NeuroKube: An Automated and Autoscaling Neuroimaging Reconstruction Framework using Cloud Native Computing and A.I.

Matthew Madany
University of California, San Diego
La Jolla, CA, USA
madany@ucsd.edu

Kyle Marcus
University of California, San Diego
La Jolla, CA, USA
krmarcus@ucsd.edu

Steven Peltier
University of California, San Diego
La Jolla, CA, USA
speltier@ucsd.edu

Mark H. Ellisman
University of California, San Diego
La Jolla, CA, USA
ORCID iD: 0000-0001-8893-8455

Ilkay Altintas
University of California, San Diego
La Jolla, CA, USA
ialtintas@ucsd.edu

***Abstract*—The Neuroscience domain stands out from the field of sciences for its dependence on the study and characterization of complex, intertwining structures. Understanding the complexity of the brain has led to widespread advances in the structure of large-scale computing resources and the design of artificially i ntelligent a nalysis s ystems. H owever, t he s cale of problems and data generated continues to grow and outpace the standards and practices of neuroscience. In this paper, we present an automated neuroscience reconstruction framework, called *NeuroKube*, for large-scale processing and labeling of neuroimage volumes. Automated labels are generated through a machine-learning (ML) workflow, w ith d ata-intensive steps feeding through multiple GPU stages and distributed data locations leveraging autoscalable cloud-native deployments on a multi-institution Kubernetes system. Leading-edge hardware and storage empower multiple stages of machine-learning, GPU-accelerated solutions. This demonstrates an abstract approach to allocating the resources and algorithms needed to elucidate the highly complex structures of the brain. We summarize an integrated gateway architecture, and a scalable workflow-driven segmentation and reconstruction environment that brings together image big data with state-of-the-art, extensible machine learning methods.**

***Index Terms*—Neuroscience, Segmentation, Superresolution, Machine Learning, Workflow, G ateway, B ig D ata, Kubernetes**

I. Introduction

The Neuroscience domain stands out from the field of sciences for its dependence on the study and characterization of complex, intertwining structures. Biologists broadly use large-data techniques to sequence and analyze the genes and molecules of living organisms, integrating techniques from across the omics spectrum. While neuroscience also depends on these methods, the greater understanding of cognition and disease comes from the structures and distributions of cells in the brain. However, brain cells such as neurons and glia form massively complex networks, projections, and highly organized subcellular systems to form the central nervous system. The instrumentation and methodology required to resolve and analyze these structures is notoriously difficult, harboring some of the largest and most complex data challenges when it comes to translating image data to research impact.

Volume electron microscopy (VEM) methods are commonly used to reconstruct samples of brain with enough resolution to resolve the fine processes, distributions of structures, and the connections between neurons, synapses, while also harboring the throughput to image extents and fields large enough to contain populations and circuits of neurons [1]. This approach is comprehensive with one major barrier: the creation of datasets with exponential footprints on hardware storage systems and data-specific challenges that make the design of management systems and analysis frameworks inaccessible [2]. As a result, the study of micro-, meso-, macroconnectomics, and projectomics has remained largely limited to a handful of institutes worldwide, with studies from long-tail laboratories limited to the characterization of fragments, samples, and subsets of brain cells with analysis extents insufficient to contain whole neurons [3], [4]. Neuroscience has remained one the world's most least understood domains with the throughput needed to understand the structure-function relationship of the brain, cognition and neurodegenerative disease, limited by the size and complexity of data [5], [6]. A rethinking of the hardware-software relationship is needed in order to carry this data through meaningful and scalable computations.

The Pacific Research Platform (PRP) [7] and CHASE-CI [8] together provide a purpose-built hardware and software ecosystem spanning multiple institutions with the mission of solving the big data challenges presented by numerous sciences. The three foremost features are:

- 10Gb/s to 100Gb/s disk-to-disk connectivity,
- 500+ GPU-accelerated node distribution,
- Fully decentralized high-bandwidth storage infrastructure, and
- Kubernetes [9] engine for fully cloud-native development.

Contributions. In this paper, we present an automated neuroscience reconstruction framework, called *NeuroKube*, that builds on top of these capabilities to provide a community environment for large-scale processing and labeling of brain images. Automated labels are generated through a machine-learning (ML) workflow, with data-intensive steps feeding through multiple GPU stages and distributed data locations. Instead of relying on traditional data-center driven computes, we leverage the CHASE-CI's distributed node hardware and PRP's high-bandwidth, interconnected, optic fiber network to transiently deploy storage and input locations in a way that can be dynamically rotated onto requisite processing, visualization, and development interfaces. We summarize an integrated gateway architecture, and a scalable workflow-driven segmentation and reconstruction environment that brings together image big data with state-of-the-art, extensible machine learning methods. In stark contrast to existing HPC-based solutions, this platform offers: (i) Fully interface-able, cloud-hosted, software for data visualization; (ii) Auto-scaling execution solutions for parallel processing of reconstruction workflows; (iii) Interoperable filesystems designed for specific I/O data challenges; and (iv) A web-based community interface for experiment management. Through a thorough deployment of NeuroKube cloud-native services to the PRP, large neuroscience data is demonstrably more accessible to both machine intelligence and users.

Outline. The rest of this paper is organized as follows. In Section 2, we introduce the new team science and a reference architecture to make it effective. Section 3 introduces the NeuroKube framework architecture on top of CHASE-CI, and the NeuroKube Gateway. In Section 4, we introduce the overview the labeling and segmentation methods implemented in NeuroKube. Section 5 provides a summary of our experimental results and provides a discussion of the performance of the segmentation methods on large image data. We review related work in Section 6 and conclude in Section 7.

II. Comprehensive Segmentation of Different Brain Regions

The brain is composed primarily of neurons and glial cells interlacing through the neuropil. Computer vision is needed to resolve several hierarchies of information regarding the structures, distribution, and subcellular content of these cells. Computer vision-based approached also provide the potential to automatically generate labels using ML (see Figure 1). A challenge of ML-driven electron microscopy (EM) image labeling is that the volumes are too large to be hosted in memory, and must be partitioned and windowed in pre-processing and merged in post-processing. The first neural networks to infer biological structures were 2D, and 3D volumes were tiled into 2D squares. Recent advances in ML architectures have enabled 3D neural networks with height, width, as well as depth, to drive higher accuracy measurements in VEM data [10]. While this approach has elevated the fidelity of ML analysis to make real and accurate structural analysis of biological tissue, it has the notable drawback of necessitating that the windowing and partition process be likewise volumetric. ML models underperform at the edges of their viewpoints, so windows have to overlap to some degree and be blended back into outputs to remove checkerboarding artifacts. The use of permutations gives ML models the opportunity to view data at different viewpoints in order to merge these predictions together in ensemble techniques to further drive accuracy improvements. The combination of permutations and volumetric windowing makes the internal application design very data-intensive. A further challenge is that large image volume inputs are 8-bit grayscale data, a data type specific to reducing the storage footprint of large volumes. Outputs are semantic, object, and instance segmentations with bit types required to encompass a vast number of labels.

One popular way to describe the challenges affiliated with volumetric processing is the curse of dimensionality [11]. Not only does this affect application design, but it can limit the ability of machine intelligent systems to conceptualize data and objects at scale. For example, volumetric windows applied to brain VEM data must be sampled at the resolution required to resolve synapses and their components, however these objects may not be present in every window applied during training and inference, which can result in underperforming automated labeling schemes [12]. A recent advance to address these issues is the introduction of multi-label classification schemes in which single neural networks are trained to interpret the separation of objects in a scene [13]. While this is meaningful in the ongoing sophistication of automated labeling, the challenge is always the same. The data gets bigger.

In addition to multi-label schemes, ML workflows can be orchestrated to resolve the individual cellular regions of images. Since the brain is made up of intertwined, fine processes of a vast number of cells in 3D, this requires that bit types are high enough to encode an extremely high number of region labels.

VEM data is also anisotropic, with differential resolutions across axial planes. Much work has gone into the engineering of microscopes capable of acquiring isotropic resolutions but the overwhelming amount of VEM data has high XY resolutions with Z resolutions limited by microscope type [14]. Recent advances in Generative Artificial Intelligence (AI) techniques have shown that resolution information can be translated across microscopy modes or axial planes to improve limited resolutions from specific equipment [15]. While this technique improves the acuity along the planes at which volumetric microscopes operate, it increases data size by whatever factor of isotropy is targeted. A sample with a $6nm$ x $6nm$ x $60nm$ pixel size would therefore be 10 times larger when translated to a $6nm$ x $6nm$ x $6nm$ volume. This process can be attenuated by targeting an improved axial resolution, such as $6nm$ x $6nm$ x $30nm$, in which case the data size would have doubled. Taking all data-increasing factors into account, consider the analysis of an volume of arbitrary size 100x100x100 in the brain. Say that the axial resolution on Z must be improved by at least a factor of two due to microscopy mode limitations. In order to give the ML model sufficient viewpoints, a total of 4 permutations is made, with a window overlap percentage of 25. Therefore, we would expect data size during application scaling and deployment to be

$$X \text{ x } Y \text{ x } Z \text{ x } Zs \text{ x } Zw \text{ x } P \text{ x } O \text{ x } C \text{ x } B$$

where X, Y, and Z are the dimensions of the volume, Zs is the upscaling factor targeted by a superresolution generator, Zw is the z-depth of windows, P is number of permutations, O is the percentage of overlap between 3D windows, C is the number of classes to resolve each structure of interest, and B is the higher bit type for the instance segmentation data. While the initial volume would have contained an already large number of voxels at one million, an estimation for data size during execution would be 350 times higher than the input of 100 x 100 x 100.

Not only would this data be extremely large, but it would be flowing through GPU accelerations, limited by VRAM, meaning that large-RAM/CPU node configurations would be unable to resolve runtime costs. ML Applications must be designed to resolve as many of the data challenges as possible during execution in order to produce rational intermediate and output storage footprints. Permutations can be blended at runtime, block-wise overlaps can be merged upon application end, high-bit data can be compressed, label data can be downsampled, and all can be stored in specialized concurrent data structure schemes. Even assuming all of these challenges are addressed, what if there is interest in multiple datasets from different parts of the brain? Tolerating multiple runs would further add to dimensions of the problem. What if the extent should be doubled on $XxYxZ$ to contain additional structures of interest? Doubling the $XxYxZ$ dimensions would mean 8 times more information flowing through GPU accelerators. VEM data is already one of the largest data producers in the world [16].

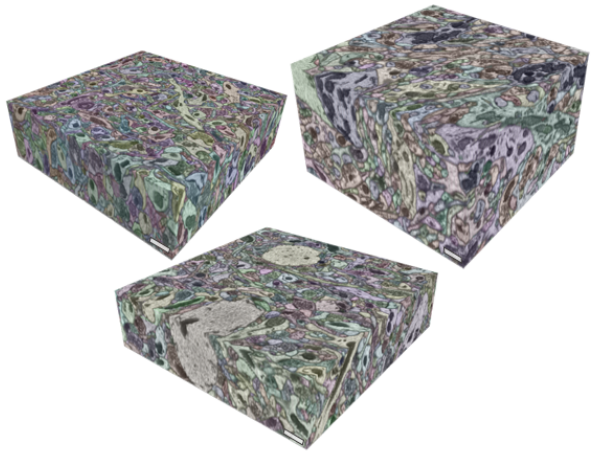


Figure 1. Volumetric electron microscopy (VEM) data with colorized labels. (Scale Bar $2\mu m$)

III. NeuroKube Framework and Gateway

NeuroKube is a framework for the analysis of large image volumes for neuroscience research. While contemporary computing approaches rely on purpose-built servers and hardware to interconnect with HPC resources predicated on the transfer of data from server to cluster [17], *NeuroKube* rotates users and resources directly onto data. Application streaming, workflow development, and database integration execute with transient compute resources dynamically allocated at runtime based on operational needs. From the application of ML techniques on large image volumes to the visualization of teravoxel information, threads and memory are made available as-needed, based on rapid and scalable storage systems. This was created using cloud-native development, a type of systems engineering that combines high-performance computing, cloud development, and application environment scaling.

Kubernetes, is used in place of contemporary scheduling systems for deployment of applications and services to drive a global neuroinformatic scheme [9]. Cloud native computing engines like Kubernetes remove reliance on server-client hierarchies that have defined high-performance computing management systems. Instead, connections are built between serverless deployments of microservices and batch commands where resource provisions are based on abstract ranges that can be re-defined and re-allocated in real-time based on demand [18]. The computing ecosystem is provided by the PRP and CHASE-CI, a multi-institution partnership hosting GPU-enabled compute nodes connected and established under a high-bandwidth ESnet Science DMZ model. The Kubernetes engine running on the PRP is called Nautilus, and research projects are separated into team userbases called namespaces. Ceph is an open source distributed storage platform used as the backbone of both the distributed filesystem and the requestable block storage in Nautilus [19]. Ceph nodes are placed among the cluster to distribute the data to allow for redundancy and fault tolerance. This is done through Rook which is the Kubernetes operator for Ceph.

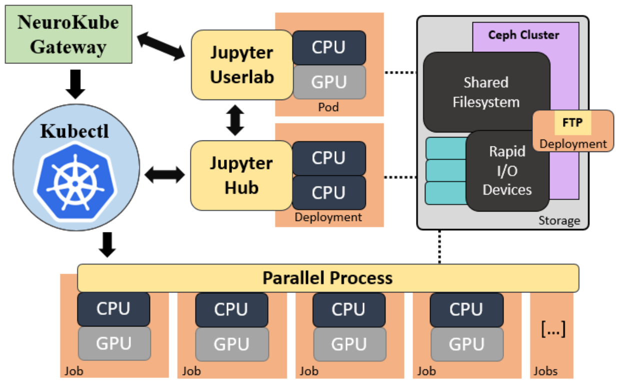
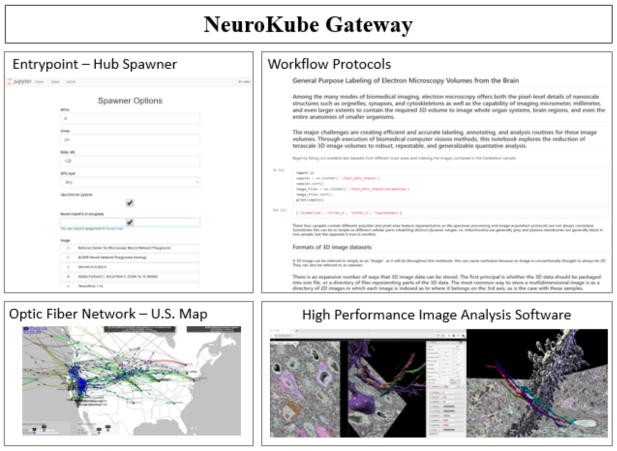


Figure 2. NeuroKube Framework Architecture

Figure 2 illustrates NeuroKube framework architecture with containerized software and application layers of the hardware and storage managed by CHASE-CI. Operating systems and software are hosted in repositories of Docker containers stored in GitLab services within the high-speed network. Docker containers are layered, and components of their constituent operating system, stored in files, are hierarchical, with additive building of containers based on a preset of built layers. When executed on Kubernetes, runtime software data is accrued as an additional layer, and when the cost of computation exceeds available local resources, the container is flashed onto a new compatible node which does possess the available request, and data links are maintained alongside runtime information to carry on the computation at the higher cost. Because Docker layers are pre-cached onto nodes, the transient transfer of containers happens rapidly across the networked infrastructure. This ‘stateless’ execution of containerized applications enables a transient automatic way to scale the virtualization applications.



General Purpose Labeling of Electron Microscopy Volumes from the Brain

Among the many modes of biomedical imaging, electron microscopy offers both the pixel-level details of nanoscale structures such as orgnelles, synapses, and cytoskleletons as well as the capability of imaging micrometer, millimeter, and even larger extents to contain the required 3D volume to image whole organ systems, brain regions, and even the entire anatomies of smaller organisms.

The major challenges are creating efficient and accurate labeling, annotating, and analysis routines for these image volumes. Through execution of biomedical computer visions methods, this notebook explores the reduction of terascale 3D image volumes to robust, repeatable, and generalizable quantative analysis.

Begin by listing out available test datasets from different brain areas and indexing the images contained in the Cerebellum sample.

```
In [1]: import os
        samples = os.listdir('./Test_Sets_Stacks')
        samples.sort()
        image_files = os.listdir('./Test_Sets_Stacks/Cerebellum')
        image_files.sort()
        print(samples)

Out [1]: ['Cerebellum', 'Cortex_1', 'Cortex_2', 'Hypothalamus']
```

These four samples contain different acquities and pixel-wise feature representation as the specimen processing and image acquisition protocols are not always consistent. Sometimes this can be as simple as different cellular parts inhabiting distinct dynamic ranges, i.e. mitochondria are generally grey and plasma membranes are generally black in one sample, but the opposite is true in another.

Formats of 3D image datasets

A 3D image can be referred to simply as an "image", as it will be throughout this notebook, this can cause confusion because an image is conventionally thought to always be 2D. They can also be referred to as volumes.

There is an expansive number of ways that 3D image data can be stored. The first principal is whether the 3D data should be packaged into one file, or a directory of files representing parts of the 3D data. The most common way to store a mutlidimensional image is as a directory of 2D images in which each image is indexed as to where it belongs on the 3rd axis, as is the case with these samples.

Figure 3. NeuroKube Gateway Components

Figure 3 diagrams the Neurokube front-end. The user first interacts with a spawner that sets minimum resource requirements: GPUs, CPUs, Memory, and Storage Types (upper-left). Workflow protocols are a collection of Jupyter notebooks that wrap around the HPC application workflow stages in a way that models can be trained and tested (upper-right). Current distribution of PRP optic fiber network (bottom-left, see traceroute.nautilus.optiputer.net). Automatically labeled brain VEM volumes in PRP-Jupyter-hosted Paintera software inferace with wiring diagrams from hypothalamus (left) and cerebellum (right).

To reduce the burden of expertise in scientific programming with low-level languages, a resource-transient application for interactive development is provided with a JupyterHub service [20] through a user gateway. Figure 3 shows the resource allocation, notebook-based workflows and image analysis software interfaces provided through the *NeuroKube Gateway*. With Kubernetes, support for deployed services allows server-side perpetual hosting of full-stack services like JupyterHub (see Figure 2). In this model, the hub spawns JupyterLab instances horizontally, inhabiting the same larger Kubernetes cluster, yet the hub requisitions some abstract allocation of resources, with 1/4th of a CPU being minimally threaded on average and running on multiple full CPUs based on real-time user demand. The downstream JupyterLab is given a user-defined minimum resource request, and further memory and threads are allocated during execution of notebooks. Within the JupyterLab instance, kernels for Python, Matlab, and R alongside the machine-learning frameworks Tensorflow, PyTorch, and Caffe create a comprehensive environment for workflow and ML development using only high-level scientific computing languages [21]. One or more GPUs can be requested to enable high-throughput training and inference of demanding model architectures and data inputs from the JupyterHub.

JupyterLab runs alongside a node.js environment where extensions can be installed that enable graphical interaction with data [22]. Image volumes and model outputs are visualized in-line with 3D slice viewers and scalable volume rendering engines, a capability that goes beyond a more commonly-used Matlab IDE for volumetric image processing [23], [24]. The markdown of notebooks and mounting of shared storage spaces creates both a scalable and shareable protocol hub for multiple users to collaborate. A NextCloud service provides a cloud sharing web service for easy-to-access data downloads of notebooks and intermediate data and a RocketChat service for a forum messaging space, all hosted on PRP/CHASE-CI [25], [26].

The Jupyter Desktop Xubuntu environment hosts an image analysis software stack. The primary and commonly-used tools installed are IMOD and ImageJ [27], [28]. The gateway also hosts *Paintera*, a large-scale, open-source, volume segmentation, validation, and visualization software that imports and formats data from ML workflows [29]. This application offers real-time mesh generation and teravoxel segment editing, however this exacts a heavy CPU cost and requires a GPU to run efficiently in real-time, making local multi-user configurations expensive.

As a summary, *NeuroKube* provides low-latency application streams of common imaging software at dynamic throughputs. The *NeuroKube* gateway can spawn any number of users into GPU-enabled instances with abstract definitions of CPU cores for a full operating environment for high-performance software.

IV. Labeling and Segmentation Methods in NeuroKube

Applications to resolve a number of biological structures using a manifold of ML techniques are developed, and deployed on the same distributed environment to decompose each challenge of a neuroscience image segmentation problem. ML model trainings and inference demos are partitioned and tested in Jupyter UserLab instances, with a downstream batch Kubernetes process being deployed for full big-data inferences. Models detect objects in a 3D scene, boost resolution, and feed-forward into a full instance cell segmentation of cells in the brain.

Figure 4 shows the end-to-end workflow from the data access to segmentation, labeling and visualization of the reconstruction. The first stage, *Superresolution*, involves addressing the resolution deficit across one of the three cardinal axes in the microscopy volume. A pair of generative adversarial networks are trained on image pairs with and without the target resolution in order to infer the prospective additional sections to the volume. The datasets had section thicknesses

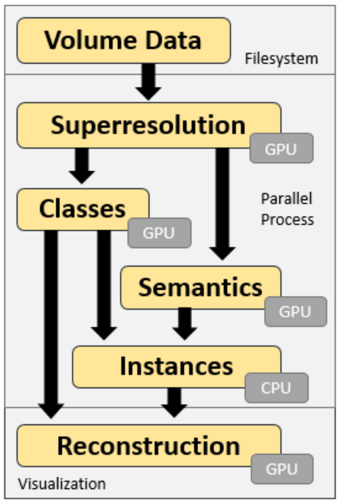


Figure 4. Workflow Steps

in the range of $40 - 60nm$, with a generator resolving to $20 - 30nm$, doubling each image size.

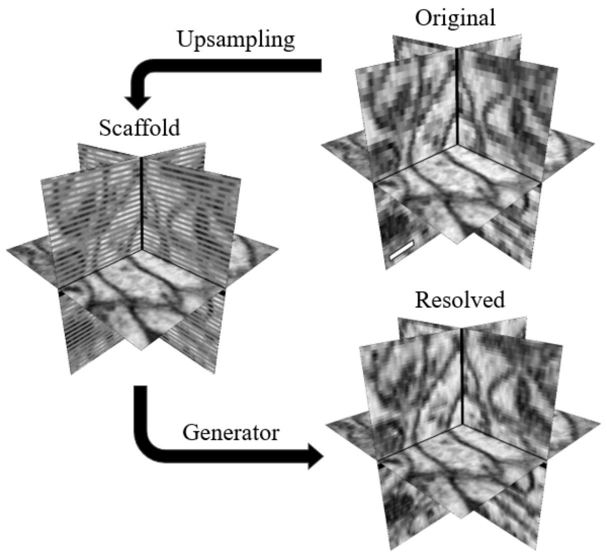


Figure 5. VEM Superresolution (Scale Bar $250nm$)

Figure 5 Orthogonal views of a VEM subvolume. The original view displays the anisotropic problem with the XY plane presenting higher resolution than the XZ or YZ planes. The section thickness is artificially halved by upsampling blank slices interleaved with the original image. A generator then resolves these blank sections into the target resolution by artificially translating pixel features across VEM sections. The resolved orthogonal views now have twice as many voxels along the z-axis with visibly smoother feature details.

Training images for resolution boosting were unpaired, with two separate generative adversarial networks (GANs) cycling images cross-currently to create a pairwise alignment that can be passed to a least-squares objective. Known as a cycleGAN architecture, two GAN pairs and four neural networks are ensembled during training to optimize one single generator, which infers pixel features across the z-axis towards some target resolution [30]. Other similar methods have demonstrated how VEM supperresolution can improve image acuity and automated reconstruction results [31], [32].

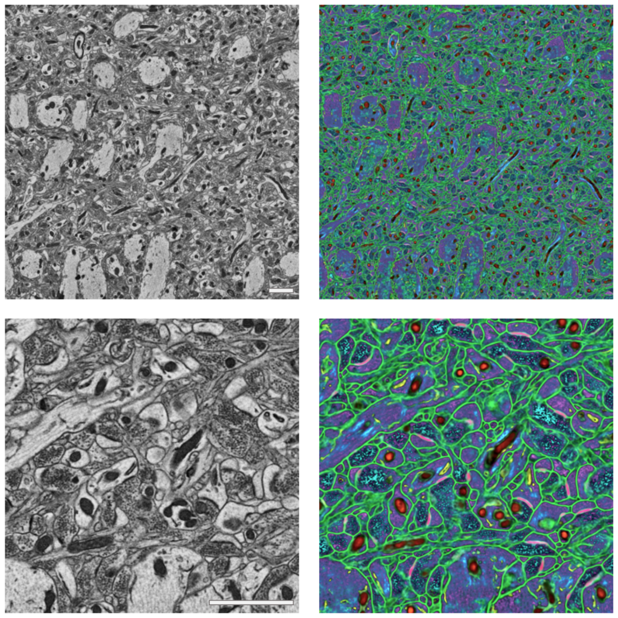


Figure 6. Multilabel segmentation in the Hippocampus

Figure 6 A multilabel classifier output in the Hippocampus. A single neural network is trained to predict seven distinct neuropil structures. Synapses (pink), plasma membranes (green), mitochondria (orange), endoplasmic reticulum (yellow), neurotransmitter vesicles (cyan), cytoskeleton (blue) and cytosol (purple). A 2D slice of 3D volume used in inference (upper left) and the colorized classifier predictions (upper right). A higher magnification shows more of the structural detail (bottom row). (Scale Bar $2\mu m$)

The next two workflow stages, *Semantics* and *Classes*, use a 3D multilabel neural network to classify different biological structures and semantically index regions in the volume. The object detection model operates on nanoscale image features relevant to biological analysis as well as regional attributes to inform downstream instance labeling of cellular regions. As shown in Figures 6 and 7, objects from this stage in the segmentation workflow part into different downstream operational components. The membranes and synapses compile themselves into a boundary map that is used in the cellular instance segmentation to demarcate neuronal and glial processes in the volume. An accessory binary classifier was also used to infer boundary map probabilities. Other subcellular objects such as the cytoskeleton, endoplasmic reticulum, and mitochondria are forwarded to the end reconstruction to await the cellular context computation from the boundary map.

The cell segmentation is solved via the *Instances* step, the workflow's only CPU-driven component. During this phase the boundary map is oversegmented with watershed algorithms and then agglomerated into a final segmentation using global optimization with costs derived from boundary evidence. In order to solve this optimization efficiently at scale, a hierarchical block-wise decomposition scheme is employed. This approach

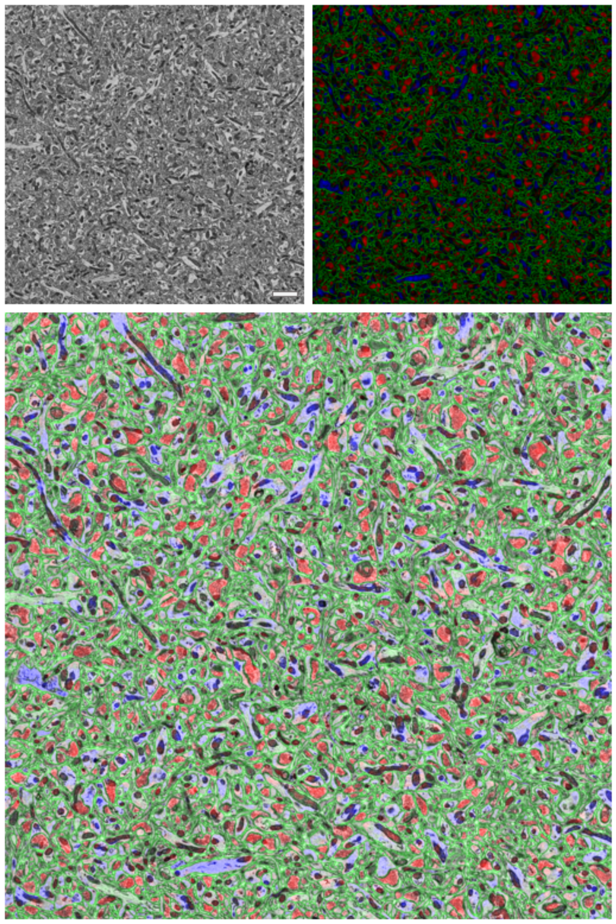


Figure 7. Semantic segmentation in the Cortex (Scale Bar 2μm)

Figure 7 Classification of region semantics in the cortex. A multilabel classifier predicts the appearance of neurites as either axons (red) or dendrites (blue). This classification creates a signal of each for downstream use in biological-prior cell segmentation. A boundary map is inherited from the previous stage's membrane and synapse predictions (green). A 2D VEM section from the 3D volume (upper left) with the colorized predictions (upper right) and colorized VEM overlay (bottom).

stands out for demonstrations that accuracy is sustained at different scales [33]. Further work by *Pape et al.* [34] showed that regional semantics enhance cell segmentation, so an additional GPU-driven workflow step detects axonal vs. dendritic properties on a volume and returns a signal that can be passed alongside the boundary map. This approach encodes spatial semantic attributes as long-range interactions. Thus, it can incorporate information derived from ML-detected regional attributes with respect to known shape and morphological rules to improve separation of cells.

For application design, several different ML architectures were factored into each appropriate function. For generator-discriminator adversarial pairs, deep residual layers were used with upsampling operators [35]. Mutlilabel and binary classification used Unet3D and inception layers respectively [36], [37]. A data concurrency and compression scheme for all internal and external application I/O was defined using n5 files with a Zarr backend [38], [39]. Tensor processing is implicitly parallel using Parsl parallel scripting [40]. For big data inference, node parallelization definitions are compiled in a Python template engine. This is unique from other HPC solutions as knowledge in Python is solely required for scaling design, otherwise HPC systems require learning MPI or other batching commands or scheduling [41], [42]. Models were trained on Jupyter UserLab environments using ephemeral RAM-disk mounts for rapid I/O, and model inference used a scratch allocation to node's local NVMe or SSD space with an I/O exchange with the Ceph's central shared filesystem at the beginning and end of workflow stages. Several rapid I/O ceph block devices are provisioned to store Jupyter userdata or software-specific data at the workflows beginning and end stages, such as the .n5 concurrent files required to import data to *Paintera*.

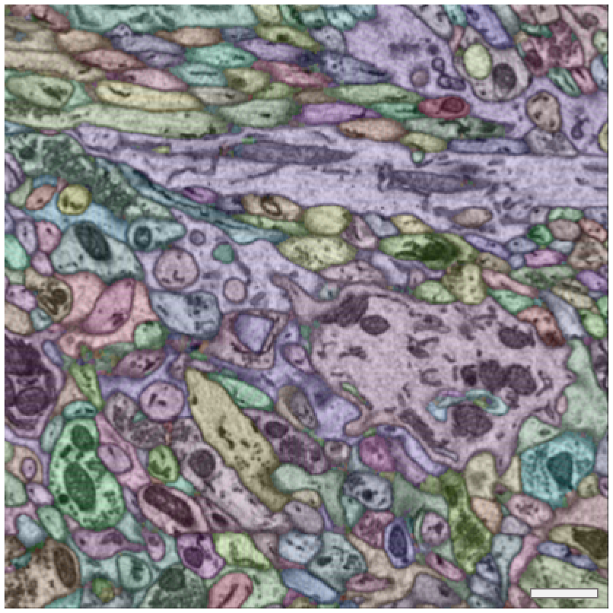


Figure 8. Instance Segmentation in the Cerebellum (Scale Bar 1μm)

Figure 8 Instance segmentation in the cerebellum. The cellular regions in the image are delineated into distinct colors labeling different neurite or glial cell processes.

V. Experimental Results

Workflow stages were executed via a 512 batch Kubernetes job. The parallelization schemes are defined using *python-jinja* template engine to define iterations and model indexes to their appropriate iterative process, with a unique Kubernetes job hosting each batch. At the time of execution the PRP and CHASE-CI had approximately as many GPUs as there were batch jobs submitted. Stages with higher storage footprints were more likely to be re-assigned nodes during execution in order to maintain the required fluid stateless execution state during batch runtime, in which case job the status is "evicted" (See Figure 10). The Kubernetes pod describes a job's specific deployment to some node in the resource pool. During evicted phases, pods preserve runtime information prior to being killed and are re-assigned a node that is able to handle whatever resource demand had been overrun, whether it was RAM, Disk, or CPU usage, allowing jobs to maintain their state regardless of node re-assignments.

Three primary image volumes were processed of the hippocampus, cortex, and cerebellum sourced from mouse tissue. These volumes were approximately 24 x 24 x 24 μm with pixel sizes of 6 x 6 *nm* and slice thicknesses in the 40-60 *nm* range. Two other volumes were also tested with specific workflow stages: A sample of mouse hypothalamus with EM and fluorescent probes for exploratory integration of correlated light-electron microscopy (CLEM) methods (See Figure 3). Another was a cerebral cortex volume of a human specimen biopsy from a patient with Alzheimer's disease (AD) for the purposes of testing these A.I. methods directly on clinically-relevant samples (See Figure 9).

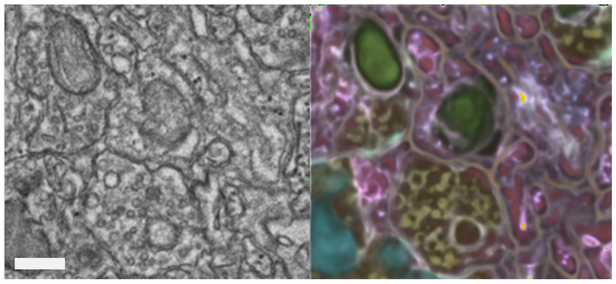
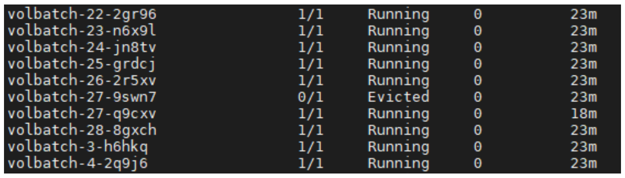


Figure 9. Multilabel segmentation of Alzheimer's disease human biopsy (Scale Bar 250 *nm*)

Figure 9 shows a subregion of a VEM volume acquired from a biopsy of a patient with Alzheimer's disease. Multilabel segmentation labels standard neuropil structures such as synapses (aquamarine), synaptic vesicles (gold), endoplasmic reticulum (pink), mitochondria (green), glycogen (pale blue), cytoplasm (red), and plasma membranes (orange). Pathologically-relevant structures such as autophagic stress (navy blue) are also detected.

Accounting for sections added by superresolution, approximately 40 billion voxels flowed through workflow stages from these five datasets. While GPU provisions were chronologically fluid (See Figure 11), peaks of 80+ GPUs assigned to execution were noticed. Because the Kubernetes Job object assigns resources dynamically based on loads, requests, and resource-availability, exact global processing times and storage footprints are not easily discerned, however the following observations were made. Model trainings were performed in Userlabs with 4 GPUs where convergence time took about 24 hours for classification models (multilabel segmentors) and 72 hours for regressive models (cycleGANs). Time costs were reversed for inference, where classifications took 4-8 hours per dataset and cycleGAN supperresolutions only took 1-3 hours. These short inference times can be attributed to the *PRP/CHASE-CI* ability to load-balance its hundreds of GPUs based on real-time demand, and that inference can be more easily divided, batched, and distributed than model training.

Figure 10 shows this stateless execution during the parallel-process runtime, in which Kubernetes jobs spawn pods onto nodes to iteratively complete a batched assignment. Kubectl



```
volbatch-22-2gr96            1/1     Running     0          23m
volbatch-23-n6x9l            1/1     Running     0          23m
volbatch-24-jn8tv            1/1     Running     0          23m
volbatch-25-grdcj            1/1     Running     0          23m
volbatch-26-2r5xv            1/1     Running     0          23m
volbatch-27-9swn7            0/1     Evicted     0          23m
volbatch-27-q9cxv            1/1     Running     0          18m
volbatch-28-8gxch            1/1     Running     0          23m
volbatch-3-h6hkq             1/1     Running     0          23m
volbatch-4-2q9j6             1/1     Running     0          23m
```

Figure 10. Status of stateless job executions during runtime.

can print the status of all currently operating pods. In this case, *volbatch-27* had overrun some resource allocation or other pods on the node overran their respective resource allocations, so the pod became *evicted*. This means that the job preserved runtime information of the pod prior to being terminated, and transmitted that data to another node with staged Docker layers prepared to inherit the evicted pod's current runtime information, allowing for execution to continue seamlessly in a second running *volbatch-27* pod.

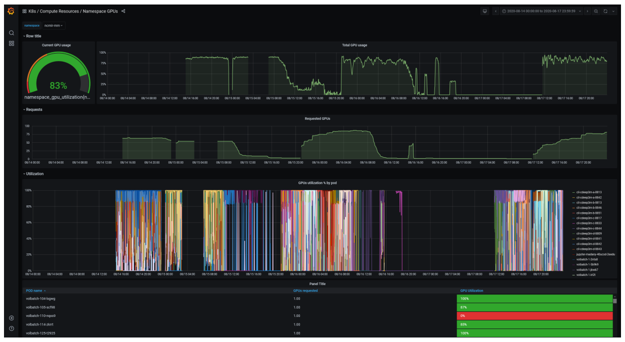


Figure 11. Dashboard of GPU resources

Figure 11 shows a Grafana dashboard of resource utility with respect to graphics accelerations. Workflow stages escalate and de-escalate resource usage as GPUs are added and released over time for highly batched processes.

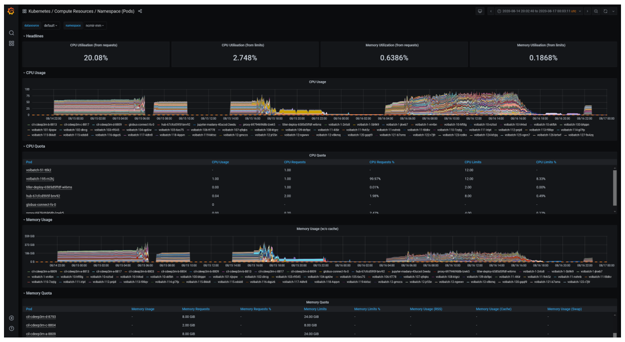


Figure 12. Dashboard of CPU resources

Figure 12 shows a Grafana dashboard of central processing unit (CPU) utility. CPUs are primarily used to import and factor volumetric image data into tensors to be exchanged through the GPU. However, perpetual services built through Kubernetes also provision CPU utility, such as with the JupyterHub service.

Results from segmentation stages are compiled and rendered in 3D to confirm the appearance of targeted neuronal processes (see Figure 13). Final mesh and voxel renders are visualized in VAST and Amira while intermittent stage results are visualized in Paintera, IMOD, and ImageJ [44], [45].

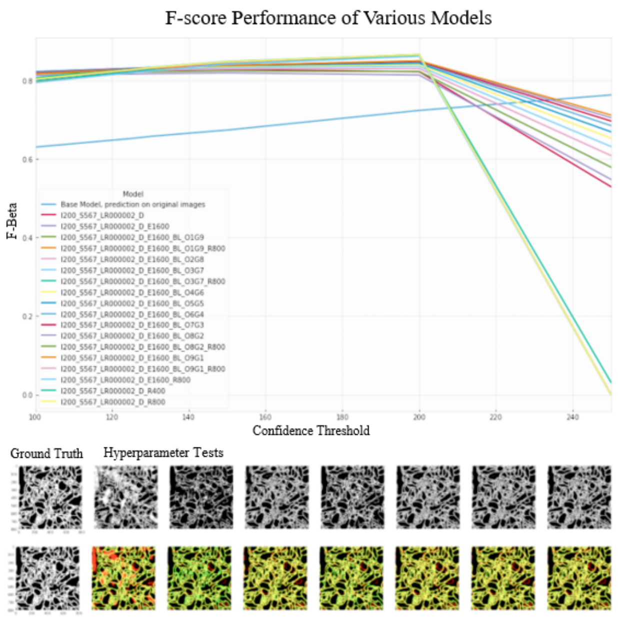


Figure 13. Evaluation of Model Performances

Figure 13 is a representation of the model performance evaluation used by this framework. Various moduls are tested to be integrated into the workflows available by *NeuroKube*. This figure displays tests of automated domain translation by cycleGAN image-to-image regression where inference data was translated to appear more like the training data used by classifiers. This approach has been shown in related work to improve segmentation results [43]. The JupyterHub spawned multiple labs where different users were able to test different hyperparameters with the model names representing a different metric tested. F-scores are evaluated (upper plot) for choosing models and deciding whether a potential workflow stage should be integrated into future designs. A visual of different model outputs (bottom row) referenced against a ground truth volume with true positives (yellow), false positives (red), false negatives (green) and true negatives (black).

VI. Related Work

The NVIDIA Grid vGPU was one of the first distributed computing projects to guarantee real-time access to virtualized GPU resources [46]. This model relied on the construction of smaller, more distributed datacenters to drive real-time access to a networked compute resource. This has furthered the idea that application streaming can be an additional hallmark of cloud computing, not just HPC work [47]. Similarly, the *Neurokube*-PRP relationship is decentralized, and local nodes can be requested on high-bandwidth university networks for low-latency streams for demanding software. A Jupyter Desktop Extension launches users into an Xubuntu environment where software responds in real-time to user interactions. Recent image analysis software has been developed to handle more complex tasks in tandem with ML workflows, however the computational demand of these solutions increases as well, furthering the need for virtualized cloud-enabled resources.

The Janelia DVID system mirrors several of the NeuroKube's key features but with several distinct differences [48].

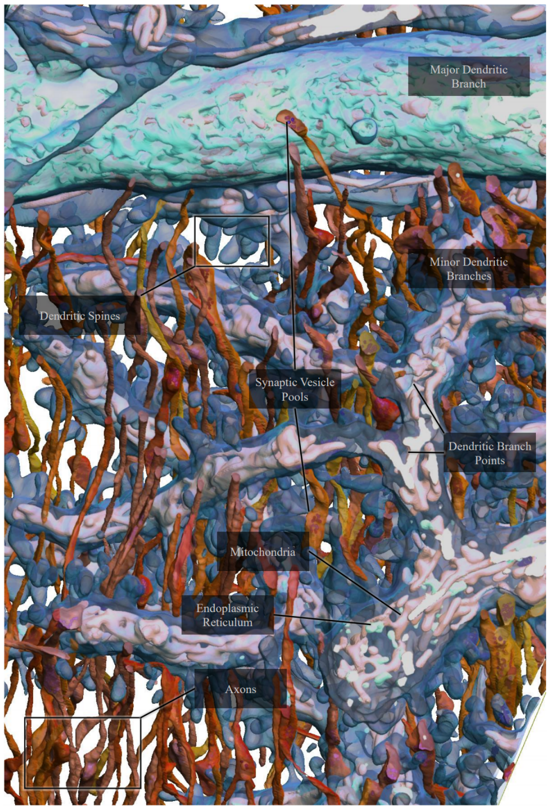


Figure 14. Fully computer-vision generated 3D neuronal scene

Figure 14 shows a subset of neurites from cerebellum neuropil that was extracted and rendered in 3D with structures of interest labeled.

DVID is a volumetric image storage service for neuroscience data with APIs to interact with cloud-hosted databases and local clusters with analysis workflows. DVID is written in GO with a RestAPI for porting into web services, local storage clusters, viewable public databases, and resources for segmentation workflows. *Neurokube* is built fully cloud-native ontop of a Kubernetes engine, which is a compiled GO middleware, with data storage completely decentralized. No single datacenter or compute cluster hosts any of the totality of *Neurokube* scientific data. DVID stores large immutable grayscale data into arrays of HDD drives with label and annotation data compressed into SSD arrays in sequence. *NeuroKube* uses purely NVMe and SSD storage, with large immutable grayscale data stored in a Ceph shared filesystem, and label/annotation data stored into rapid I/O ceph block devices with RADOS protocols for S3 and Swift APIs [19]. These partitioned volumes can span multiple nodes worldwide. At the time of writing, DVID focuses on a public interface for Petascale data and databasing, while NeuroKube emphasises Terascale access to zero-latency HPC software and a diverse set of GPU-powered ML stages in image processing, segmentation, and analysis workflows.

For clinical neurology, DATAVIEW is a workflow processing system for clinical imaging such as magnetic resonance (MRI) and other spectroscopies [49]. Because this system is used in diagnostic decision-making, scaling and sophistication is important, as the system must account for patient load without exception. DATAVIEW thereby correlates job submissions and integrated HPC grids with contact information for algorithm developers for specific workflow stages. *NeuroKube* uses a JupyterHub deployment, enabling developers to access the Userlabs of individual users executing workflows in real-time, allowing for closer interplay with exception-handling, scaling executions, and use case specifics.

A few other scalable neuroimaging frameworks and pipelines are being developed prior to publication with varying degrees of human-in-the-loop and machine intelligence dependence, and end-goals specific towards clinical neurology, experimental neurobiology, or connectomics [50], [51].

VII. Conclusions and Future Work

The *NeuroKube* framework demonstrates a new way to compile complex computational tasks into a centralized usable space. While resources and executions happen transiently over nodes worldwide, the framework regardless provides a central hub for workflow development, testing, and software integration at the throughput necessary to tackle the challenges of automated reconstructions of neural tissue.

Automated label-analysis of the brain has broad implications in our understanding of brain connectivity, cognition, and disease. Simple measurements like the precise total number of brain cells and synapses in a vertebrate organism remain unknown. The human brain is remarkable for both its size and complexity, particularly in regions associated with memory functions or reasoning, when compared to non-human primate brains, yet knowledge of the fine details of cellular circuits in brain areas which have structurally adapted to enable advanced performance remains limited due to in our ability to build and analyze detailed electron-microscopically-charted wiring diagrams of connectivity in rodent or primate brains. Determining the organization of cellular structures and the spatial proteome withing brain cells (neurons and glia) provides an even more daunting challenge. For example, despite fulfilling a critical role in major brain functions and hallmarks of neuropathology, glial cells such as astrocytes constitute a large proportion of the cell types in the brain, yet even less is understood about their cell morphology and tissue-level context due to the their greater difficulty in computationally-enhanced reconstructions when compared to neurons [52], [53].

One of the greatest challenges in brain research is to accurately determine the trajectory of long projections of neuronal processes from one brain region to another, to define what is referred to the “projectome”, differentiated from “connectomes”, which to date remain smaller-scale maps within up to a cubic *mm* of brain tissue. Neurons can be of both staggering and insignificant size when measured by either their projection extents or cell-body size. The cell bodies of Betz and Purkinje neurons of the motor cortex and cerebellum, respectively, are extraordinarily large, up to 100 μm in diameter, with fiber lengths of 10-100s *cm* (depending upon species), while smaller neuron types, such as cortical interneurons, display cell bodies of approximately 10 μm in diameter, with axon projection lengths in the 10-100s of micrometers [54], [55]. Virtually any metric applied to the morphology of the neuron can be expected to vary by orders of magnitude across individual cells. Synapses as well are heterogeneous, and brain areas with higher cognitive functions contain more structurally diverse synapses [12]. As such, the study of all components of visible ultrastructure should be evaluated when studying the brain. Mitochondria, one of the most ubiquitous and vital organelles, play roles in energy production and also modulate calcium signals in actively-firing neurons, as does the endoplasmic reticulum [56]. Synaptic vesicles in axon terminals actively release neurotransmitter in organized structural patterns that can indicate the strengths and types of synapses [57], [58]. Endoplasmic reticulum, often structured into organized arrays, can distribute in ways that suggest a role in modulation of synaptic plasticity [59]. The precise location of a synapse harbors ultrastructural specializations which correlate with complex organization of localized post-synaptic protein assemblies, which modulate the post-synaptic response to pre-synaptic input [60]. Using A.I., reducing all of these structures to metrics will elucidate the complex hierarchy of information required to demystify the grandest challenges of neuroscience.

Research into dysfunctional brain states and their underlying mechanisms can be greatly accelerated by the presence of high-level cyberinfrastructure for A.I.-driven analysis of brain circuits. As humans age, dementias driven by processes underlying Alzheimer’s or Parkinson’s diseases remain a monumental burden on human health. Alzheimer’s is characterized by the accumulation of neurofibrillary tangles; Parkinson’s and Alzheimer’s both show stress, structural dystrophy, and loss of mitochondria and synapses [61]. All can be imaged by electron microscopy and detected and analyzed by image processing augmented by machine intelligence. Demyelinating diseases such as Multiple Sclerosis (MS) and Amyotrophic Lateral Sclerosis (ALS) form disastrous complications in the brains of patients, with neuron-specific structures becoming a false target of the immune system [62]. Vascular disorders that cause strokes, Epilepsy-induced seizures, and cancers of the brain all damage the highly sophisticated organization of brain tissue, and much remains to be discovered about the implications of all the biological component affects [63]. Our premise is that advancing scalable A.I. technologies will provide new gateways to propel advances in knowledge and understanding of the earliest and causative processes which underly these neurodegenerative diseases.

Kubernetes provides a vast toolkit of high-level cyberinfrastructure. In collaboration with PRP and CHASE-CI, *NeuroKube* has so far only scratched the surface of the potential of cloud-native development. The design of stateful and stateless applications and databases will be explored as a direction to handle large and more complex data across broad computational solutions, testing next-generation web services and new computing chips, and furthering the integration of artificially intelligent systems. We also intend these workflows to be test-driven and collaboratively developed using the PPoDSlab (see ppods.ai).

ACKNOWLEDGMENT

The authors would like to thank Anton Maximov, Takaki Komiyama, Daniela Boassa, Eric Bushong and Thomas Deerinck for providing data. Ashok Mondal, Daniel Silva, Doreh Kazemisefat, Matthew Bond, and Peishan Sha helped test and evaluate workflow stages. The PRP and CHASE-team members, in particular Tom DeFanti, Dima Mishin and John Graham, provided the Kubernetes environment along with the distributed GPU resources. We thank Constantin Pape for manuscript comments. The work was supported by NIH U24 NS120055-01 to MHE and NIH, NIGMS P41GM103412 and RO1 GM086197, GM082949 to MHE, as well as NINDS RO1 NS027177 to MHE. The PRP and CHASE-CI are supported through the NSF grants 1541349 and 1730158.

REFERENCES

[1] K. L. Briggman and D. D. Bock, *Volume electron microscopy for neuronal circuit reconstruction*, Feb. 2012. DOI: 10.1016/j.conb.2011.10.022.

[2] P. Bajcsy and N. Hotaling, “Interoperability of Web Computational Plugins for Large Microscopy Image Analyses”, DOI: 10.6028/NIST.IR.8297.

[3] A. Motta, M. Berning, K. M. Boergens, *et al.*, “Dense connectomic reconstruction in layer 4 of the somatosensory cortex”, *Science*, vol. 366, no. 6469, Nov. 2019, ISSN: 10959203. DOI: 10.1126/science.aay3134.

[4] M. Helmstaedter, K. L. Briggman, S. C. Turaga, *et al.*, “Connectomic reconstruction of the inner plexiform layer in the mouse retina”, *Nature*, vol. 500, no. 7461, pp. 168–174, 2013, ISSN: 14764687. DOI: 10.1038/nature12346.

[5] J. Kornfeld and W. Denk, *Progress and remaining challenges in high-throughput volume electron microscopy*, Jun. 2018. DOI: 10.1016/j.conb.2018.04.030.

[6] S. Chopra and M. R. Rao, “The partition problem”, *Mathematical Programming*, vol. 59, no. 1-3, pp. 87–115, Mar. 1993, ISSN: 00255610. DOI: 10.1007/BF01581239.

[7] L. Smarr, J. Graham, C. Crittenden, *et al.*, “The pacific research platform: Making high-speed networking a reality for the scientist”, in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Jul. 2018, ISBN: 9781450364461. DOI: 10.1145/3219104.3219108.

[8] I. Altintas, K. Marcus, I. Nealey, *et al.*, “Workflow-driven distributed machine learning in CHASE-CI: A cognitive hardware and software ecosystem community infrastructure”, in *Proceedings - 2019 IEEE 33rd International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2019*, Institute of Electrical and Electronics Engineers Inc., May 2019, pp. 865–873, ISBN: 9781728135106. DOI: 10.1109/IPDPSW.2019.00142. arXiv: 1903.06802.

[9] *Kubernetes Documentation - Kubernetes*. [Online]. Available: https://kubernetes.io/docs/home/.

[10] E. Ahmed, A. Saint, A. Shabayek, *et al.*, “A survey on Deep Learning Advances on Different 3D Data Representations”, vol. 1, no. 1, 2019. arXiv: 1808.01462v2.

[11] R. Bellman, *Dynamic Programming*. Courier Corporation, 1957, ISBN: 978-0486428093.

[12] W. Zhu, Q. Lou, S. Vang, *et al.*, “Deep Multi-instance Networks with Sparse Label Assignment for Whole Mammogram Classification”, Tech. Rep. arXiv: 1612.05968v1.

[13] J. Read, A. Puurula, and A. Bifet, “Multi-label Classification with Meta-Labels”, in *Proceedings - IEEE International Conference on Data Mining, ICDM*, vol. 2015-January, Institute of Electrical and Electronics Engineers Inc., Jan. 2014, pp. 941–946, ISBN: 9781479943029. DOI: 10.1109/ICDM.2014.38.

[14] K. J. Hayworth, D. Peale, M. Januszewski, *et al.*, “Gas cluster ion beam SEM for imaging of large tissue samples with 10 nm isotropic resolution”, *Nature Methods*, vol. 17, no. 1, pp. 68–71, Jan. 2020, ISSN: 15487105. DOI: 10.1038/s41592-019-0641-2.

[15] L. Heinrich, J. A. Bogovic, and S. Saalfeld, “Deep Learning for Isotropic Super-Resolution from Non-isotropic 3D Electron Microscopy”, in *Medical Image Computing and Computer-Assisted Intervention*, M. Descoteaux, L. Maier-Hein, A. Franz, *et al.*, Eds., Cham: Springer International Publishing, 2017, pp. 135–143, ISBN: 978-3-319-66185-8.

[16] D. G. C. Hildebrand, M. Cicconet, R. M. Torres, *et al.*, “Whole-brain serial-section electron microscopy in larval zebrafish”, *Nature*, vol. 545, no. 7654, pp. 345–349, May 2017, ISSN: 14764687. DOI: 10.1038/nature22356.

[17] Ş. Mocanu, R. Din, D. Saru, *et al.*, “Using graphics processing units for accelerated information retrieval”, *Studies in Informatics and Control*, vol. 23, no. 3, pp. 249–256, 2014, ISSN: 1841429X. DOI: 10.24846/v23i3y201403.

[18] T. Janssen, *What is Cloud-Native? Is It Hype or the Future of Software Development?*, 2018. [Online]. Available: https://stackify.com/cloud-native/.

[19] *Ceph — Documentation*. [Online]. Available: https://docs.ceph.com/docs/master/.

[20] *Project Jupyter — Documentation*. [Online]. Available: https://jupyter.org/documentation.

[21] K. Dinghofer and F. Hartung, “Analysis of Criteria for the Selection of Machine Learning Frameworks”, in *2020 International Conference on Computing, Networking and Communications, ICNC 2020*, Institute of Electrical and Electronics Engineers Inc., Feb. 2020, pp. 373–377, ISBN: 9781728149059. DOI: 10.1109/ICNC47757.2020.9049650.

[22] *Node.js — Documentation*. [Online]. Available: https://nodejs.org/en/docs/.

[23] Patel Mohak, *ImageSliceViewer3D: 3D image or volume slice viewer for jupyter notebook*. [Online]. Available: https://github.com/mohakpatel/ImageSliceViewer3D.

[24] *Ipyvolume — Documentation*. [Online]. Available: https://ipyvolume.readthedocs.io/en/latest/.

[25] *Nextcloud — Documentation*. [Online]. Available: https://docs.nextcloud.com/.

[26] *RocketChat — Documentation*. [Online]. Available: https://docs.rocket.chat/.

[27] J. R. Kremer, D. N. Mastronarde, and J. R. McIntosh, “Computer visualization of three-dimensional image data using IMOD”, *Journal of Structural Biology*, vol. 116, no. 1, pp. 71–76, Jan. 1996, ISSN: 10478477. DOI: 10.1006/jsbi.1996.0013.

[28] J. Schindelin, I. Arganda-Carreras, E. Frise, *et al.*, *Fiji: An open-source platform for biological-image analysis*, Jul. 2012. DOI: 10.1038/nmeth.2019.

[29] P. Hanslovsky, I. Pisarev, V. Leite, *et al.*, *Saalfeldlab/Paintera: Paintera 0.23.3*, Feb. 2020. DOI: 10.5281/ZENODO.3692024. [Online]. Available: https://zenodo.org/record/3692024.

[30] J.-Y. Zhu, T. Park, P. Isola, *et al.*, “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networkss”, in *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[31] Z. Wang, J. Chen, and S. C. Hoi, “Deep Learning for Image Super-resolution: A Survey”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, Mar. 2020, ISSN: 0162-8828. DOI: 10.1109/tpami.2020.2982166. arXiv: 1902.06068.

[32] L. Fang, F. Monroe, S. W. Novak, *et al.*, “Deep Learning-Based Point-Scanning Super-Resolution Imaging”, *bioRxiv*, p. 740 548, Oct. 2019. DOI: 10.1101/740548.

[33] C. Pape, T. Beier, P. Li, *et al.*, “Solving Large Multicut Problems for Connectomics via Domain Decomposition”, in *Pro-*

ceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017, vol. 2018-Janua, Institute of Electrical and Electronics Engineers Inc., Jul. 2017, pp. 1–10, ISBN: 9781538610343. DOI: 10.1109/ICCVW.2017.7.

[34] C. Pape, A. Matskevych, A. Wolny, *et al.*, "Leveraging Domain Knowledge to Improve Microscopy Image Segmentation With Lifted Multicuts", *Frontiers in Computer Science*, vol. 1, no. 6, p. 6, Oct. 2019, ISSN: 2624-9898. DOI: 10.3389/fcomp.2019.00006.

[35] K. He, X. Zhang, S. Ren, *et al.*, "Deep residual learning for image recognition", in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-Decem, IEEE Computer Society, Dec. 2016, pp. 770–778, ISBN: 9781467388504. DOI: 10.1109/CVPR.2016.90. arXiv: 1512.03385.

[36] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation", in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9351, Springer Verlag, May 2015, pp. 234–241, ISBN: 9783319245737. DOI: 10.1007/978-3-319-24574-4_28. arXiv: 1505.04597.

[37] M. G. Haberl, C. Churas, L. Tindall, *et al.*, "CDeep3M—Plug-and-Play cloud-based deep learning for image segmentation", *Nature Methods*, vol. 15, no. 9, pp. 677–680, Sep. 2018, ISSN: 15487105. DOI: 10.1038/s41592-018-0106-z. [Online]. Available: https://doi.org/10.1038/s41592-018-0106-z.

[38] C. Pape, *Constantinpape/z5*, 2019. DOI: 10.5281/ZENODO.3585752. [Online]. Available: https://zenodo.org/record/3585752.

[39] S. Saalfeld, *n5 — Documentation*. [Online]. Available: https://github.com/saalfeldlab/n5.

[40] Y. Babuji, A. Woodard, Z. Li, *et al.*, "Parsl: Pervasive parallel programming in Python", in *HPDC 2019- Proceedings of the 28th International Symposium on High-Performance Parallel and Distributed Computing*, New York, NY, USA: Association for Computing Machinery, Inc, Jun. 2019, pp. 25–36, ISBN: 9781450366700. DOI: 10.1145/3307681.3325400. arXiv: 1905.02158.

[41] *Jinja — Documentation*. [Online]. Available: https://jinja.palletsprojects.com/en/2.11.x/.

[42] I. Foster, I. Foster, and J. Foster, *Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering*, ser. Literature and Philosophy. Addison-Wesley, 1995, ISBN: 9780201575941.

[43] M. Januszewski and V. Jain, "Segmentation-Enhanced CycleGAN", *bioRxiv*, p. 548 081, Feb. 2019. DOI: 10.1101/548081.

[44] D. R. Berger, H. S. Seung, and J. W. Lichtman, "VAST (Volume Annotation and Segmentation Tool): Efficient Manual and Semi-Automatic Labeling of Large 3D Image Stacks", *Frontiers in Neural Circuits*, vol. 12, p. 88, Oct. 2018, ISSN: 1662-5110. DOI: 10.3389/fncir.2018.00088.

[45] D. Stalling, M. Westerhoff, and H. C. Hege, "Amira: A highly interactive system for visual data analysis", in *Visualization Handbook*, Elsevier Inc., 2005, pp. 749–767, ISBN: 9780123875822. DOI: 10.1016/B978-012387582-2/50040-X.

[46] *Accelerated Virtual Desktops for Mobile and Office Workers — NVIDIA Virtual GPU Solutions*. [Online]. Available: https://www.nvidia.com/en-us/data-center/virtual-gpu-technology/.

[47] S. Smallen, W. Cirne, J. Frey, *et al.*, "Combining workstations and supercomputers to support Grid applications: The parallel tomography experience", *Proceedings of the Heterogeneous Computing Workshop, HCW*, pp. 241–252, 2000. DOI: 10.1109/hcw.2000.843748.

[48] W. T. Katz and S. M. Plaza, "DVID: Distributed Versioned Image-Oriented Dataservice", *Frontiers in Neural Circuits*, vol. 13, no. 5, p. 5, Feb. 2019, ISSN: 1662-5110. DOI: 10.3389/fncir.2019.00005.

[49] H. Hamidian, S. Lu, S. Rana, *et al.*, "Adapting Medical Image Processing Tasks to a Scalable Scientific Workflow System", Institute of Electrical and Electronics Engineers (IEEE), Sep. 2014, pp. 385–392. DOI: 10.1109/services.2014.74.

[50] R. Vescovi, H. Li, J. Kinnison, *et al.*, "Toward an Automated HPC Pipeline for Processing Large Scale Electron Microscopy Data", Nov. 2020. arXiv: 2011.03204.

[51] E. Johnson, M. Wilt, L. Rodriguez, *et al.*, "Toward A Reproducible, Scalable Framework for Processing Large Neuroimaging Datasets", *bioRxiv*, p. 615 161, Apr. 2019. DOI: 10.1101/615161.

[52] N. J. Allen and B. A. Barres, *Neuroscience: Glia - more than just brain glue*, Feb. 2009. DOI: 10.1038/457675a.

[53] E. A. Bushong, M. E. Martone, Y. Z. Jones, *et al.*, "Protoplasmic astrocytes in CA1 stratum radiatum occupy separate anatomical domains", *Journal of Neuroscience*, vol. 22, no. 1, pp. 183–192, Jan. 2002, ISSN: 02706474. DOI: 10.1523/jneurosci.22-01-00183.2002.

[54] S. Herculano-Houzel, "The human brain in numbers: a linearly scaled-up primate brain", *Frontiers in Human Neuroscience*, vol. 3, no. NOV, p. 31, Nov. 2009, ISSN: 16625161. DOI: 10.3389/neuro.09.031.2009.

[55] B. B. Andersen, L. Korbo, and B. Pakkenberg, "A quantitative study of the human cerebellum with unbiased stereological techniques", *Journal of Comparative Neurology*, vol. 326, no. 4, pp. 549–560, 1992, ISSN: 10969861. DOI: 10.1002/cne.903260405.

[56] E. Britti, F. Delaspre, J. Tamarit, *et al.*, "Mitochondrial calcium signalling and neurodegenerative diseases", *Neuronal Signaling*, vol. 2, no. 4, p. 20 180 061, Dec. 2018, ISSN: 2059-6553. DOI: 10.1042/ns20180061.

[57] S. O. Rizzoli and W. J. Betz, *Synaptic vesicle pools*, 2005. DOI: 10.1038/nrn1583.

[58] T. Steinkellner, M. Madany, M. G. Haberl, *et al.*, "A genetic probe for visualizing glutamatergic synapses and vesicles by 3D electron microscopy", *bioRxiv*, p. 1090, Aug. 2020. DOI: 10.1101/2020.07.31.230995.

[59] N. Holbro, Å. Grunditz, and T. G. Oertner, "Differential distribution of endoplasmic reticulum controls metabotropic signaling and plasticity at hippocampal synapses", *Proceedings of the National Academy of Sciences of the United States of America*, vol. 106, no. 35, pp. 15 055–15 060, Sep. 2009, ISSN: 00278424. DOI: 10.1073/pnas.0905110106.

[60] A. Dosemeci, R. J. Weinberg, T. S. Reese, *et al.*, "The Postsynaptic Density: There Is More than Meets the Eye", *Frontiers in Synaptic Neuroscience*, vol. 8, no. AUG, p. 23, Aug. 2016, ISSN: 1663-3563. DOI: 10.3389/fnsyn.2016.00023.

[61] E. Bereczki, P. T. Francis, D. Howlett, *et al.*, "Synaptic proteins predict cognitive decline in Alzheimer's disease and Lewy body dementia", *Alzheimer's and Dementia*, vol. 12, no. 11, pp. 1149–1158, Nov. 2016, ISSN: 15525279. DOI: 10.1016/j.jalz.2016.04.005.

[62] E. Nutma, M. C. Marzin, S. A. Cillessen, *et al.*, "Autophagy in White Matter Disorders of the CNS: Mechanisms and Therapeutic Opportunities", *The Journal of Pathology*, path.5576, Nov. 2020, ISSN: 0022-3417. DOI: 10.1002/path.5576.

[63] V. L. Feigin, E. Nichols, T. Alam, *et al.*, "Global, regional, and national burden of neurological disorders, 1990–2016: a systematic analysis for the Global Burden of Disease Study 2016", *The Lancet Neurology*, vol. 18, no. 5, pp. 459–480, May 2019, ISSN: 14744465. DOI: 10.1016/S1474-4422(18)30499-X.