Dealing with I/O bottlenecks in high-throughput model training

Jan Sellner[1,2,3], Silvia Seidlitz[1,2,3,4], Lena Maier-Hein[1,2,3,4]

[1] Division of Intelligent Medical Systems, German Cancer Research Center (DKFZ), Heidelberg, Germany
[2] Helmholtz Information and Data Science School for Health, Karlsruhe/Heidelberg, Germany
[3] Faculty of Mathematics and Computer Science, Heidelberg University, Heidelberg, Germany
[4] National Center for Tumor Diseases (NCT) Heidelberg, a partnership between DKFZ and Heidelberg University Hospital, Heidelberg, Germany
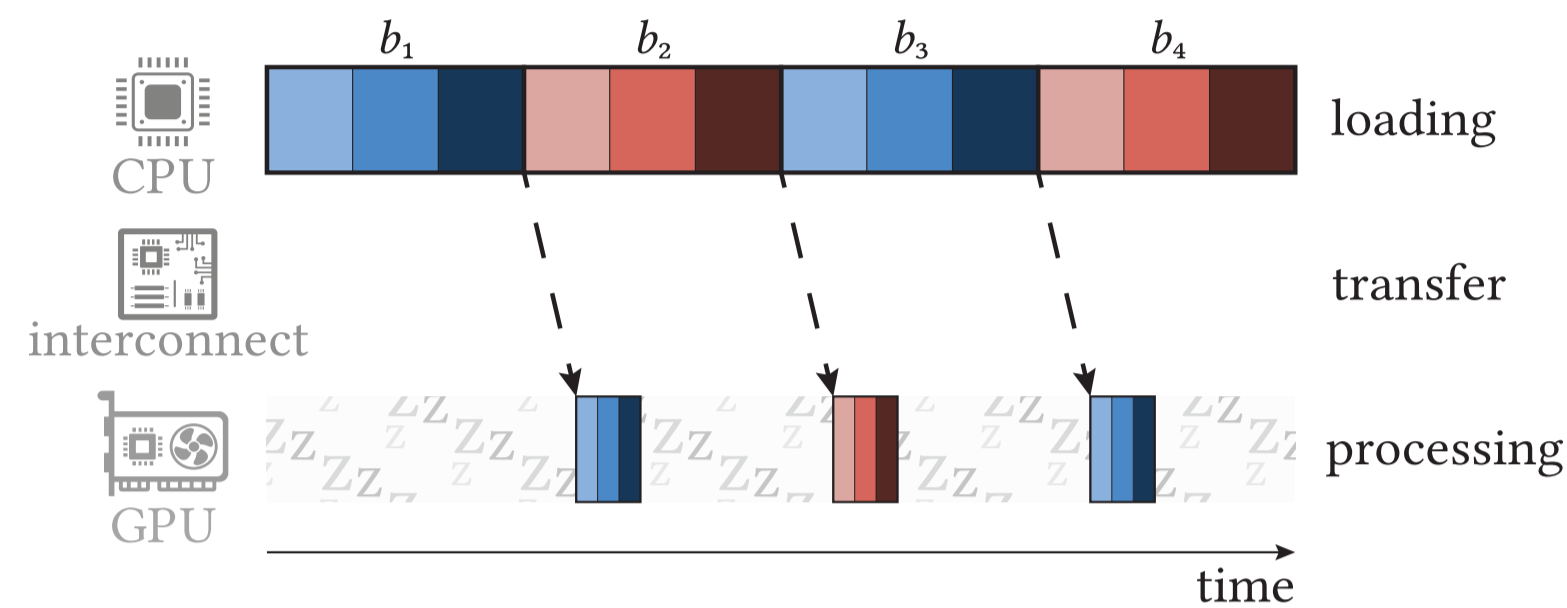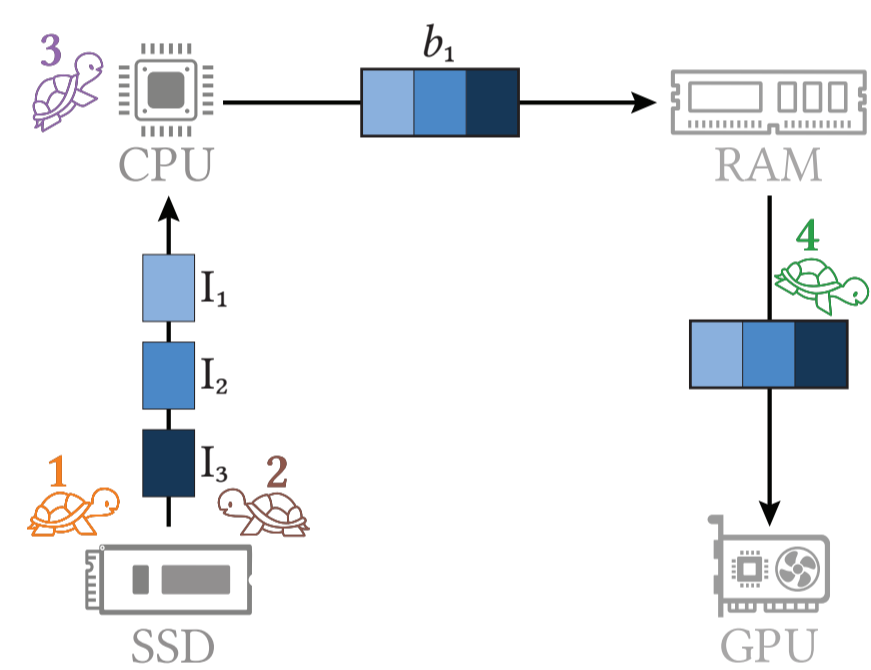
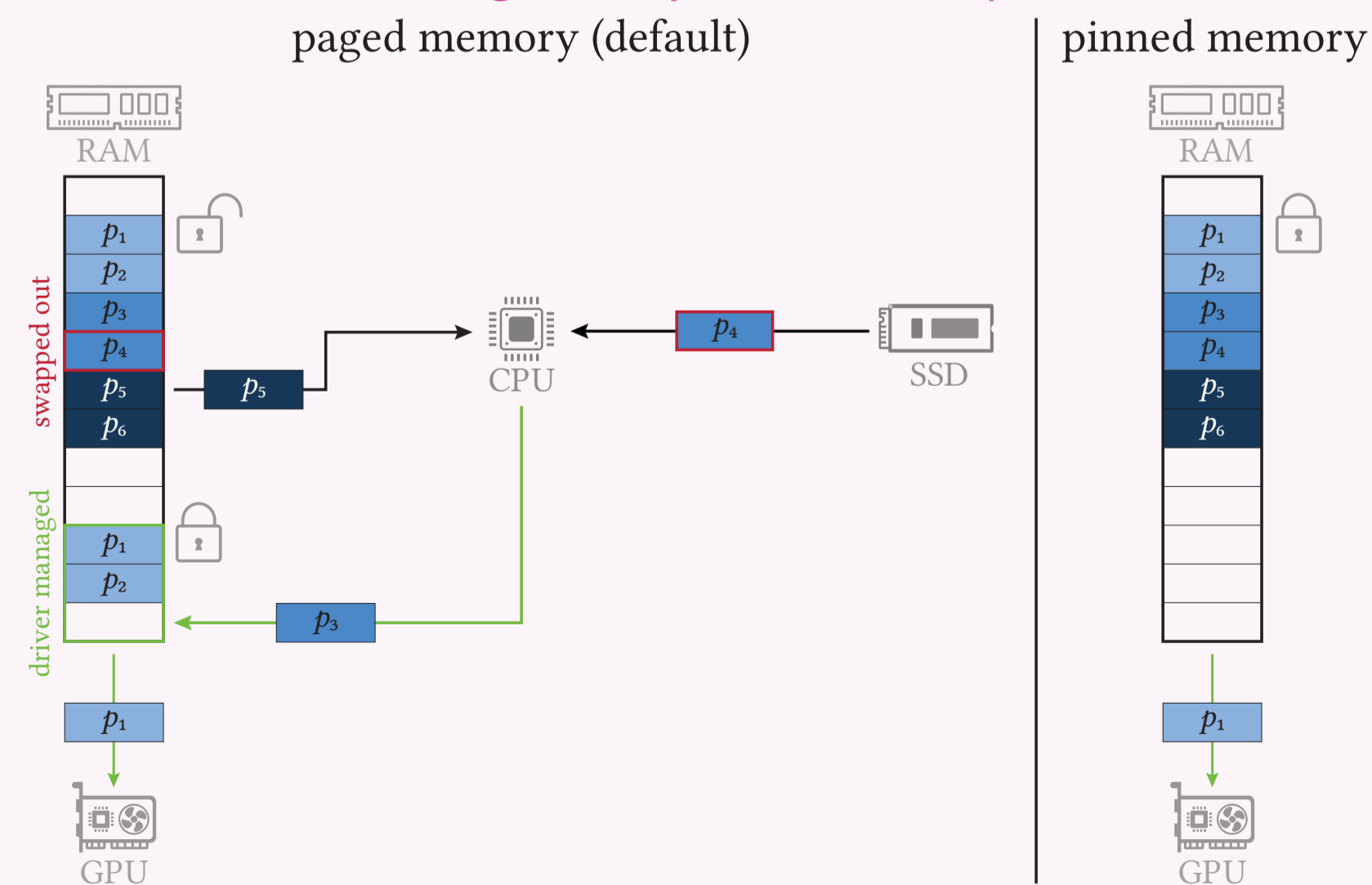PyTorch Conference 2023

Data loading bottlenecks

In high-throughput model training, the GPU is subject to idle times if the CPU doesn't deliver the data in time (CPU- and I/O-bound training).

Several bottlenecks retard the data loading from the SSD to the GPU.

Background: pinned memory

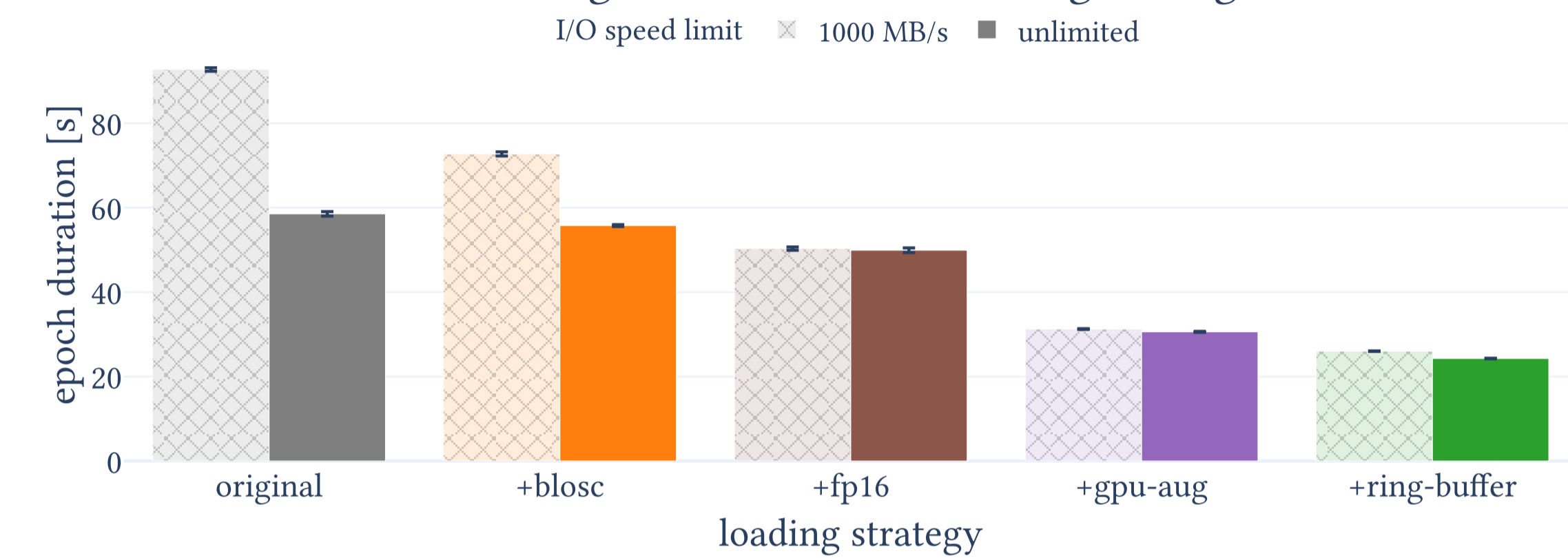paged memory (default) | pinned memory

With pinned memory, data can be moved directly from the RAM to the GPU since it is ensured that the memory pages aren't swapped out by the OS. This avoids unnecessary data transfers.

Independent counter-strategies

file size — loading effort

0: original

Data samples of shape (480, 640, 100) = (height, width, channels) with float32 values.

1: +blosc

BLOSC

Storing data compressed on disk reduces disk load. An efficient decompresser (e.g. BLOSC[1]) should be used to avoid long decompression times.

2: +fp16

`dtype=torch.float16`

Minimizing input data precision whilst maintaining its information content reduces loading times from disk and transfer times to the GPU.

3: +gpu-aug

`device='cuda'`

If the CPU is the bottleneck, moving computations to the GPU (e.g. via Kornia[2]) frees resources on the CPU.

4: +ring-buffer

worker1 — I_9
worker2 — I_{11}

b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8

GPU

shared, pinned memory

Using a fixed (initialized on training start), shared (accessible by all workers) and pinned memory buffer avoids unnecessary memory transfers.
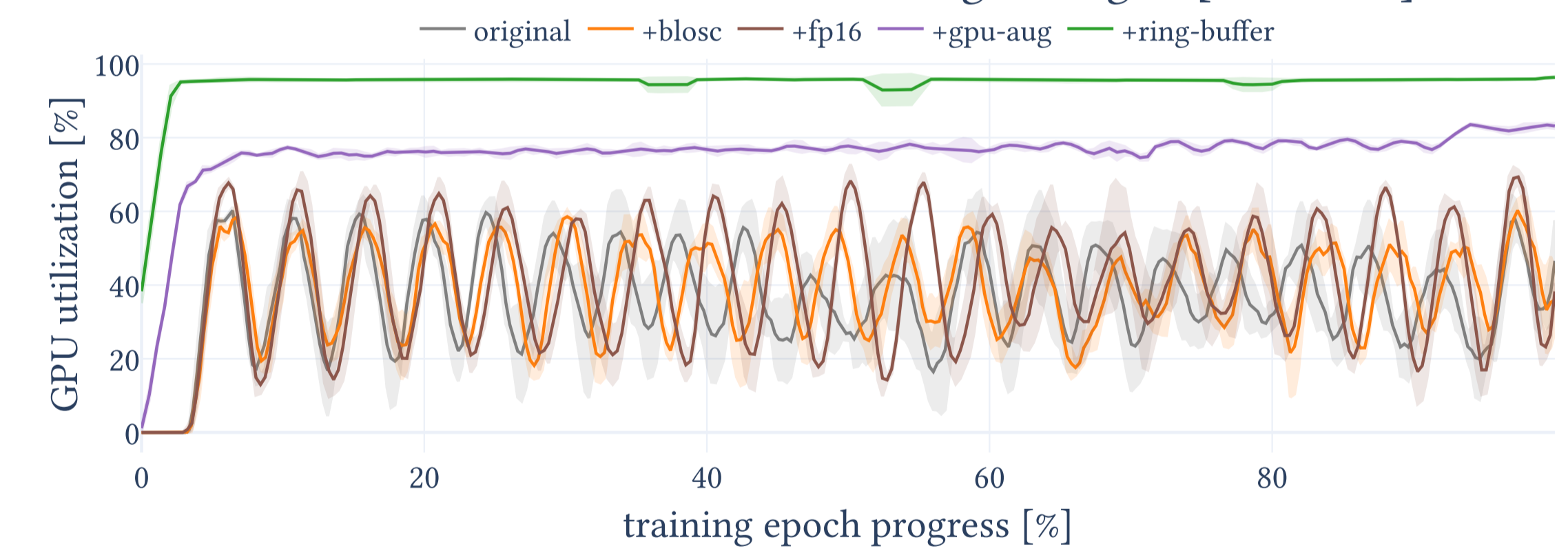
Results

Benchmarking of different data loading strategies

I/O speed limit: 1000 MB/s — unlimited

epoch duration [s] vs loading strategy (original, +blosc, +fp16, +gpu-aug, +ring-buffer)

Through the presented counter-stragegies, I/O and CPU load is reduced and training is up to 3.6 times faster.

GPU utilization of different data loading strategies [unlimited]

original — +blosc — +fp16 — +gpu-aug — +ring-buffer

GPU utilization [%] vs training epoch progress [%]

If data is not delivered in time, GPU utilization is suboptimal. In these cases, alternating loading and waiting times can be observed.

Application: Hyperspectral Imaging (HSI) in surgery

HSI camera → hyperspectral image (HSI) → RGB image

Postoperative death is the third leading cause of death as it accounts for 7.7 % of all deaths globally.

Hyperspectral **T**issue **C**lassification

Hyperspectral Imaging extends the human vision by removing the arbitrary restriction of capturing only red, green and blue colors. Instead, a fine-grained reflectance spectrum is recorded, providing additional information about tissues and their functional properties (e.g. oxygenation).