Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L et al (eds) *Advances in neural information processing systems*, vol 25. Curran Associates Inc., New York, pp 1097–1105
- Kuprieiev R, Skshetry, Rowlands P et al (2023) DVC: data version control—git for data & models. <https://doi.org/10.5281/zenodo.7990791>
- LeCun Y, Bottou L, Bengio Y et al (1998) Gradient-based learning applied to document recognition. In: *Proceedings of the IEEE*, pp 2278–2324
- Li X, Xu Y, Xiang F et al (2021) Kinet: a non-invasive method for predicting Ki67 index of glioma. In: 2021 IEEE international conference on image processing (ICIP), pp 150–154. <https://doi.org/10.1109/ICIP42928.2021.9506741>
- Li L, Han D, Yu Y et al (2022) Artificial intelligence-assisted interpretation of Ki-67 expression and repeatability in breast cancer. *Diagn Pathol* 17(20):1746–1796
- Lin TY, Goyal P, Girshick RB et al (2017) Focal loss for dense object detection. 2017 IEEE international conference on computer vision (ICCV), pp 2999–3007
- Liu W, Anguelov D, Erhan D et al (2016) SSD: single shot multibox detector. In: Leibe B, Matas J, Sebe N et al (eds) *Computer vision—ECCV 2016*. Springer, Cham, pp 21–37
- Liu Y, Li X, Zheng A et al (2020) Predict Ki-67 positive cells in H & E-stained images using deep learning independently from IHC-stained images. *Front Mol Biosci*. <https://doi.org/10.3389/fmolb.2020.00183>
- Lu X, Zhang S, Liu Z et al (2022) Ultrasonographic pathological grading of prostate cancer using automatic region-based Gleason grading network. *Comput Med Imag Graph* 102:125. <https://doi.org/10.1016/j.compmedimag.2022.102125>
- Mungle T, Tewary S, Arun I et al (2017) Automated characterization and counting of Ki-67 protein for breast cancer prognosis: a quantitative immunohistochemistry approach. *Comput Methods Program Biomed* 139:149–161. <https://doi.org/10.1016/j.cmpb.2016.11.002>
- Negahbani F, Sabzi R, Bea Pakniyat Jahromi (2021) Pathonet introduced as a deep neural network backend for evaluation of Ki-67 and tumor-infiltrating lymphocytes in breast cancer. *Sci Rep* 11(1):8489
- Neubeck A, Van Gool L (2006) Efficient non-maximum suppression. In: *Proceedings of the 18th international conference on pattern recognition—volume 03*. IEEE Computer Society, USA, ICPR '06, pp 850–855. <https://doi.org/10.1109/ICPR.2006.479>
- Niazi M, Tavolara T, Arole V et al (2018) Identifying tumor in pancreatic neuroendocrine neoplasms from ki67 images using transfer learning. *PLoS ONE*. <https://doi.org/10.1371/journal.pone.019562>

- Nielsen TO, Leung SCY, Rimm DL et al (2020) Assessment of Ki67 in breast cancer: updated recommendations from the international Ki67 in breast cancer working group. *J Natl Cancer Inst* 113(7):808–819. <https://doi.org/10.1093/jnci/djaa201>
- Paszke A, Gross S, Massa F et al (2019) Pytorch: an imperative style, high-performance deep learning library. In: Wallach H, Larochelle H, Beygelzimer A et al (eds) *Advances in neural information processing systems* 32. Curran Associates, Inc., pp 8024–8035
- Pedregosa F, Varoquaux G, Gramfort A et al (2011) Scikit-learn: machine learning in Python. *J Mach Learn Res* 12:2825–2830
- Redmon J, Farhadi A (2018) Yolo v3: an incremental improvement. [arXiv:1804.02767](https://arxiv.org/abs/1804.02767)
- Ren S, He K, Girshick R et al (2016) Faster R-CNN: towards real-time object detection with region proposal networks. [arXiv:1506.01497](https://arxiv.org/abs/1506.01497)
- Robbins H, Monro S (1951) A stochastic approximation method. *Ann Math Stat* 22(3):400–407. <https://doi.org/10.1214/aoms/1177729586>
- Ronneberger O, Fischer P, Brox T (2015a) U-net: convolutional networks for biomedical image segmentation. In: *International conference on medical image computing and computer-assisted intervention*. Springer, New York, pp 234–241
- Ronneberger O, Fischer P, Brox T (2015b) U-net: convolutional networks for biomedical image segmentation. <https://doi.org/10.48550/ARXIV.1505.04597>
- Rueden CT, Schindelin J, Hiner MC et al (2017) ImageJ 2: ImageJ for the next generation of scientific image data. *BMC Bioinform*. <https://doi.org/10.1186/s12859-017-1934-z>
- Saha M, Chakraborty C, Arun I et al (2017) An advanced deep learning approach for Ki-67 stained hotspot detection and proliferation rate scoring for prognostic evaluation of breast cancer. *Sci Rep* 7(3213):2045–2322
- Schneider CA, Rasband WS, Eliceiri KW (2012) NIH image to ImageJ: 25 years of image analysis. *Nat Methods* 9:671–675. <https://doi.org/10.1038/nmeth.2089>
- Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. <https://doi.org/10.48550/ARXIV.1409.1556>
- Smith B, Hermesen M, Ravichandar LDE et al (2020) Developing image analysis pipelines of whole-slide images: pre- and post-processing. *J Clin Transl Sci* 5(1):e38
- Srinidhi CL, Ciga O, Martel AL (2021) Deep neural network models for computational histopathology: a survey. *Med Image Anal* 67(101):813
- Szegedy C, Liu W, Jia Y et al (2014) Going deeper with convolutions. [arXiv:1409.4842](https://arxiv.org/abs/1409.4842)
- Vos KD. Cell counter. <https://imagej.nih.gov/ij/plugins/cell-counter.html>. Accessed 12 Jul 2022
- Wang CY, Liao HYM, Yeh IH et al (2019) CSPNET: a new backbone that can enhance learning capability of CNN. <https://doi.org/10.48550/ARXIV.1911.11929>
- Wang Y, Acs B, Robertson S et al (2022) Improved breast cancer histological grading using deep learning. *Ann Oncol* 33(1):89–98
- Wei DM, Chen WJ, Meng RM et al (2018) Augmented expression of Ki-67 is correlated with clinicopathological characteristics and prognosis for lung cancer patients: an up-dated systematic review and meta-analysis with 108 studies and 14,732 patients. *Respir Res* 19(1):150
- Xing F, Su FAU, Neltner J et al (2014) Automatic Ki-67 counting using robust cell detection and online dictionary learning. *IEEE Trans Biomed Eng* 61(3):859–70. <https://doi.org/10.1109/TBME.2013.2291703>
- Xing F, Cornish TC, Bennett TD et al (2019) Pixel-to-pixel learning with weak supervision for single-stage nucleus recognition in Ki67 images. *IEEE Trans Biomed Eng* 66:3088–3097

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