A Python application programming interface for accessing Philips iSyntax whole slide images

Nita Mulliqi[1], Lars Egevad[2], Pekka Ruusuvuori[3,4], Martin Eklund[1], Kimmo Kartasalo[1,3]

1. Department of Medical Epidemiology and Biostatistics, Karolinska Institutet, Stockholm, Sweden.
2. Department of Oncology and Pathology, Karolinska Institutet, Stockholm, Sweden.
3. Faculty of Medicine and Health Technology, Tampere University, Tampere, Finland.
4. Institute of Biomedicine, Cancer Research Unit and FICAN West Cancer Centre, University of Turku and Turku University Hospital, Turku, Finland.

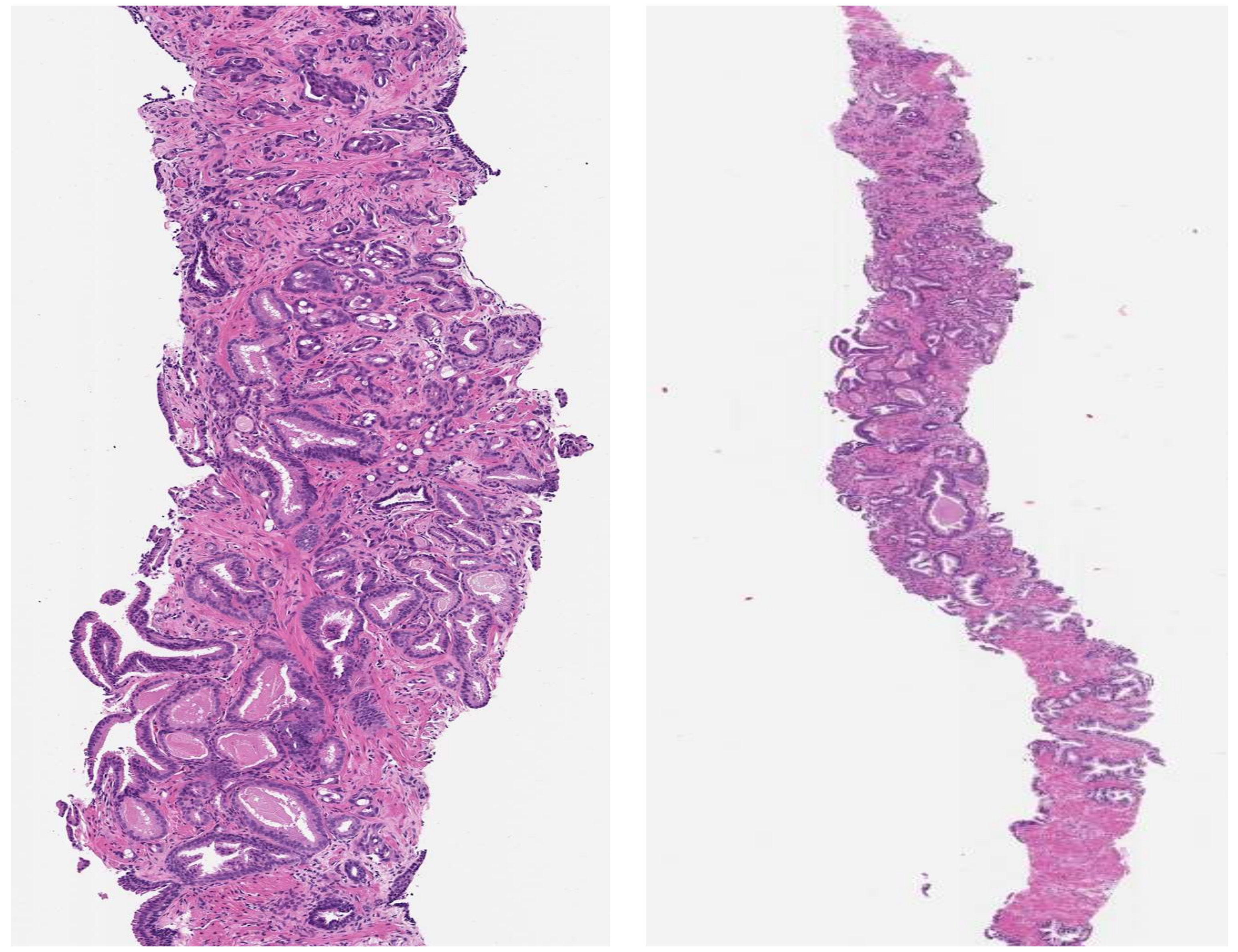

Figure 1. Examples of WSIs.

Introduction

Digital pathology enables utilization of Artificial Intelligence for improving diagnosis and prognosis of various diseases and reducing inter and intra-observer variability among pathologists. Glass slides scanned at minimally 20x lens magnification produce multi-gigapixel Whole Slide Images (WSI) **(Fig. 1)**. The size of WSIs makes their interoperability, management and storage very difficult.

Scanners from different vendors store WSIs in different proprietary image formats (DICOM, Aperio SVS, Hamamatsu NDPI/VMS/VMU, Leica SCN, 3DHisTech MRXS, Philips iSyntax etc.) with corresponding libraries, tools and viewers. Vendor-neutral libraries such as OpenSlide lack support for the iSyntax format used by the Philips IntelliSite scanner, gaining in popularity due to FDA approval.

Philips recently released a Software Development Kit (SDK) for accessing iSyntax, however this represents a relatively low-level interface requiring familiarity with the iSyntax format itself.

Methodology

Our Application Programming Interface (API) enables access to the label image, low-resolution macro image of the slide and the WSI scanned at high resolution **(Fig. 2)**.

The read_region method extracts pixel data from a rectangular region in the WSI at a desired resolution level. It provides easy random access at different location and resolution levels for any downstream applications typically requiring limited amount of pixels at any given time. The get_thumbnail and read_wsi methods read the entire WSI at maximum dimensions or a specified resolution, respectively. This is typically performed to obtain a more detailed view of the entire scanned area than the macro image.

DICOM compliant metadata attributes are extracted as DICOM tags. All other metadata information are provided adhering to the OpenSlide generic properties format. The names and input parameters of the methods adhere to the OpenSlide API, allowing developers to use existing code developed on top of OpenSlide without modifications.

Availability and implementation

Our API is developed on Ubuntu 18.04 and Python 3.6.9, and it is available as an open-source software under the MIT license at: https://gitlab.com/BioimageInformaticsGroup/openphi.

The Philips SDK is required and available at: https://www.openpathology.philips.com.

Results and Conclusions

Images used for testing the API are scanned needle biopsies from patients in the prospective Stockholm-3 study (around 50,000 biopsies stored as iSyntax). During the experiments, we verified that our API can successfully interface the iSyntax format with a previously developed artificial intelligence system for Gleason grading.

The presented API serves as a powerful interface between algorithm developers and the iSyntax format. On top of the API, developers can easily build algorithms while spending minimal effort on dealing with the proprietary format or modifying existing applications.

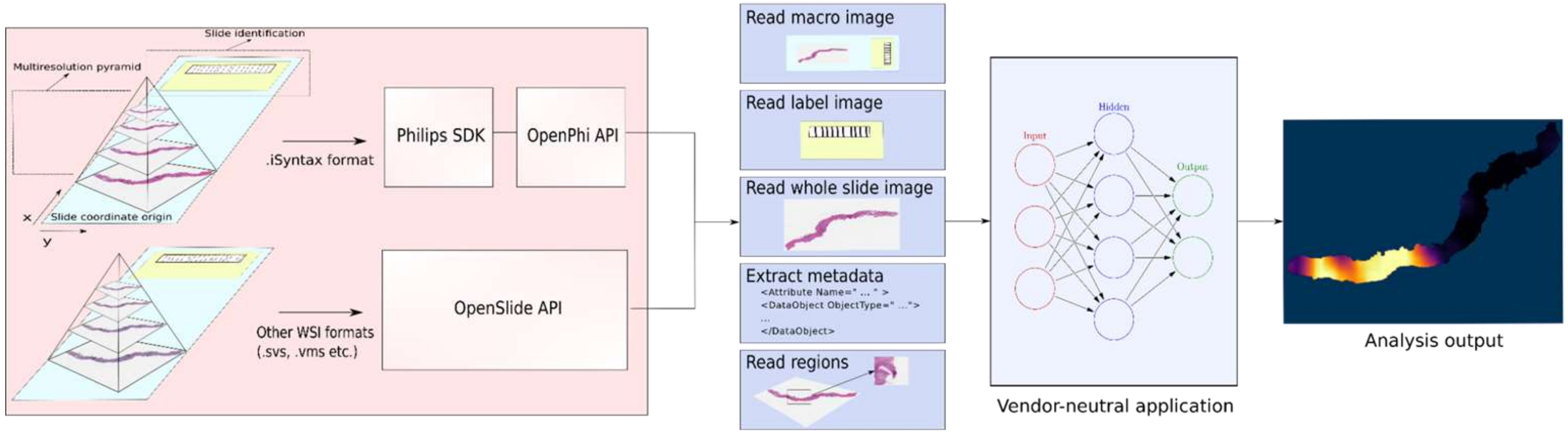

Figure 2. Left: WSIs contain: label image for sample identification, macro image of the glass slide, and multiresolution pyramid of the high-resolution pixel data. Our API provides a compatibility layer, facilitating access to iSyntax WSIs via the Philips SDK, while adhering to a unified API consistent with OpenSlide. Middle: Streamlined access to the images and metadata is provided to developers by a set of high-level methods. Right: Applications capable of analyzing WSIs in vendor-neutral manner can be built relying on the unified API.

Karolinska Institutet
Department of Medical Epidemiology and Biostatistics
Nobels väg 12, Solna, Sweden

Nita Mulliqi
Doctoral student
Nita.Mulliqi@ki.se