Using the Jupyter notebook in practice

ATLAS - Google technical meeting, 9th June 2021

Fang-Ying Tsai

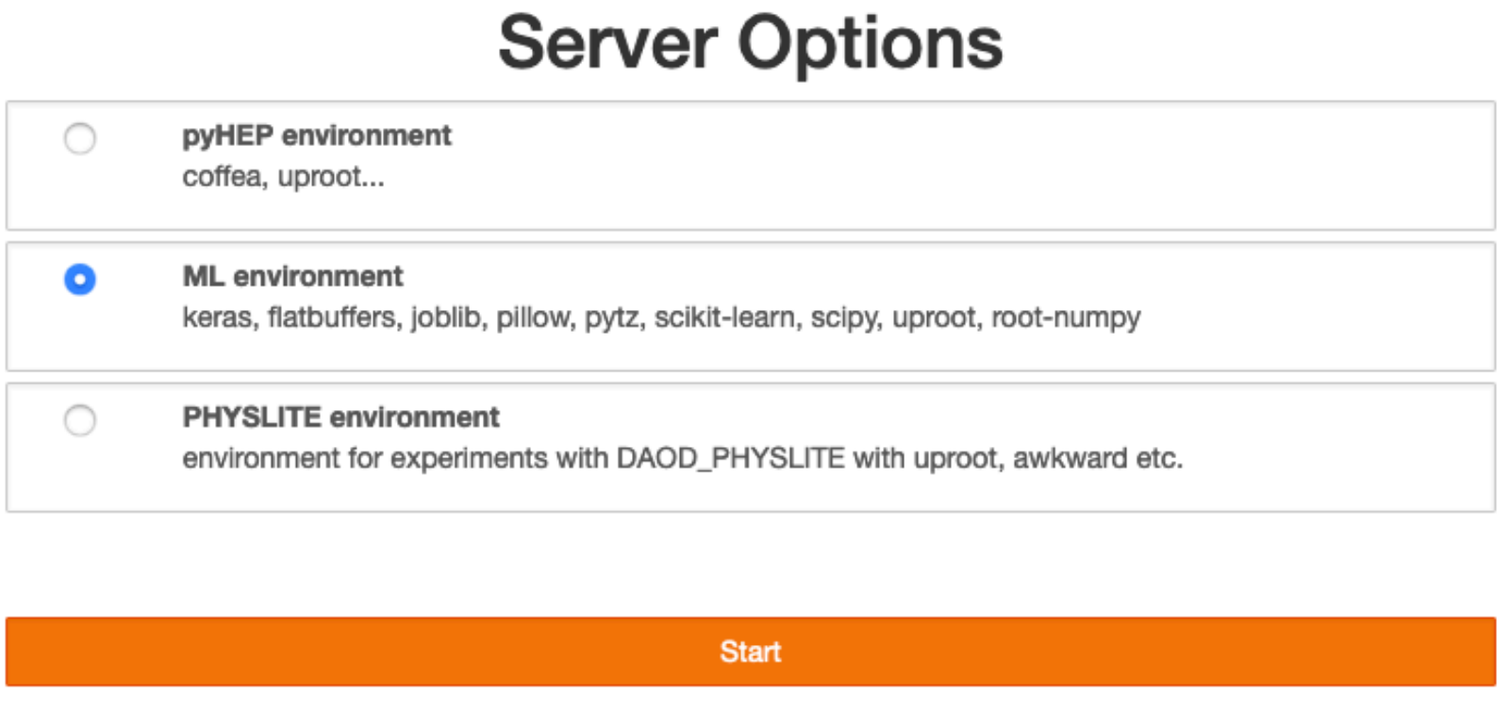
Project Participants for the VH(bb) analysis and Google-ATLAS R&D

John Hobbs, Giacinto Piacquadio,
Torre Wenaus, Alexei Klimentov

Demonstration

- What I want to have is displaying all the plots at the same time, so that I can easily scrutinize them to debug or to checkout the performance.
  - I'm used to using a web browser, e.g. here. Once I have an html template, there is no much effort to show them all when changing different hyper-parameters.
- What I have done with the notebook?

Step 1: Launch a server with an image that includes all the packages needed by users framework. (contact Fernando if a specific env.)



Demonstration

Step 2: Importing my DSNNr_lib.py where all my functions are defined. This is tricky because notebooks are not python files, but luckily, following up on this instruction I can import a Jupyter notebook as a module.

```
import DSNNr_lib as DSNNr
```
```
importing Jupyter notebook from DSNNr_lib.ipynb
```

Step 3: Loading the model and selected dataset.

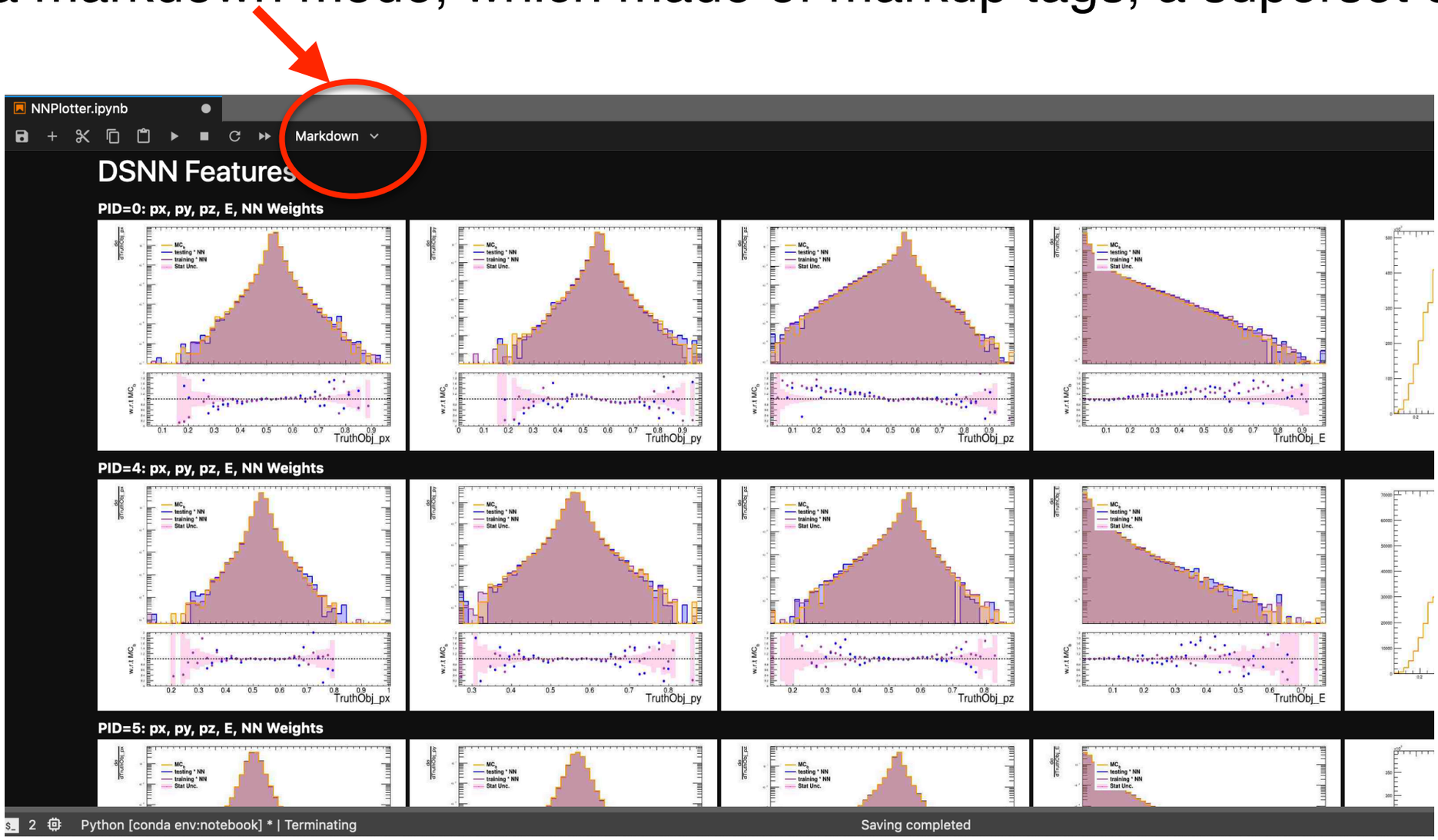
- upload the trained model and datasets manually through the left sidebar where you can also find what files in the work area.
- download a dataset stored in the GCS, see the backup.

```
[7]: #Loading the model
     model = keras.models.load_model("saved_models/"+"BatchSize_20000"+".ckpt")
```
```
2021-06-10 03:16:20.673899: I tensorflow/core/platform/cpu_feature_guard.cc:142] This TensorFlow binary is o
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2021-06-10 03:16:20.687774: I tensorflow/core/platform/profile_utils/cpu_utils.cc:104] CPU Frequency: 229999
2021-06-10 03:16:20.689508: I tensorflow/compiler/xla/service/service.cc:168] XLA service 0x564eb4d28740 ini
2021-06-10 03:16:20.689537: I tensorflow/compiler/xla/service/service.cc:176]   StreamExecutor device (0): H
```
```
[8]: #Getting datasets: Nominal_v3_2.root (3.2G) and Altv3_2.root (3.3G) for about 2M events each.
     (MCa, MCb, MCa_weights, MCb_weights, maxObjCount) = DSNNr.get_data("Nominal_v3_2.root", "Altv3_2.root")
```
```
get_data:  Loading data
```

Demonstration

Step 4: Plotting code! After my normal/standard plotting script, I save the histograms into .jpg files. (as I would like to display them in a way I want in one cell instead of lining up on the same row.)

- once the histograms are saved, I execute the cell by loading them in a markdown mode, which made of markup tags, a superset of HTML.



Next

1. I find the notebook is useful when it comes to debug. I don't have to rerun the whole script but a target cell where I made a typo.
2. But there seems to be not easy to share plots with people who I'm working with to discuss?
   - Perhaps a slideshow functionality can help?
     e.g. http://www.youtube.com/watch?v=UhidS7fZZko&t=40m45s
3. Job submission! I will also train my NN model using Google resources and make comparisons.

Backup: accessing datasets

Rucio download from CVMFS:

1. Asking Cedric/Mario a special permissions/quota on GOOGLE_EU2.
2. Uploading the certificates (in ATLAS standard way, here)
3. Checking the fgal2 dependency (it should be built in the image, so contact Fernando if not).
   - to see if it works by running *conda list | grep gfal*

```
(notebook) jovyan@jupyter-fangying:~$ conda list | grep gfal
gfal2                     2.19.0          h2073588_0    conda-forge
gfal2-util                1.6.0           pyhd8ed1ab_0          conda-forge
python-gfal2              1.10.0          py38h112ff3b_0        conda-forge
```

4. Then I'm able to download through the following setup.

```
(notebook) jovyan@jupyter-fangying:~$ export CERN_USER=fatsai
(notebook) jovyan@jupyter-fangying:~$ export RUCIO_ACCOUNT=$CERN_USER
(notebook) jovyan@jupyter-fangying:~$ export RUCIO_AUTH_TYPE=x509_proxy
(notebook) jovyan@jupyter-fangying:~$ rucio download user.sjiggins:user.sjiggins.mc15_13TeV.
410470.PhPy8EG_A14_ttbar_hdamp258p75_nonallhad.evgen.EVNT.e6337.VHbb_DSNNr_ttbar-v3_1_Lep --nrandom 1 --rse GOOGLE_EU
```

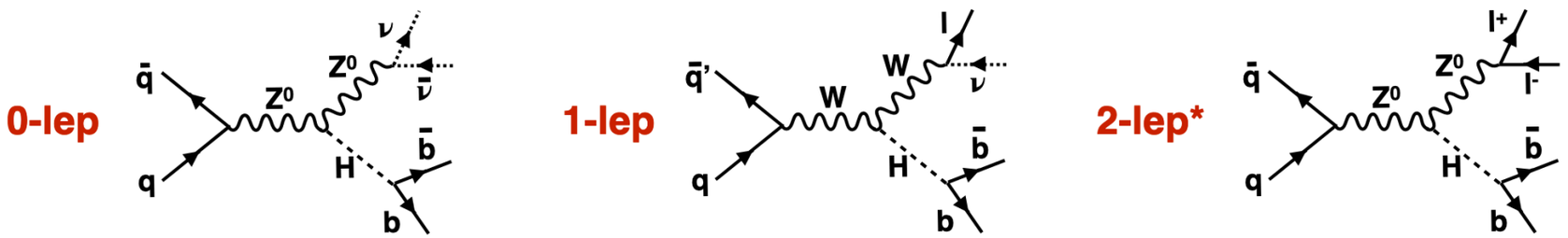
Nikolai's Rucio notebook

Contact Nikolai if you want to know the second way. The page is missing at the time I'm making this slide.

Backup

Deep Sets Neural Network reweighting (DSNNr)

- Why are we interested in DSNNr?
  - Looking for a CPU/GPUs intensive ML task for the US ATLAS Google project as a use case.
  - Ultimately we want to achieve high utilization of using CPU/GPUs for ML work.
  - We can have a generic classification for VH(bb) as well as VH(cc) analyses in both boosted and resolved regimes.
    → generate a mapping function between two MC configurations that is independent of the reconstruction scheme.



- Our framework is built based on the ParticleFlow Neural Network.
  - It's the application of the algorithm.
  - It's a fresh technique in the analyses/ATLAS.

Deep Sets Neural Network reweighting (DSNNr)

- DSNN architecture (ref.)

$$f(\{x_1, x_2 ... x_M\}) = F(\Sigma_{i=1}^{M} \Phi(x_i))$$

- Permutation invariance sets. The NN will learn the same when permuting the input objects.

- Φ: to embed datasets into a vector space from x elements → R^{ℓ}

$$\{x_1, x_2 ... x_M\} \xrightarrow{\text{mapping through a NN}} \begin{bmatrix} \text{obj}_1 \\ \text{obj}_2 \\ ... \\ \text{obj}_\ell \end{bmatrix}$$

- Adding up all particle representations in multi-dimensional space.
- F: applying a nonlinear transformation yielding event representations from R^{ℓ} elements →Y

$$\begin{bmatrix} \text{obj}_1 \\ \text{obj}_2 \\ ... \\ \text{obj}_\ell \end{bmatrix} \xrightarrow{\text{The choice of the distributions through a NN}} \text{F\_signal, F\_background}$$