Fuzzy Fusion of Decisions from Heterogeneous Deep Machine Learning Models

Grant Scott, Alex Yang & Bryce Murray

June 23, 2019

University of Missouri – Data Science and Analytics

University of Missouri

Mizzou Team

Organizer

- **Grant Scott**
  - Assistant Professor, EECS
  - Director, Data Science and Analytics, MUII

Content Creators

- **Alex Yang**
  - PhD Candidate
  - Mizzou Electr Engr & Comp Sci
- **Bryce Murray**
  - PhD Candidate
  - Mizzou Electr Engr & Comp Sci

Synopsis

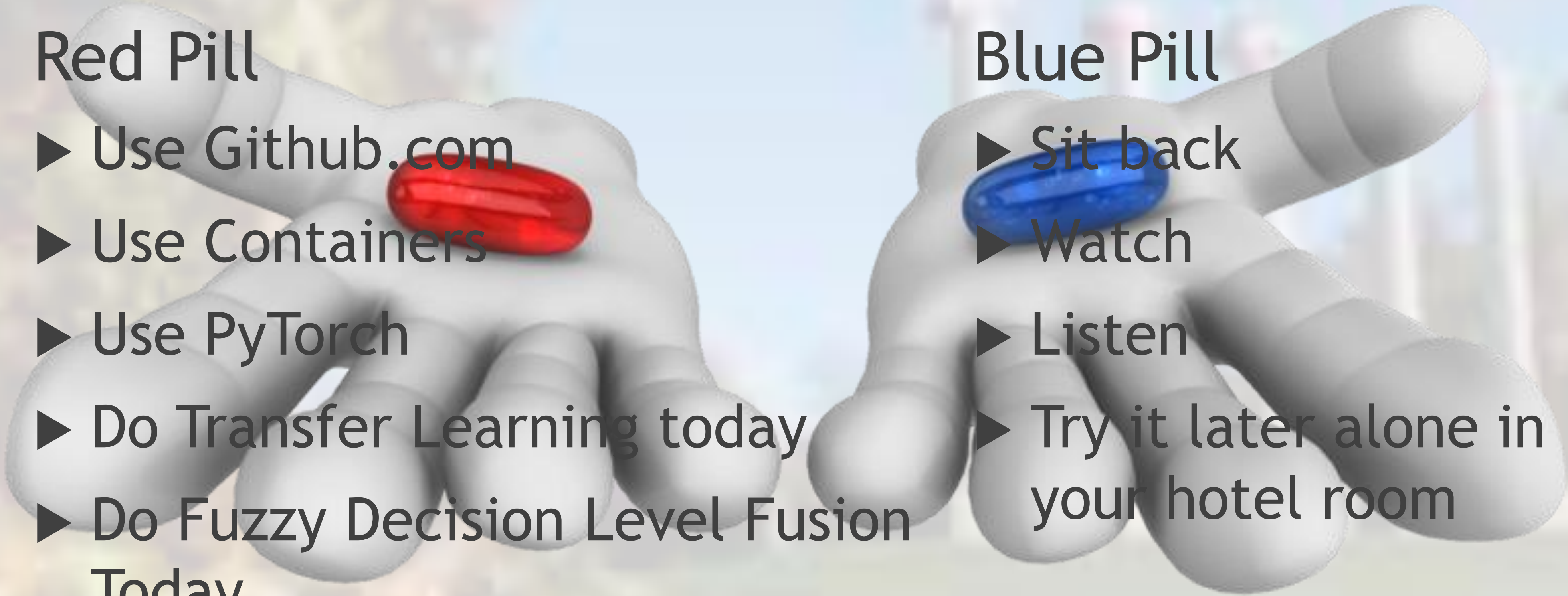
- Brief introduction to ~~TensorFlow and Keras~~→ PyTorch
- Deep learning models and transfer learning techniques
- Fuzzy machine learning model fusion

The tutorial session will be broken into these three portions, each of which culminates in code examples that can be immediately migrated from the tutorial to the participants’ own research thrusts (theories and applications).

Setup!

Containers, Git, and Jupyter … oh my!

Choose your tutorial adventure!



Red Pill

- Use Github.com
- Use Containers
- Use PyTorch
- Do Transfer Learning today
- Do Fuzzy Decision Level Fusion Today

Blue Pill

- Sit back
- Watch
- Listen
- Try it later alone in your hotel room

Getting Set Up

Network:
JWMarriott_Conference

Password:
fuzzieee2019

Computing Environment

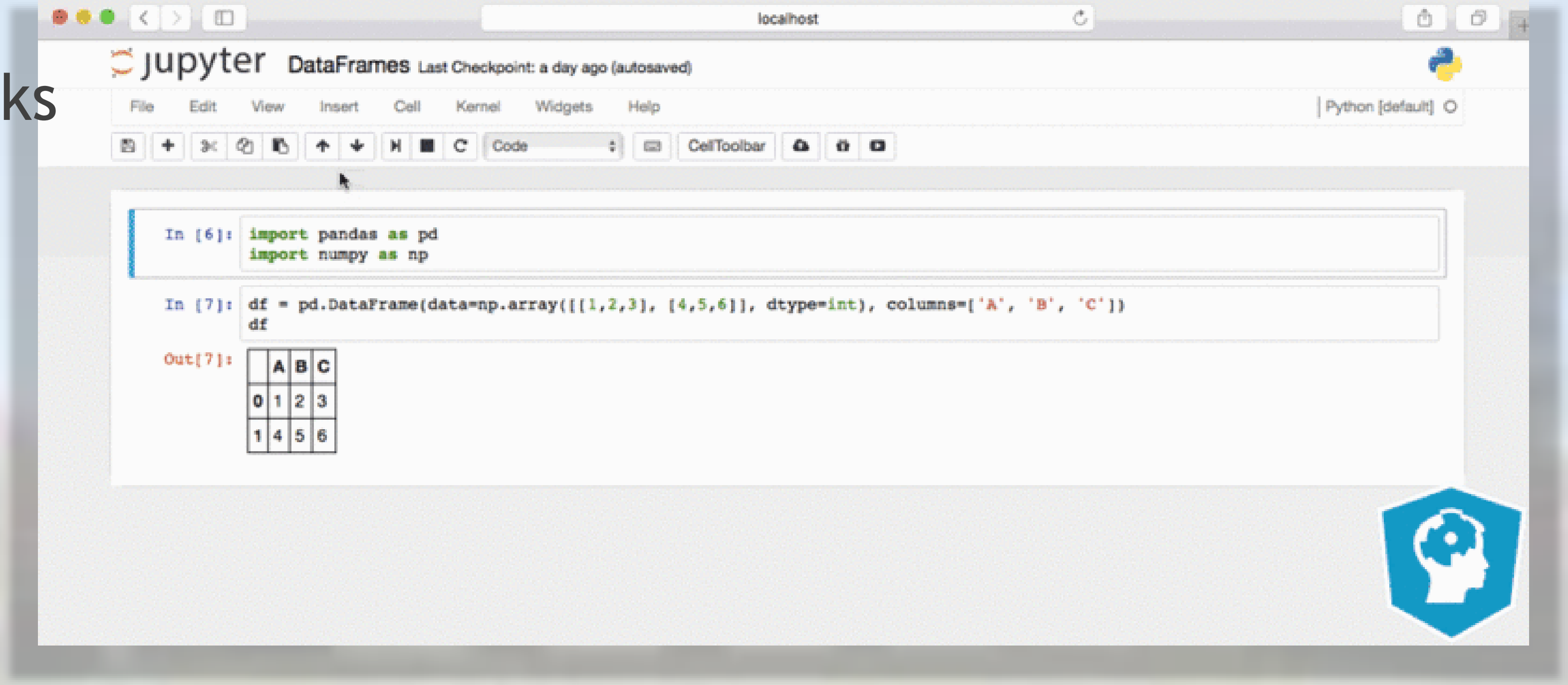
- Required Equipment for Interactive Learning
  - Laptop with Google Chrome Browser
  - Working internet connection
- Github account credentials (github.com)
  - To be used to authenticate into a MU DSA Learning Environment during tutorial
  - Fork and Clone of tutorial repository
  - Saving your work from the tutorial

Activity #0

- Sign-in to Github.com
  - Or create a new github.com account

Jupyter : Interactive Learning and Coding

- Formerly iPython notebooks



```
In [6]: import pandas as pd
        import numpy as np

In [7]: df = pd.DataFrame(data=np.array([[1,2,3], [4,5,6]], dtype=int), columns=['A', 'B', 'C'])
        df

Out[7]:
```

	A	B	C
0	1	2	3
1	4	5	6

- Jupyter Notebook Tutorial

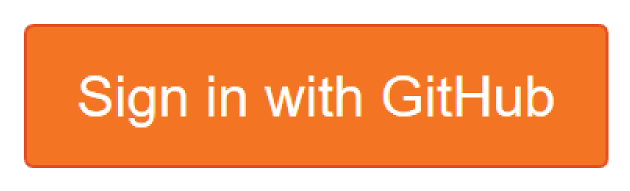
https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook

Accessing the Computational Intelligence Environment

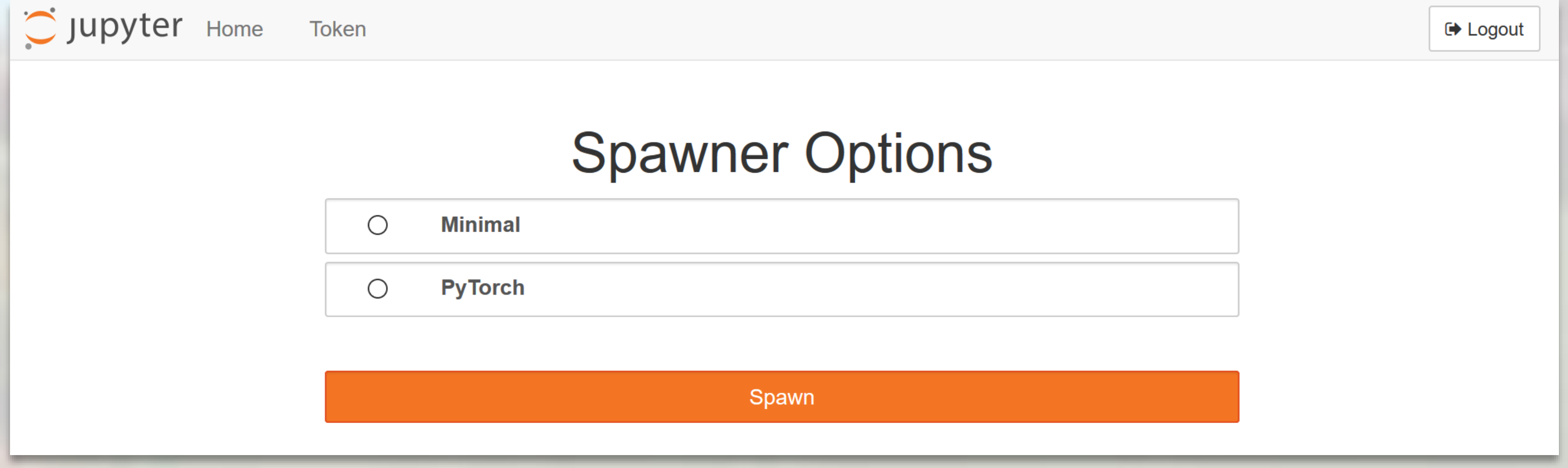
- In Google Chrome access the Tutorial system at:

  ieee.dsa.missouri.edu

- Click



- On the resulting container launch page, choose PyTorch and click Spawn



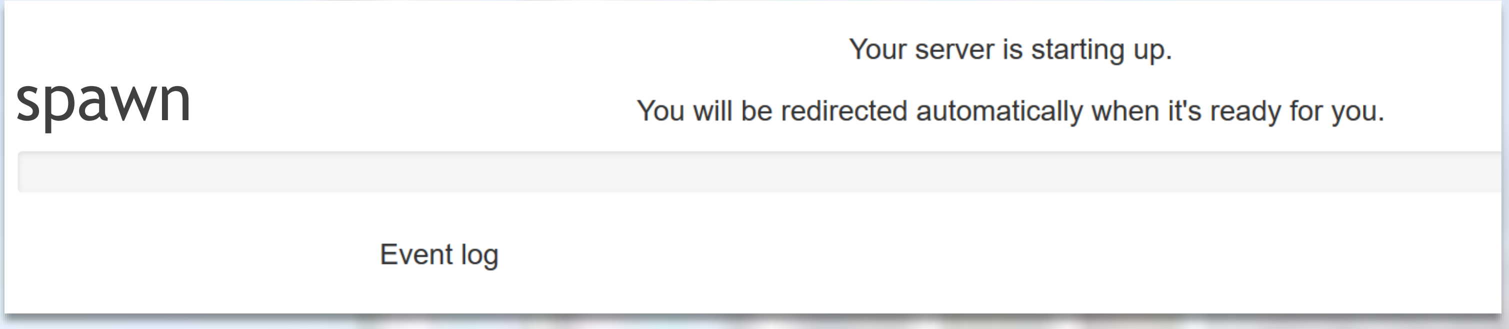
Spawner Options

○ Minimal

○ PyTorch

Accessing the Tutorial Content

- Wait for the container to spawn



Your server is starting up.

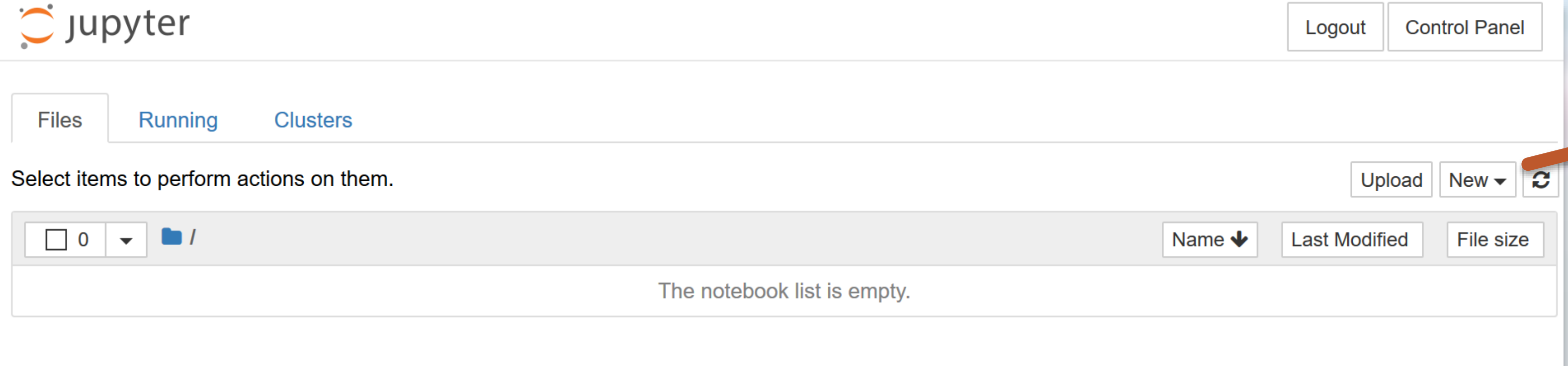
You will be redirected automatically when it's ready for you.

… while the container is spawning (may take a minute or four)

- Fork the following repository on Github.com to your personal Github account
  - https://github.com/scottgs/FuzzyFusion_DeepLearning_Tutorial

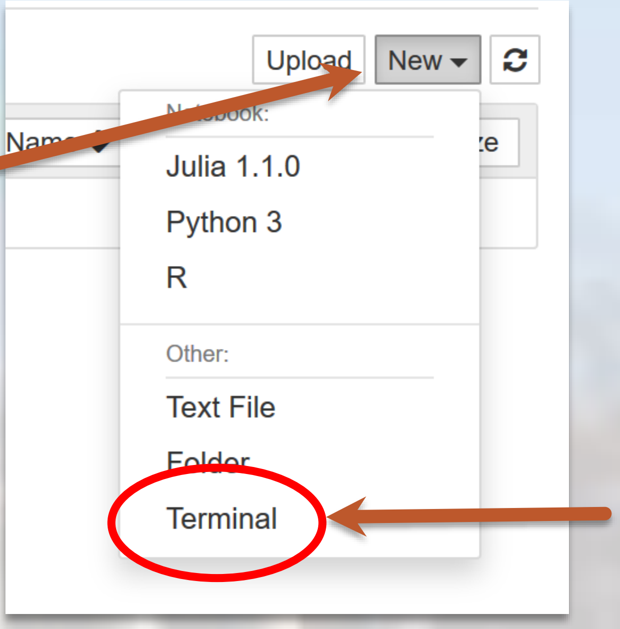
Accessing the Tutorial Content

- Open the Terminal in Jupyter

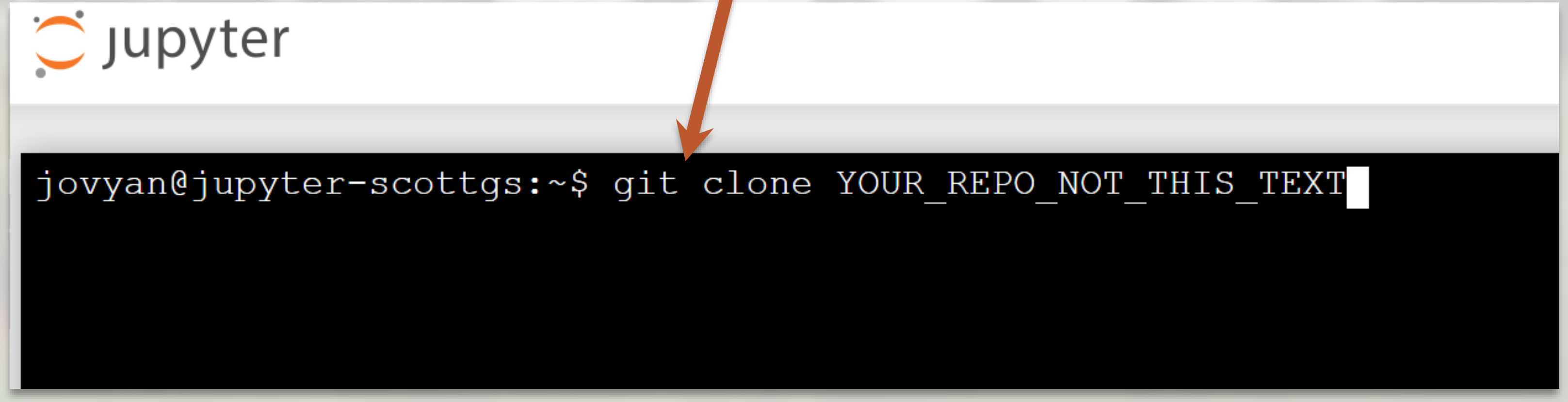


Select items to perform actions on them.

The notebook list is empty.



- Clone your fork of the repository on Github.com
  - Choose the https option if you are not familiar with PKI
  - Choose the SSH option if you are familiar with PKI



```
jovyan@jupyter-scottgs:~$ git clone YOUR_REPO_NOT_THIS_TEXT
```

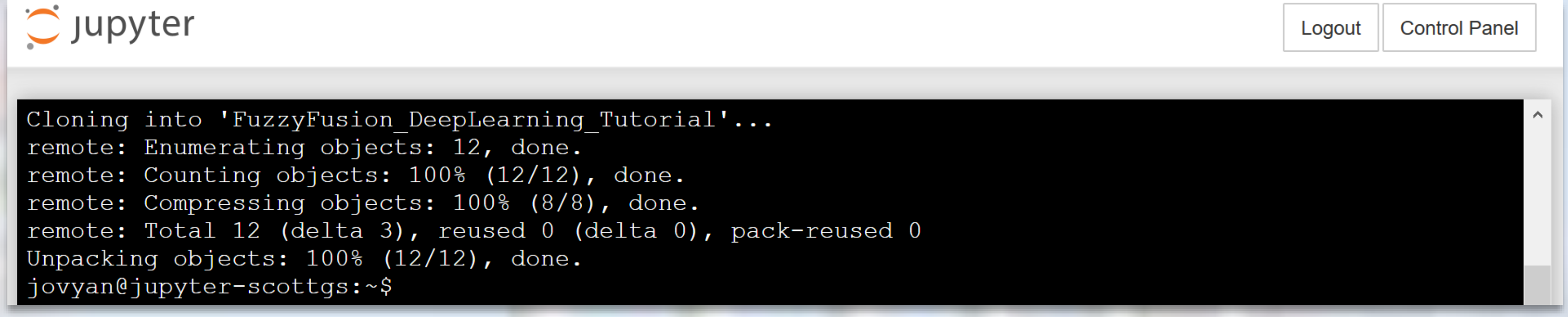
You (may) have the tutorial!

- IF cloned THEN

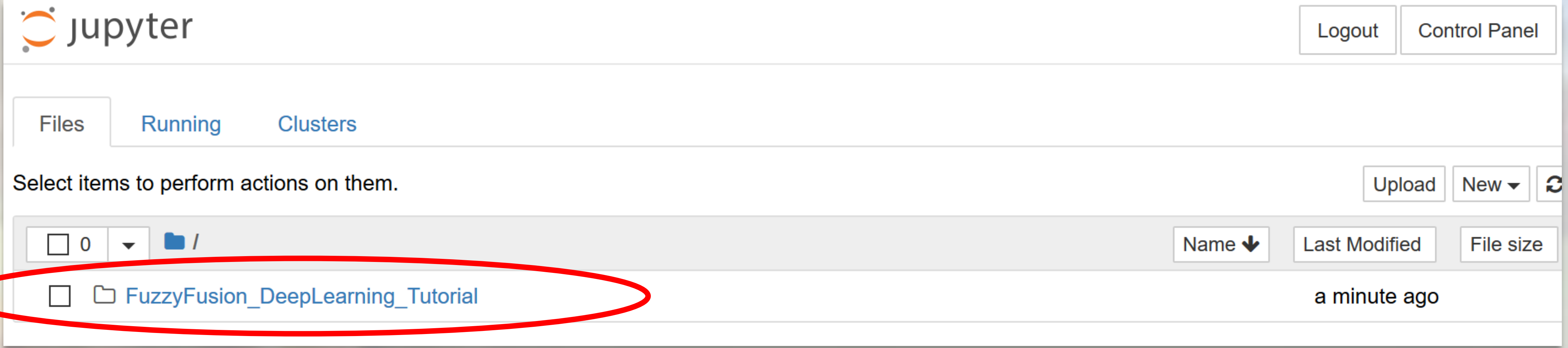
  Folder

- ELSE

  Raise Hand



```
Cloning into 'FuzzyFusion_DeepLearning_Tutorial'...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 12 (delta 3), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (12/12), done.
jovyan@jupyter-scottgs:~$
```



Select items to perform actions on them.

□ FuzzyFusion_DeepLearning_Tutorial a minute ago

Part 1

Introduction to PyTorch

PyTorch (https://pytorch.org)

Deep Learning Platform that Provides a Seemless Path from Research Prototyping to Production Deployments

- PyTorch is the Python interface to Torch library
- Navigation : Part1
  - *IntroductionToPyTorch.ipynb*

PyTorch Key Take Aways

- High-level library for computing over *n-dimensional* tensors
  - N-dimensional tensor often represents states of NN architectures
- Tensors are the main computational data structure
- Autograd is magic!
  - PyTorch use automatic differentiation to compute gradients of computational functions used in forward pass of NN models
- PyTorch *nn* package provides predefined layer types for constructing neural networks

PyTorch for Future Investigation

- Glow – Accelerating DML
  - https://github.com/pytorch/glow
- PyTorch Geometric – Irregular input (graphs, point clouds, manifolds)
  - https://github.com/rusty1s/pytorch_geometric
- Skorch – SciKit Learn Compatibility
  - https://github.com/skorch-dev/skorch

Part 2

Deep Learning Models and Transfer Learning Techniques

ResNet50 in PyTorch

So easy a caveman could do it!

```
def build_res50():
    model = models.resnet50(pretrained=True)
    model.fc = torch.nn.Sequential(
        torch.nn.Linear(
            in_features=2048,
            out_features=101
        ),
        torch.nn.ReLU()
    )
    return model.cuda()
```

- Model object from built-in definition
- Weights seeded with ImageNet
- Replace fully-connected layers (fc) with a new fully-connected set of layers and new classifier dimensions

PyTorch Deep Learning and Transfer Learning

ResNet50 – for smarties (not dummies)

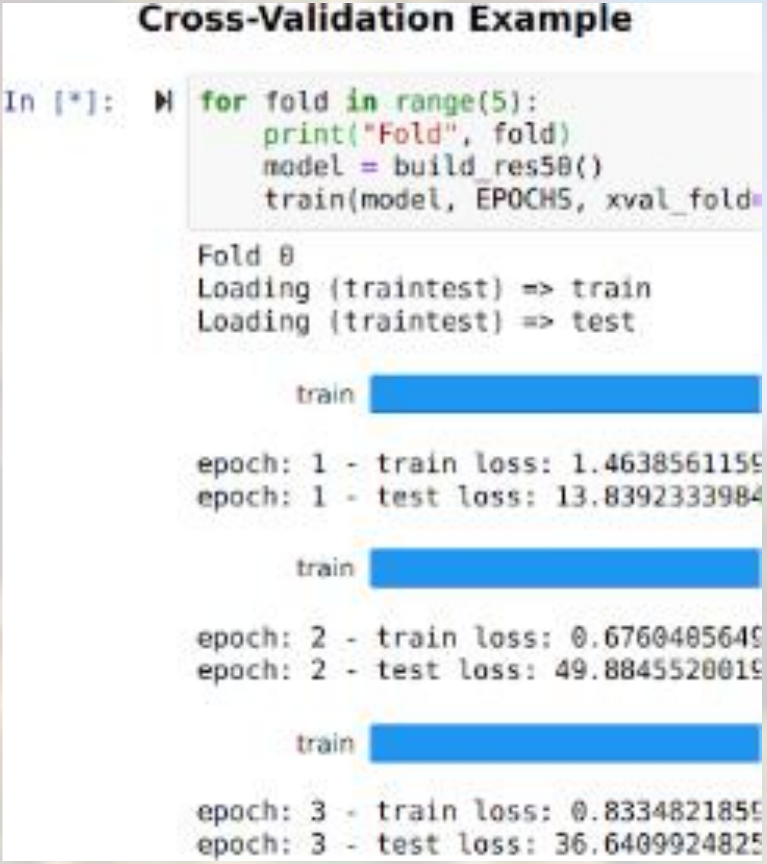
- Instantiating ResNet50 deep convolutional neural network
  - Loading ImageNet weights, then fine tuning (Transfer Learning)
- Navigation : Part2
  - *PyTorchResNet50.ipynb*

Image Loading & Transform for NN Input

- Operations may be limited to types of data they are applicable to:
  - Tensors
  - Images
  - Etc.
- We will transform images into normalized tensors

```
transform_pipe =
torchvision.transforms.Compose([
    torchvision.transforms.Resize(
        size=(299, 299)
    ),
    torchvision.transforms.ToTensor(),
    torchvision.transforms.Normalize(
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    )
])
```

Cross-Validation can be used to determine best hyperparameters for final training



Cross-Validation Example

```
In [*]:    for fold in range(5):
               print("Fold", fold)
               model = build_res50()
               train(model, EPOCHS, xval_fold=
```

```
Fold 0
Loading (traintest) => train
Loading (traintest) => test
```

```
epoch: 1 - train loss: 1.4638561159
epoch: 1 - test loss: 13.8392333984
```

```
epoch: 2 - train loss: 0.6760485649
epoch: 2 - test loss: 49.8845520019
```

```
epoch: 3 - train loss: 0.8334821859
epoch: 3 - test loss: 36.6409924825
```

- ▶ What is the right optimizer?
- ▶ Learning rate?
- ▶ Comparison of architectures?

```
for fold in range(5):
    print("Fold", fold)
    model = build_res50()
    train(model, EPOCHS, xval_fold=fold,
            lr=1e-3, is_inception=False)
```

Key Takeaways

- Extending the *torch.utils.data.Dataset*
- Normalizing input data
- Train / Test Splits and Cross-Validation
- Training DML
  - Hyperparameters
  - Adam vs SGD

Part 3

Fuzzy Machine Learning Model Fusion

Python Fuzzy Choquet Integral

Fusing heterogeneous architectures at the Decision-In – Decision-Out phase

- Custom Python Choquet Library
  - Computing the fuzzy measure
  - Decision-level Information Fusion
- Navigation : Part3
  - *FuzzyFusion.ipynb*

Choquet Fuzzy Integral
Sugeno Lambda Fuzzy Measure

- Each network has a per-class performance accuracy
- This becomes our trust in this network’s classification of each class
- Use input densities to solve for the Sugeno Lambda Fuzzy Measure

```
chi = ChI.ChoquetIntegral()
densities = [a,b,c]
chi.train_chi_sugeno(densities)

print(chi.fm)

fzyfused = chi.chi_sugeno(data)
```

Choquet Fuzzy Integral
Data-driven Fuzzy Measure

- Learn the fuzzy measure based on training samples
- Training data includes
  - Heterogeneous network outputs for an image
  - Expected class label for an image

```
chi = ChI.ChoquetIntegral()
chi.train_chi_quad(
           train_data,
           label_data)
print(chi.fm)

fzyfused = chi.chi_quad(data)
```

Key Takeaways

- Fusion can enhance system performance by combining decision-level output from multiple DML
  - In this case, three deep convolutional neural networks
- Fuzzy measure of Choquet Integral can be computed from input densities or learned from data

The Ending Credits

Special Thanks To …

Matthew Blackwood - MUII – DSA

Will Starms - MU Center for Geospatial Intelligence

MU Informatics Institute – Data Science and Analytics (MUII – DSA)

Program for

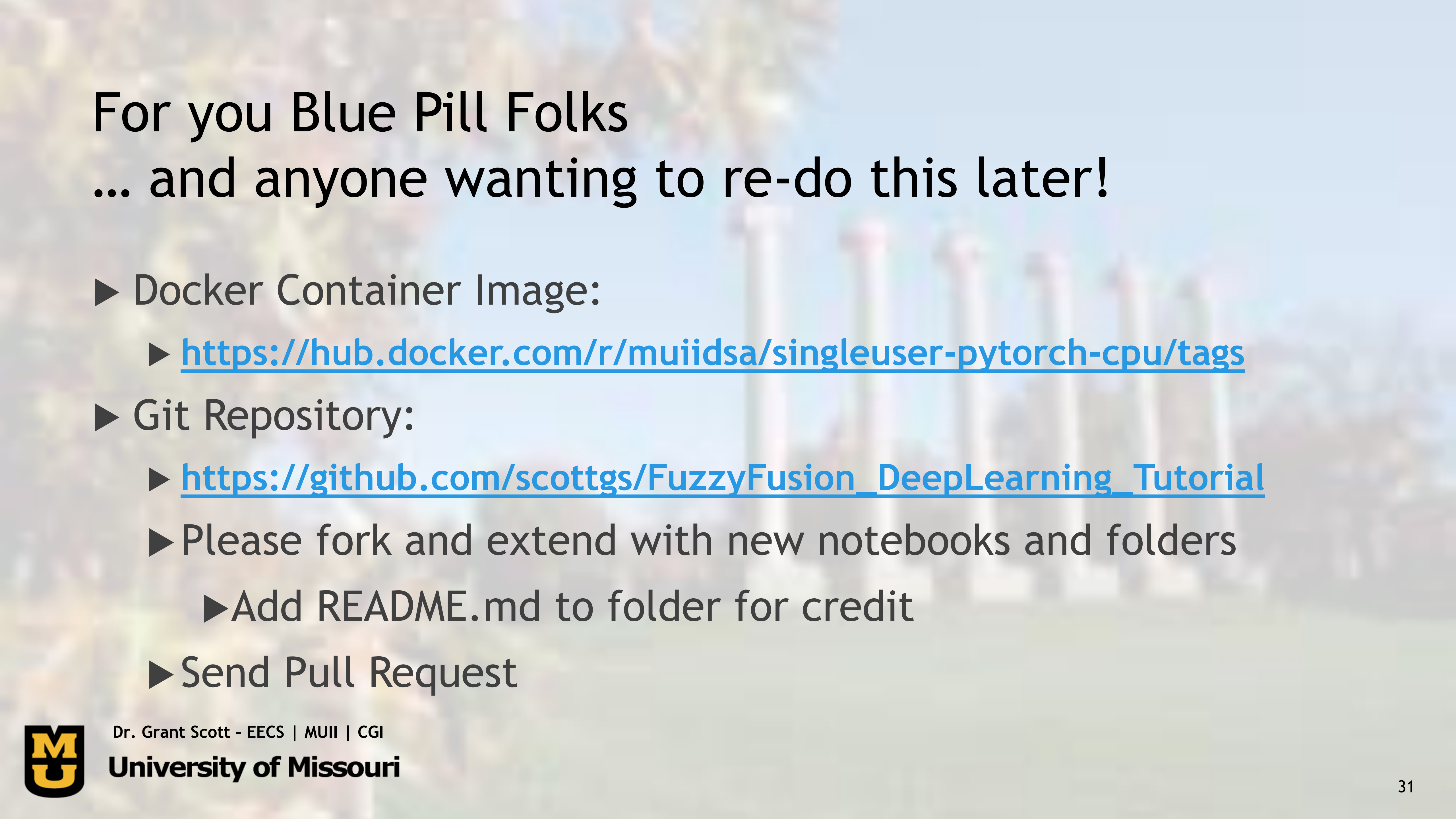
- Developing the PyTorch Container images
- Hosting the tutorial learning infrastructure

Hungry for more?
Watch Github.com for updates on Wednesday!

- Go see Bryce Murray’s talk:

Transfer Learning for the Choquet Integral

  - Date / Time : Monday, **3:57** PM
  - Session: AGG-1
  - Room: Frontenac



For you Blue Pill Folks
… and anyone wanting to re-do this later!

- Docker Container Image:
  - https://hub.docker.com/r/muiidsa/singleuser-pytorch-cpu/tags
- Git Repository:
  - https://github.com/scottgs/FuzzyFusion_DeepLearning_Tutorial
  - Please fork and extend with new notebooks and folders
    - Add README.md to folder for credit
  - Send Pull Request

Relevant Links and Additional Resources

Technology Links

- PyTorch
  - https://pytorch.org/get-started/locally/
- Py Convex Optimization (CVXOPT)
  - https://cvxopt.org/
- Nvidia Containers
  - https://www.nvidia.com/en-us/gpu-cloud/containers/
- Github (Git VCS)
  - https://guides.github.com/activities/hello-world/
- Fuzzy Fusion Library
  - https://github.com/Blake-Ruprecht/Fuzzy-Fusion

Citations / Papers

- Training Deep Convolutional Neural Networks for Land-Cover Classification of High-Resolution Imagery
  - https://doi.org/10.1109/LGRS.2017.2657778
- Fusion of Deep Convolutional Neural Networks for Land Cover Classification of High-Resolution Imagery
  - https://doi.org/10.1109/LGRS.2017.2722988
- Enhanced Fusion of Deep Neural Networks for Classification of Benchmark High-Resolution Image Data Sets
  - https://doi.org/10.1109/LGRS.2018.2839092
- Data-Driven Compression and Efficient Learning of the Choquet Integral
  - https://doi.org/10.1109/TFUZZ.2017.2755002
- Enabling Explainable Fusion in Deep Learning with Fuzzy Integral Neural Networks
  - https://arxiv.org/abs/1905.04394

Questions
Or
Discussions