



**Deep Learning Acceleration** with

**MATRIX**  
*powered by* **bitfusion FLEX**

*A Technical White Paper*

# TABLE OF CONTENTS

<b>The Promise of AI</b> . . . . .	1
<b>The Challenges of the AI Lifecycle and Infrastructure</b> . . . . .	1
<b>MATRIX Powered by Bitfusion Flex Solution Architecture</b> . . . . .	3
Bitfusion Core . . . . .	4
Container Management . . . . .	5
Smart Resourcing . . . . .	7
Data Volumes . . . . .	7
On-Premise GPU Cloud . . . . .	8
Interactive Workspaces . . . . .	8
<b>Next Steps</b> . . . . .	10

## The Promise of AI

Deep Learning and other AI developments are disrupting every industry in incredible ways – self-driving cars, drones, virtual assistants, more accurate medical diagnosis, automatic lead generation, better customer service, cybersecurity, and much more.

By leveraging AI solutions, businesses are discovering ways to create exciting new products, boost profits, differentiate from the competition, and maximize the investments they've made in big data infrastructure and data science talent.

## The Challenges of the AI Lifecycle and Infrastructure

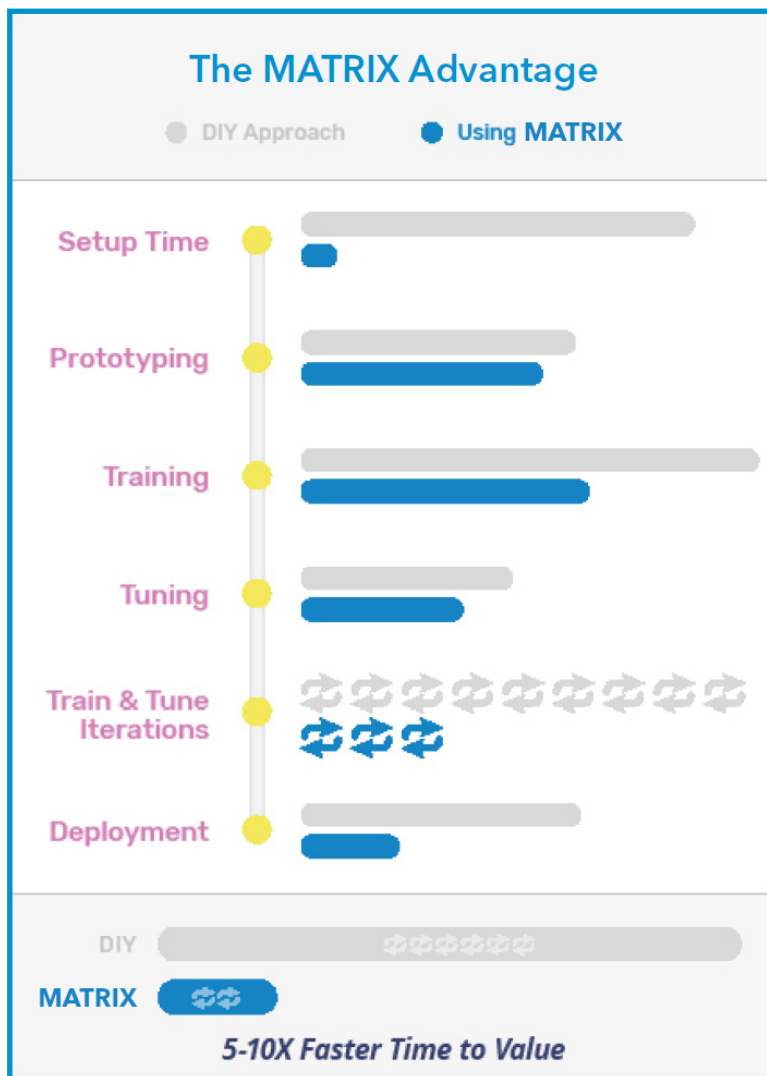
However, these opportunities are not golden tickets to success. One reason is because Deep Learning and AI is not one-size-fits-all. In order to derive the maximum benefit, it is crucial for companies to train on their own data sets and modify models and code to fit their unique use cases. The companies that stand to benefit the most are the ones taking a proactive approach with their data. They perform their own model training to create tailored algorithms and defensible intellectual property. They look to leverage Deep Learning to create feature sets adding unique and intelligent value to their services and products, all developed as efficiently as possible as entire industries race to incorporate the benefits of AI.

Typically, developing Deep Learning based applications is a lengthy and cost-intensive process. Development requires continuous iterations of training to debug and optimize the model for each use case. Once completed, companies realize that months were spent on devops and infrastructure tasks, followed by iterating in a low visibility, guess-and-test oriented environment, with limited scalability.

## 10X Faster Development, Greater Efficiency, Unlimited Scalability

The MATRIX was designed as a DL-in-a-Box solution that circumvents inefficiencies to fast track AI development and deployment. As a comprehensive line of plug & play appliances packaged with all the tools necessary to support full life cycles of Deep Learning development, the MATRIX eliminates set up overhead so users achieve faster time to productivity. Fully-integrated with the Bitfusion Flex suite, the MATRIX provides a dev environment featuring an intuitive user interface that natively supports model iteration and optimization of workloads, high visibility and control over resources, and compute virtualization capability that can easily scale up to massive numbers of GPUs for support of large datasets.

**Figure 1:** Deep Learning Do-It-Yourself vs. MATRIX



## MATRIX Solution Architecture

For fastest time to productivity, the MATRIX software, powered by Bitfusion Flex, is optimally deployed on MATRIX appliances ranging from rackscale clusters featuring advanced cooling, monitoring and power efficiency features, to high-performance servers and dev workstations. MATRIX appliances have been fully-integrated with Deep Learning software for a true plug & play experience.



[SMART]RACK AI



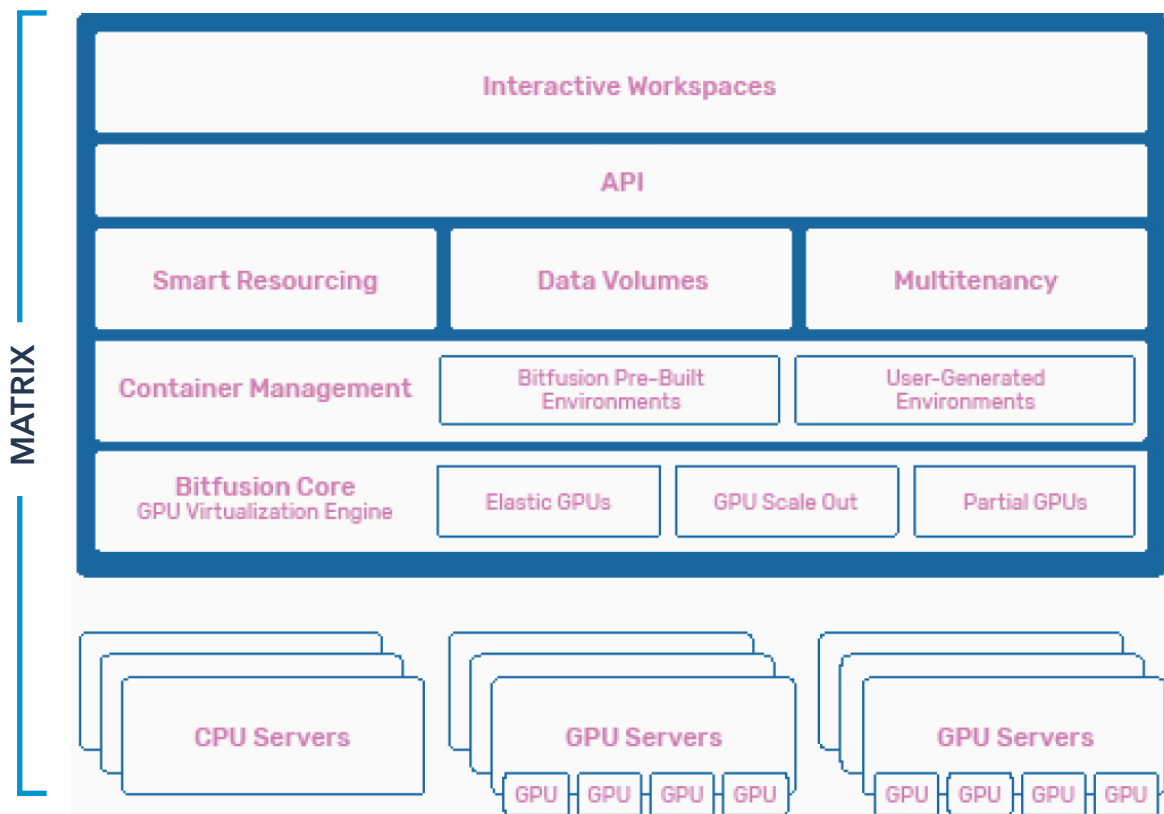
MATRIX 280



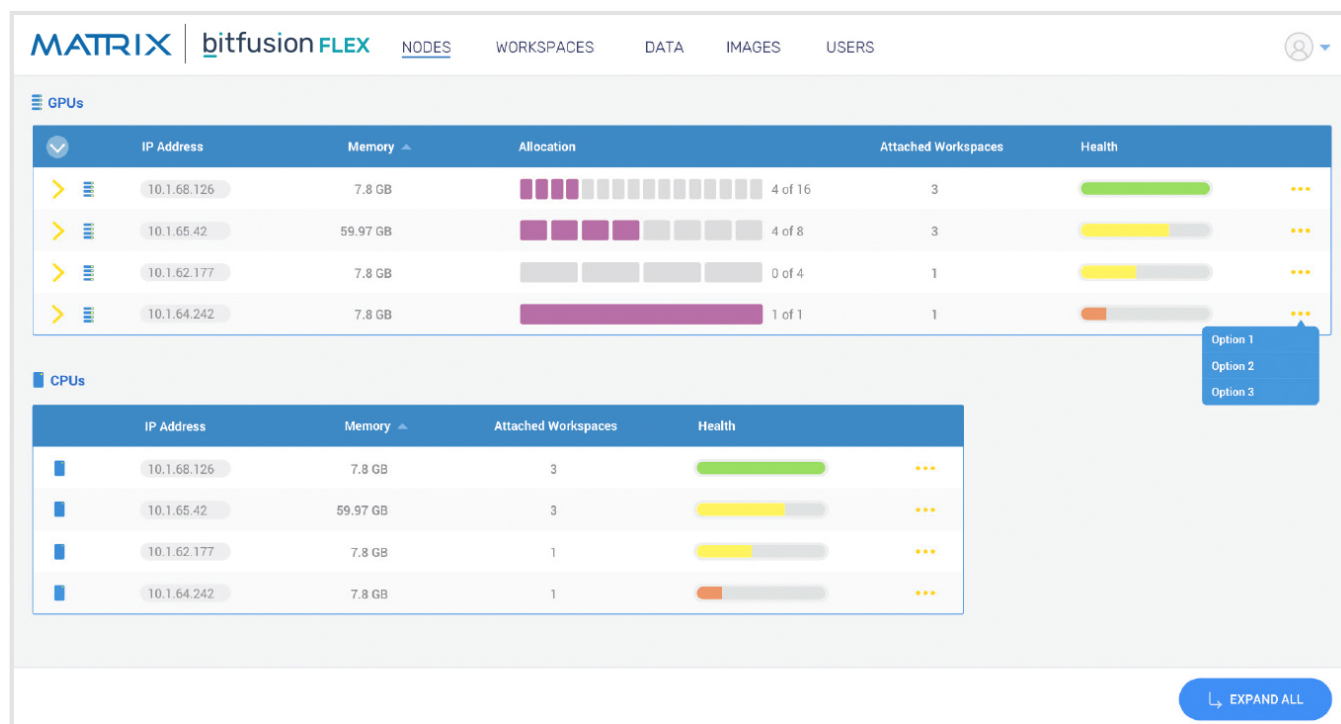
MATRIX 400

The software can also be used to aggregate a heterogeneous resource pool, such as various MATRIX appliances and existing resources to consolidate them into a highly elastic on-premise GPU cloud for maximum resource versatility.

**Figure 2:** Technology Architecture



**Figure 3:** Nodes View



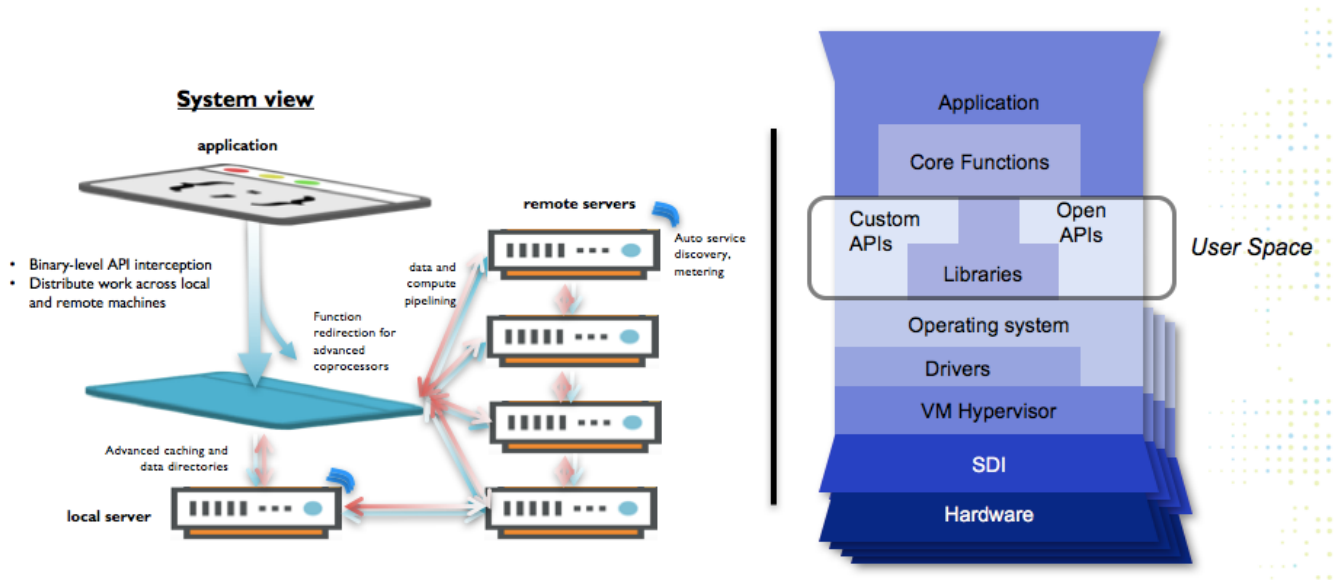
By leveraging the MATRIX, developers can begin their development with CPU compute only in order to conserve GPU resources for other users or more critical workloads. When the code is ready to be deployed for training using GPUs, elastic network-attached GPUs can be allocated to the CPU workspace. To scale up, the workload can be run using one or many local and remote GPUs for rapid model training while maintaining a simple programming paradigm.

It is recommended to utilize 10GbE data/compute fabric for workstations and dual 25GbE with ROCE for servers to provide sufficient network bandwidth between hosts and client servers. EDR Infiniband (100Gb) is recommended for large deployments that require scale-up performance across nodes and networks.

## Bitfusion Core

The Bitfusion Core GPU virtualization engine is the key to Bitfusion Flex's versatility. Bitfusion Core intercepts API calls between applications and the underlying GPU compute. This enables elastic GPU attachment across nodes over the network (network-attached GPUs), partial GPUs (memory partitioning), GPU scale out of up to 64 GPUs attached to a single work process or container, and overall optimizations for unrivaled performance.

**Figure 4:** Bitfusion Core Technology Architecture



**Supports the latest CUDA features including unified memory**

Bitfusion Core uses an “interceptor” approach instead of hypervisors. Therefore, no changes are required to the underlying hardware or virtual machine environments, or to the application code itself. This configuration gives AI developers and data scientists the ability to leverage the benefits of GPU virtualization seamlessly with minimal overhead and integration requirements.

## Container Management

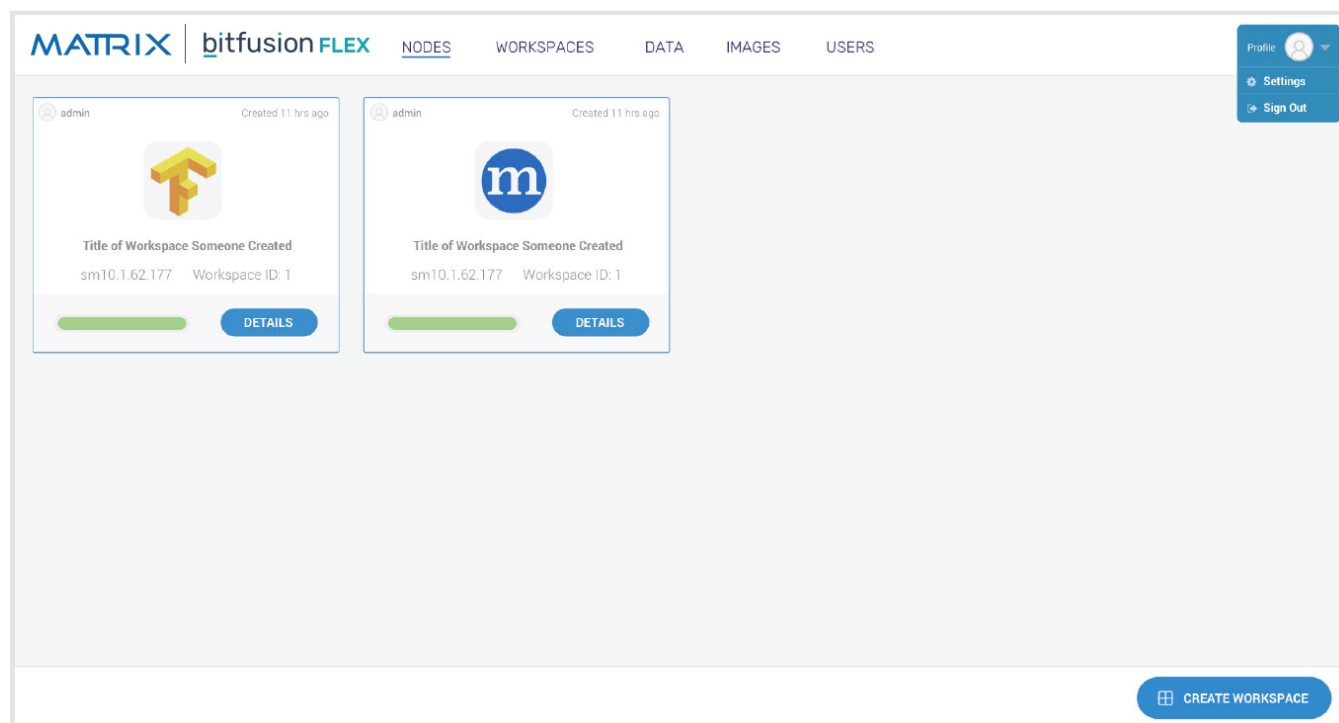
Modern IT organizations continue to realize the benefits of running applications in containers. As a resulting trend, a steady increase in adoption of frameworks like Docker is seen in the industry to replace bare metal and VM solutions. However, to optimize container implementations for GPU-accelerated workloads, the Docker source code, while open source, has to be modified. One example of a custom implementation is DockerNV from NVIDIA.

Rather than spending hundreds of thousands of engineering hours to integrate, optimize and maintain GPUs, Infiniband, DL frameworks, etc. in a containerized environment, IT organizations and researchers can now benefit from the fully-integrated MATRIX solution.

The MATRIX builds on top of its powerful compute platform and GPU virtualization capabilities with a comprehensive container management solution. By running an application on a Docker-based framework, users and administrators experience an unprecedented gain in productivity, flexibility, and manageability, including:

- Managing multiple environments to experiment with different frameworks
- Configuring custom/individual environments
- Taking snapshots for backup or to create new container templates
- At the click of a button, download pre-configured container templates with the latest DL frameworks
- Dynamically allocate GPU resources to containers to accommodate changing workloads or timelines
- Easily port containers to different systems

**Figure 5:** Example Workspaces With Pre-Built Environments



The MATRIX container management layer includes a built-in repository for managing pre-built containers while providing the option to create your own user-generated containers:

- **MATRIX Pre-Built Environments:** MATRIX containerized environments are kept up-to-date with the latest Deep Learning frameworks and data science libraries to ensure your ability to leverage the best enhancements developed by TensorFlow, Caffe, Torch, and other communities. An administrator can run a single command to pull down all the latest containers (environments) into the cluster without any disruption to running workloads.
- **User-Generated Environments:** Users can leverage “workspace snapshots” or “bring-your-own-container” as ways to modify and save container environments for use by others. Workspace snapshot leverages a ‘docker save’ workflow to duplicate an environment and add it to the repository with changes saved. The bring-your-own-container approach leverages a minimalistic base container that the MATRIX provides with just the operating system and minimum library and driver requirements to ensure it will work seamlessly. Users can modify this environment, and when ready, load it into the MATRIX for use as a standard environment.
- **Container Export:** Containers can be exported for inference or other production deployment requirements.
- **Customize Pre-Build Environments:** Users have the option to customize an existing pre-built environment to suit their workflow. These customized environments can then be saved as a template for future projects.

## Smart Resourcing

Typically an AI developer or data scientist knows how many GPUs or GPU memory modules they would like to leverage, but does not know (or does not want to know) the underlying resource allocation state at any given time. For example, take the following scenario:

*When 4 GPUs are needed, if there are 4 local GPUs (all on the same server) available, then allocate to my workload. If there are only 2 local GPUs and 2 remote GPUs available, then let me know that is all that is available and allow me to proceed or not.*

The MATRIX makes this experience automatic and transparent, optimizing ideal GPU topologies ahead of less-idea GPU topologies—to strike an important balance between resource efficiency, developer productivity, and ease of use.

**Figure 6:** Workspace Creation Smart Resourcing Slider

The screenshot shows the 'CREATE WORKSPACE' interface in the MATRIX bitfusion FLEX application. The interface has a dark blue header with the 'MATRIX | bitfusion FLEX' logo and navigation tabs for 'NODES', 'WORKSPACES', 'DATA', 'IMAGES', and 'USERS'. A user profile icon is in the top right. The main content area is a light gray box with a dark blue header bar that says 'CREATE WORKSPACE' and a toggle for 'EZ Mode ON'. Below this, there are three main sections: 'Name Your Workspace' with a text input field containing 'Squishy'; 'Choose Initial Workspace Image' with a dropdown menu showing 'TensorFlow'; and 'GPU Resources' with a horizontal slider. The slider has a scale from 0 to 8. A blue segment from 0 to 4 is labeled 'Local GPUs: 4', and a yellow segment from 4 to 6 is labeled 'Remote GPUs: 2'. A blue dot is positioned at the 6 mark on the slider. At the bottom of the form is a yellow 'DONE' button.

## Data Volumes

The data required for Deep Learning and AI workloads often comes from many different sources—online and offline, external and internal, bulk files and file systems, etc. The MATRIX simplifies the process by allowing administrators to define one or many network-attached data locations to be mapped into containers. As long as the underlying host machines have access to the data location, your containers can have access as well. This makes it simple for AI developers and data scientists to access all the data that they need in a direct and seamless way.

In addition to allowing for unlimited data volume mapping flexibility, the MATRIX solution supports a local NFS filesystem on each node. This default option provides a standard location that is workload location-agnostic. No matter where a Deep Learning workload is being run (including if it is running across multiple servers), the processes have fast, local access to the data required to get the job done.



## On-Premise GPU Cloud

Given the incredible promise of Deep Learning and AI, businesses are looking to rapidly enable an on-premise “GPU Cloud” or “GPU over Fabrics” as a service for various business groups, departments, and geographies. The MATRIX includes comprehensive group-based user management for enabling broad, self-service access to the compute and container services for running Deep Learning and AI workloads, while also maintaining a strong administrator/regular user separation.

Depending on your business policies, processes, and industry standards, our flexible approach to user management can be customized to fit your unique situation to create an ideal balance between centralized control and decentralized self-service.

## Interactive Workspaces

Interactive Workspaces, powered by Jupyter iPython notebooks, provide the fastest way for an AI developer or data scientist to get started using Deep Learning and AI.

The MATRIX does not limit the number of users you can have using a MATRIX cluster—we base our pricing and scaling around infrastructure footprint. This is because we believe strongly that a Deep Learning and AI infrastructure investment should be a solution shared broadly across an enterprise for maximum impact and return on investment, and should not be taxed on a per-developer or per-data-scientist basis.

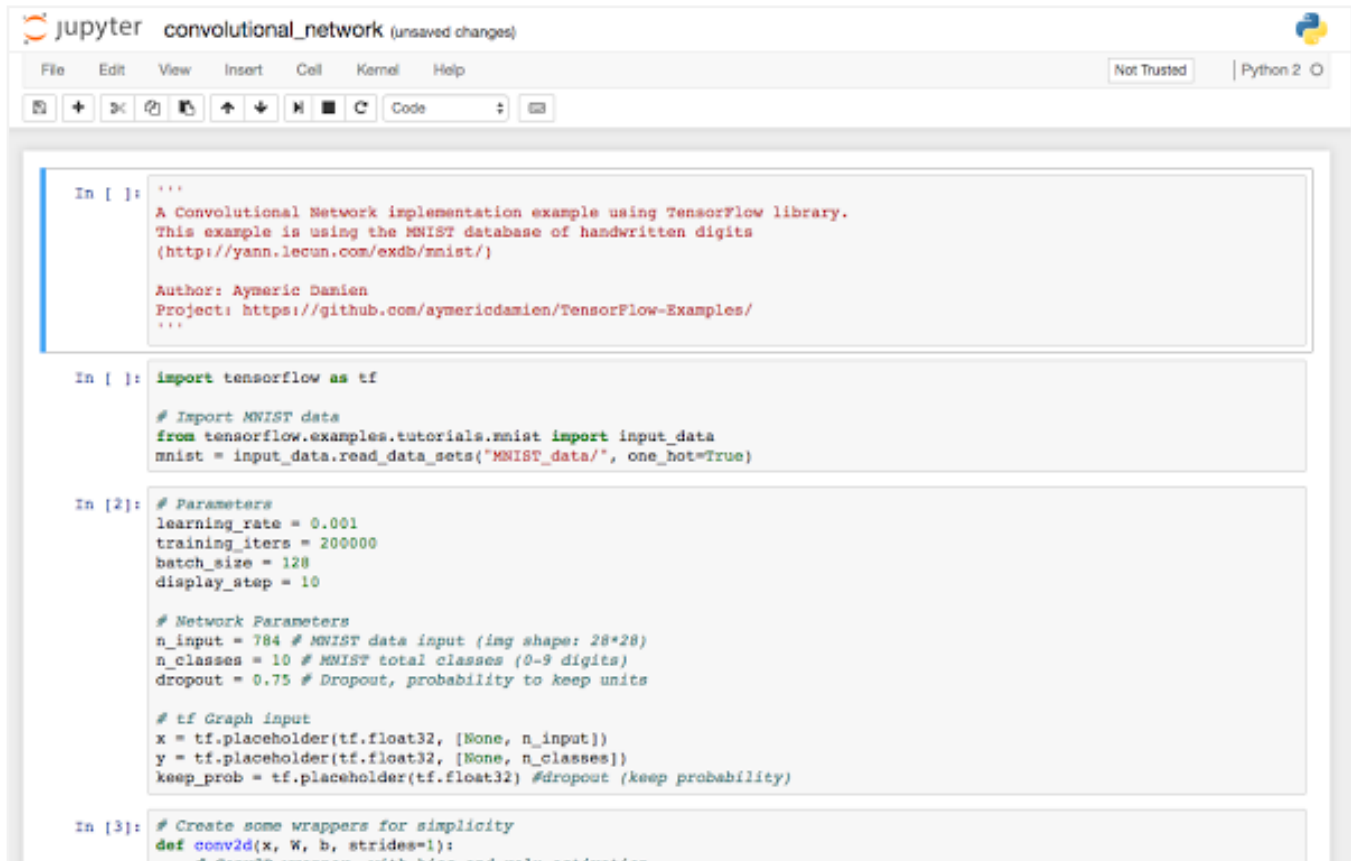
**Figure 7:** Workspace Details View

The screenshot displays the MATRIX bitfusion FLEX interface. The top navigation bar includes links for NODES, WORKSPACES, DATA, IMAGES, and USERS. The main content area is titled "DETAILS GPU workspace L1+R1" and shows the user "admin" created the workspace 11 hrs ago. Below this, the workspace configuration is listed: Ubuntu 14.04, TensorFlow 1.0.0, CUDA 7.5, and CuDNN 5.1. The interface is divided into three sections: Devices, Volumes, and Applications. The Devices section shows LOCAL GPUs and REMOTE GPUs, both with a "Kepler Tesla K80" configuration. The Volumes section shows a table with one volume named "sono" at path "/asdi/asdi/asdi/" with 3 attached workspaces, 12.1 TB used, and 15.0 TB total. The Applications section shows a Jupyter icon with a "GO" button and a "COPY LINK" button.

Name	Path	Attached Workspaces	Used	Total
sono	/asdi/asdi/asdi/	3	12.1 TB	15.0 TB

Containers are loaded in a matter of seconds with all the Deep Learning frameworks, data science libraries, and GPU drivers pre-installed. Jupyter is only a click away. The exact resourcing you desire, whether CPU-only, a partial GPU, one GPU, or multiple GPUs, is decided by a slider. More GPU resourcing? Slide right. Less GPU resourcing? Slide left. Smart Resourcing ensures you have the ideal GPU topology. The underlying Container Management and Bitfusion Core compute virtualization feature will take care of the rest.

**Figure 8:** Jupyter iPython Notebook



```
jupyter convolution_network (unsaved changes)
File Edit View Insert Cell Kernel Help Not Trusted Python 2
In [ ]: """
A Convolutional Network implementation example using TensorFlow library.
This example is using the MNIST database of handwritten digits
(http://yann.lecun.com/exdb/mnist/)

Author: Aymeric Damien
Project: https://github.com/aymericdamien/TensorFlow-Examples/
"""

In [ ]: import tensorflow as tf

# Import MNIST data
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)

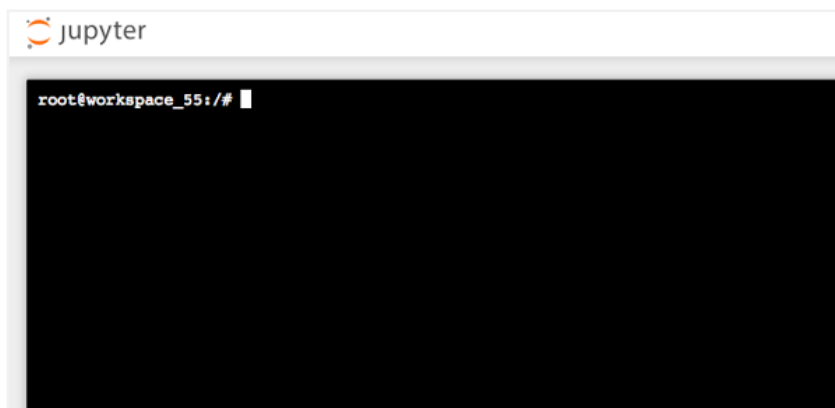
In [2]: # Parameters
learning_rate = 0.001
training_iters = 200000
batch_size = 128
display_step = 10

# Network Parameters
n_input = 784 # MNIST data input (img shape: 28*28)
n_classes = 10 # MNIST total classes (0-9 digits)
dropout = 0.75 # Dropout, probability to keep units

# tf Graph input
x = tf.placeholder(tf.float32, [None, n_input])
y = tf.placeholder(tf.float32, [None, n_classes])
keep_prob = tf.placeholder(tf.float32) #dropout (keep probability)

In [3]: # Create some wrappers for simplicity
def conv2d(x, W, b, strides=1):
    # conv2d returns a convolved matrix
```

**Figure 9:** Jupyter Terminal



```
jupyter
root@workspace_55:/#
```

## Next Steps

- **Schedule a Demo:** Reach out to us for a one-on-one consultation and software demonstration to explore how the MATRIX can drive improved productivity, time to market and overall results.  
<https://goo.gl/vPVMnd>
- **Virtualization White Paper:** Interested in more information on performance and compute virtualization? Contact us above and inquire about our Virtualization White Paper that goes into more detail on the performance and scaling characteristics of the MATRIX solution.