Using iSyntax Whole Slide Image Format in Research/ML

Alexandr Virodov, MS[1], Cody Bumgardner, PhD[1]

[1]University of Kentucky, Lexington, KY, USA.

Introduction

ISyntax is a whole-slide image file format produced by Philips whole-slide scanners [1]. As part of the first FDA-approved scanner for clinical diagnosis, the iSyntax format is widely used. However, there is no support for this format in commonly used open-source image libraries for Digital Pathology (OpenSlide, BioFormats, CuCim). This hinders the application of digital pathology (slide viewing, Machine Learning) at sites that use the Philips scanner, resulting in millions of scanned slides that are not available outside of clinical use.

Current approaches to tap this large data repository rely on either utilizing the Philips SDK (OpenPhi[4], in-house solutions), or conversion to other formats, such as OME-TIFF [3]. The Philips SDK has a closed source and is difficult to obtain due to the individual licensing requirements. The conversion process is unfortunately slow, storage-intensive, and unavoidably creates duplication of data. Additionally, there is no single conversion process that can be used across all libraries: the OME-TIFF is not supported by OpenSlide, and the generic TIFF is not supported by the BioFormats library.

We present a native integration of an open source iSyntax viewer [2] with OpenSlide. This enables us to use our iSyntax slides in slide viewing application with DSA (Digital Slide Archive) [5] and in ML applications with MONAI [6]. As we release the implementation to the open-source community, this will enable researchers across the world to access a large number of already scanned slides available at their institutions.

Methods

Goal: Support iSyntax for two usecases: Slide viewing and Machine Learning (ML). Slide viewing usually implies realtime intensive work with one or few slides, and Machine Learning implies non-realtime batch processing of 100s or 1000s of slides, often in parallel.

Decoder implementation: We used the open source iSyntax decoder implemented in Slidescape viewer [2]. It supports XML header parsing, tile decoding, and tile management for the viewing usecase.

Image library: we evaluated a few possible candidates for integration: OpenSlide, BioFormats, CuCim. OpenSlide is implemented in C with several bindings to other languages, most importantly Python. It is supported as the image reader for the Digital Slide Archive slide viewer, and as a backend for the slide reader in MONAI [6] and other ML code we use. Since both OpenSlide and Slidescape are written in C, it provided an easy integration path, while enabling both usecases of interest. Thus we chose OpenSlide as our first integration target.

Deidentification: We have deidentification tools that operate on the XML portion of the iSyntax file. This preserves the iSyntax header size, allowing for image data deduplication between identified and de-identified slides at the filesystem level.

Evaluation: We evaluated: 1. runtime performance - synthetic and real usecases. 2. peak memory consumption, as it will limit the number of parallel workers on a multi-core machine. 3. Image quality, in reference to the Philips SDK. We evaluated the conversion process to Tiff at different Jpeg compression levels, and selected 95% quality as our reference due to that compression level roughly matching the original iSyntax slides in size.

Baseline:

Philips SDK Conversion (Tiff Jpeg Quality)	**80%**	**90%**	**95%**	**100%**
Time to convert 10 slides	32m 9s	29m 48s	31m 58s	40m 26s
Additional storage for 10 converted slides (iSyntax size: 9.4 Gb)	4.7 Gb	6.9 Gb	12 Gb	27 Gb
Additional storage as percent of iSyntax size	50%	73%	128%	287%

Results

Access 100 slides	Time Tiff 95% (s)	Memory Tiff 95% (Mb)	Time Ours (s)	Memory Ours (Mb)	Time Ours / Tiff
Full Sequential read of last level	24,264 (6.74 hrs)	3,311	48,816 (13.5hrs)	7,040	2.01
Random single tile read of last level	0.24	210	3.94	7,057	16.55
Random single tile read of middle level	0.22	158	1.99	6,919	8.99
Random 100 tile read of last level	18.75	3,311	235.32	10,181	12.55
Random 100 tile read of middle level	11.71	217	5.33	7,048	0.45
Monai-MIL [7]	4081	161,698	4810	95,946	1.17

Image Quality vs Philips SDK loseless PNG	Tiff 80%	Tiff 90%	Tiff 95%	Tiff 100%	Ours
PSNR (Higher is better)	41.43	43.35	44.85	48.65	47.83

Discussion

The presented approach to accessing iSyntax slides is not as fast or memory efficient as accessing a converted Tiff. However, there are scenarios where that is not a limiting factor. In ML applications, the GPU memory and compute are major limiting factors, and thus the long read times or high cpu memory usage are not significantly limiting performance. In viewing applications, due to low throughput of human viewing and random access, it may be more efficient to access slides slightly slower rather than pay an up-front conversion cost and storage cost for a large slide library.

The image decoding implementation does not match bitwise with the SDK, which is reflected in the PSNR. Currently bit-precise decoding is not a goal for us. There are other errors, such as an occasional blue-colored pixel probably due to out-of-bounds values in YCoCb to RGB conversion. Those will be investigated further.

Some of the limitations of the performance comparison: 1. Using only a single process for conversion. With multiprocessing, horizontal scaling is easily achievable for Philips SDK conversion. 2. We use non-sparse Tiff for conversion, and Philips SDK is trimming missing tiles on the lower right corner of the slide. Our code can generate missing tiles efficiently, but does not trim the lower right corner. Thus the number and kind of tiles read in Full Sequential Read does not match precisely. However, it is still representative for e.g. Tissue Detection application in pathology.

Future work: 1. Reducing per-slide memory consumption by evicting internal structures, if needed. 2. Better multithreading support. 3. Runtime performance improvements.

References

[1] Bas Hulsken, Philips' iSyntax for Digital Pathology.
https://thepathologist.com/fileadmin/issues/App_Notes/0016-024-app-note-Philips__iSyntax_for_Digital_Pathology.pdf

[2] Pieter Valkema, Slidescape. https://github.com/amspath/slidescape

[3] Melissa Linkert, Chris Allan, Converting Whole Slide Images to OME-TIFF: A New Workflow.
https://www.glencoesoftware.com/blog/2019/12/09/converting-whole-slide-images-to-OME-TIFF.html

[4] Nita Mulliqi, Kimmo Kartasalo, Henrik Olsson, Xiaoyi Ji, Lars Egevad, Martin Eklund, Pekka Ruusuvuori, OpenPhi: an interface to access Philips iSyntax whole slide images for computational pathology, Bioinformatics, Volume 37, Issue 21, 1 November 2021, Pages 3995–3997, https://doi.org/10.1093/bioinformatics/btab578

[5] Gutman DA, Khalilia M, Lee S, Nalisnik M, Mullen Z, Beezley J, Chittajallu DR, Manthey D, Cooper LAD. The Digital Slide Archive: A Software Platform for Management, Integration, and Analysis of Histology for Cancer Research. Cancer Res. 2017 Nov 1;77(21):e75-e78. doi: 10.1158/0008-5472.CAN-17-0629. PMID: 29092945; PMCID: PMC5898232.

[6] The MONAI Consortium. (2020). Project MONAI. Zenodo. http://doi.org/10.5281/zenodo.4323059

[7] Andriy Myronenko, Ziyue Xu, Dong Yang, Holger Roth, Daguang Xu, Accounting for Dependencies in Deep Learning Based Multiple Instance Learning for Whole Slide Imaging. https://doi.org/10.48550/arXiv.2111.01556