UNCERTAINTY QUANTIFICATION

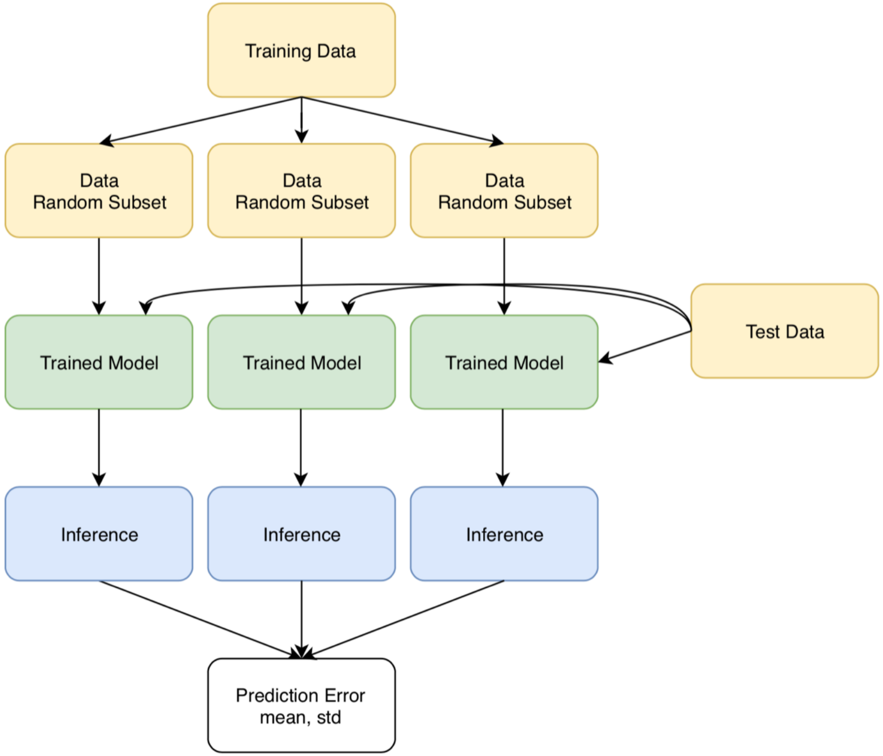
HARRY YOO
Software Engineer
Data Science and Learning Division

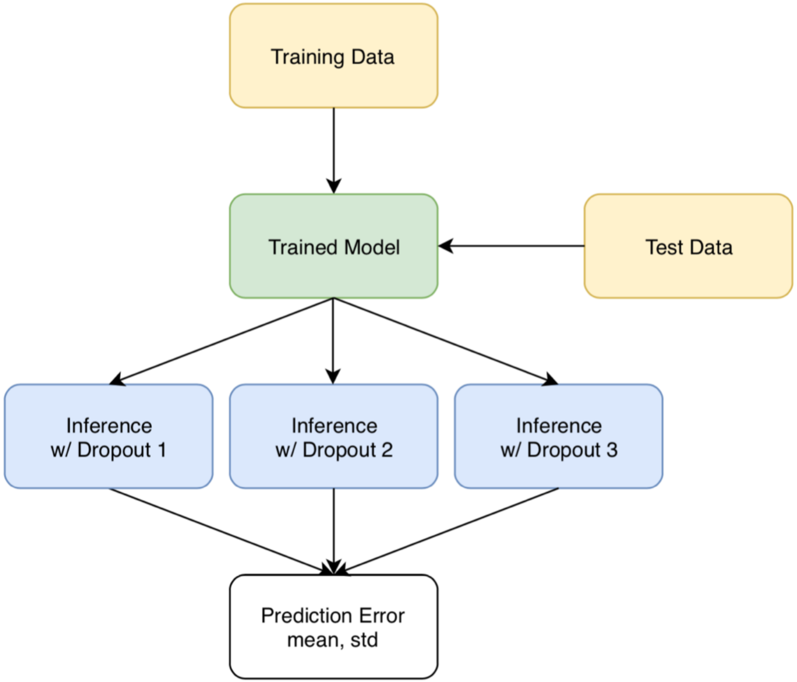
hsyoo@anl.gov
Mon, Jan 14th 2019,
ECP Annual Meeting

OUTLINE

- Uncertainty Quantification Methods: Bootstrap, Dropout
- Comparing DL UQ Variants
- CANDLE UQ Utilities
- CANDLE UPF workflow
- Demo

Bootstrap vs Dropout





Bootstrap UQ for DL

- We sample, with repetition, N = 100 training samples using the ALMANAC data set.
- For each sample we train a DL network to estimate a function that maps from

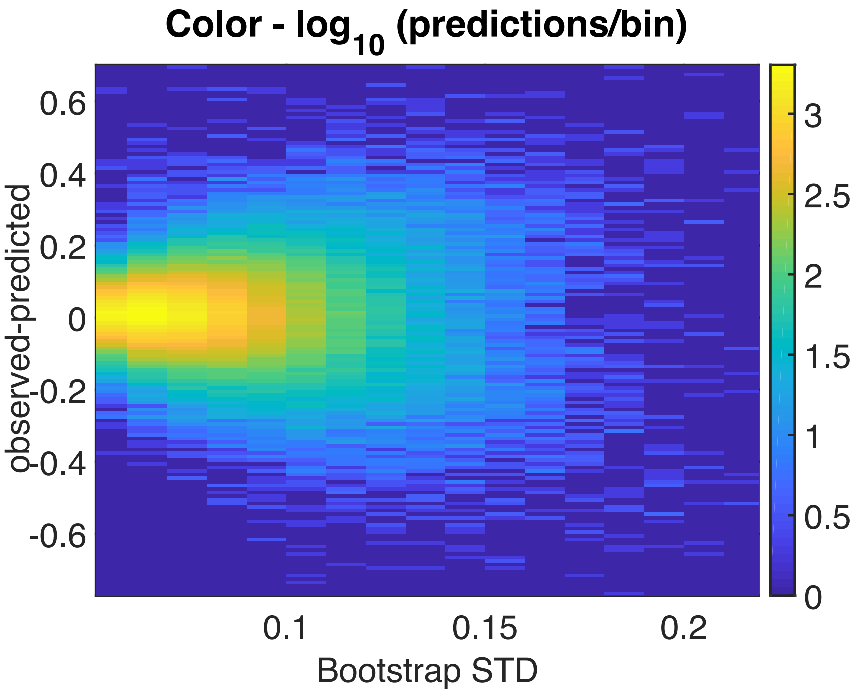
  (drug 1, drug 2, CL_GE, concentration) -> measured ALMANAC growth values
- This produces N DL functions that make growth predictions on the test set.
- For each test point we have N predictions from which we estimate the mean and standard deviation; the prediction error is (mean – measured growth)
- We analyze the correlation between the prediction error (accuracy) and the standard deviation (confidence) of the predictive distribution.

Dropout UQ for DL

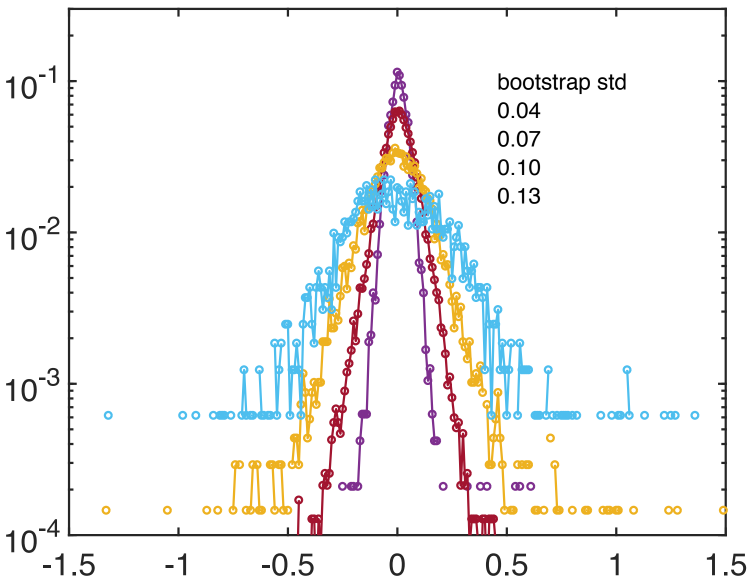
- Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning (Yarin Gal, Zoubin Ghahamani, 2016)
- https://arxiv.org/abs/1506.02142

Comparing DL UQ Variants

- Synergy for combo drugs predicted with deep learning (DL) models.
- Two UQ methods:
  - Bootstrap: bootstrap training data resulting in N models and then doing N inference runs.
  - Dropout: train once with dropout enabled and do inferencing with dropout in place N times.
- GDSC cell lines and Almanac compounds (no ground truth)
- Not all results are available (top 50K synergy samples predicted, top 50K uncertainty samples predicted)
- Files (from 01/18/2018):
  - GDSC-Bootstrap-N10-Almanac_synergy_UQ.xlsx
  - GDSC-Dropout-UQ-N100-Almanac_synergy_UQ.xlsx

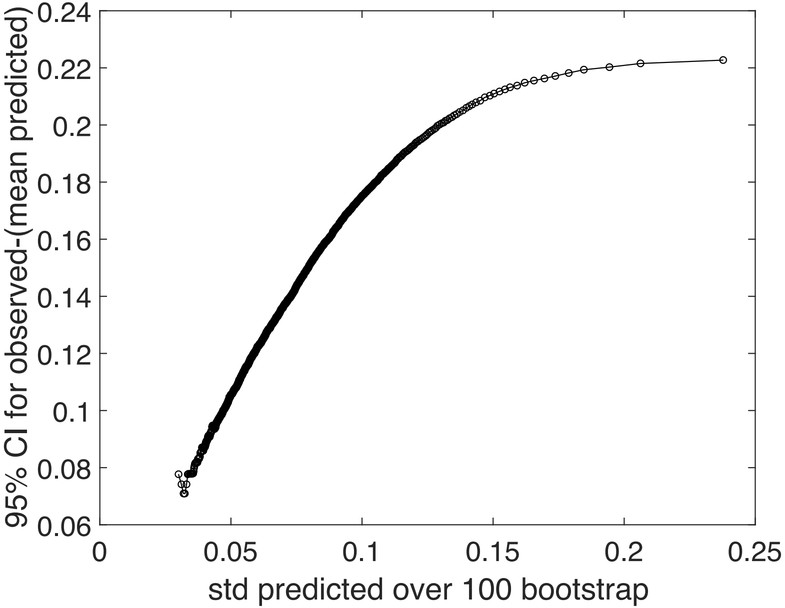


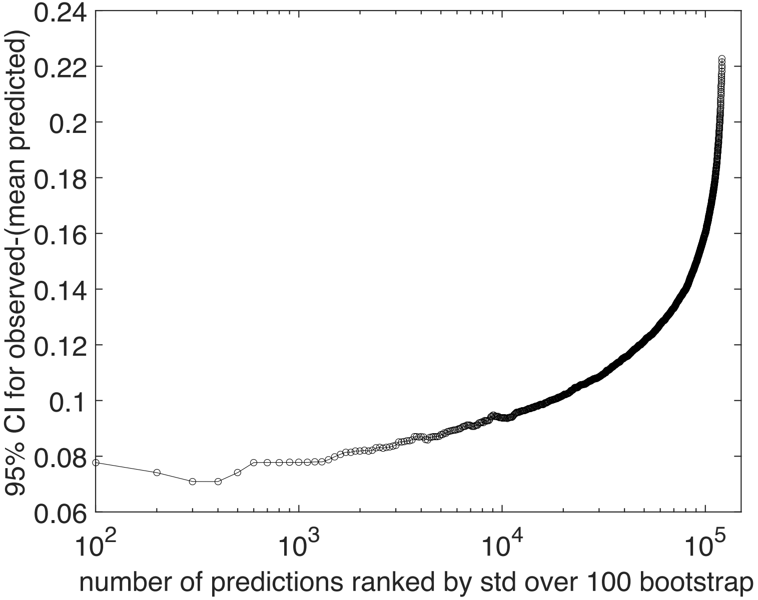
A) Bootstrap histogram in (std, error) space.



B) Probability distribution of prediction errors given bootstrap std; this is the empirical PDF for the columns in A).

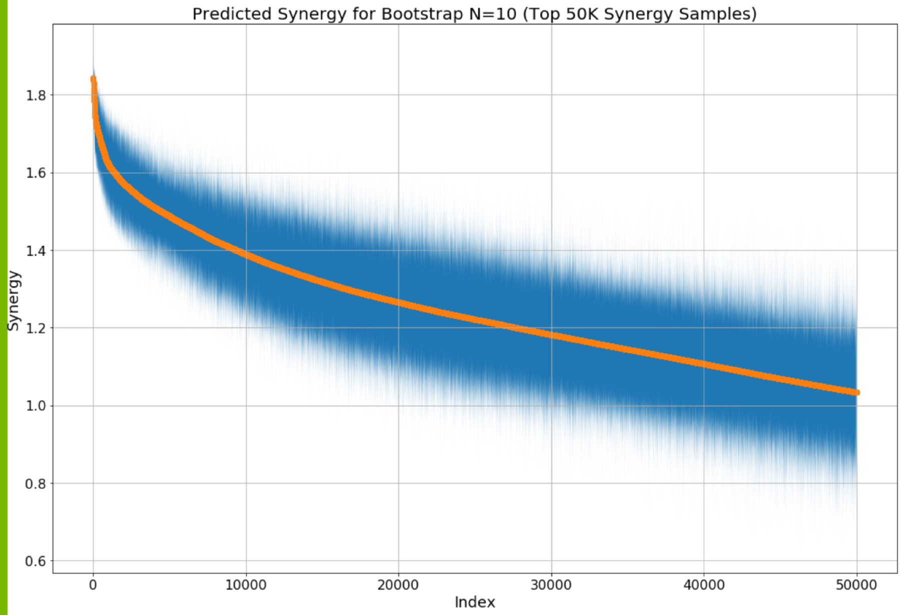
Highly confident predictions (small bootstrap std) have high accuracy (small error).

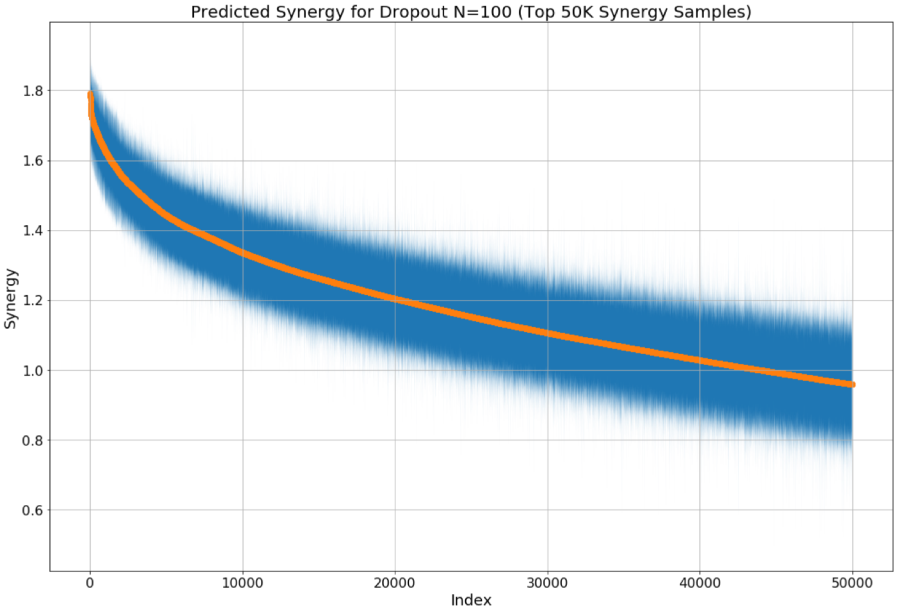




Highly confident predictions (small bootstrap std) have high accuracy (with high confidence the predictions are in a small interval around the true value).

Different methods: dropout vs bootstrap



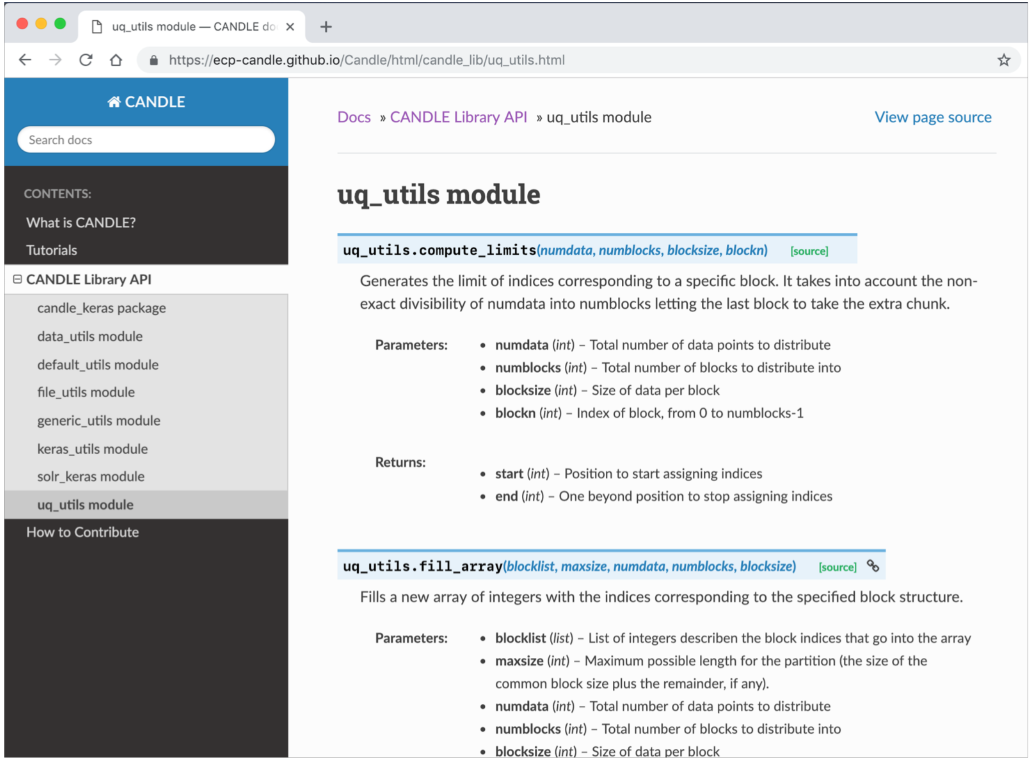


Scaling Inferencing for UQ Measurements of Drug Synergy Predictions

- 680 Drugs in Combination
- 670 Samples
- 30 Replicates

JobID	Node Size	Successful Node	Failed Node	Infers/Node:	Infers/Job	Duration	Infer/Sec/Node	Dropout Rate
X004	670	670	0	13,872,000	9,294,240,000	6:56:08	555.587	0.1
X006	670	670	0	13,872,000	9,294,240,000	6:58:10	553.875	0.2
X007	670	670	0	13,872,000	9,294,240,000	6:56:18	555.364	0.3
X012	670	670	0	13,872,000	9,294,240,000	6:57:44	554.447	0.4
X018	670	670	0	13,872,000	9,294,240,000	6:57:33	553.686	0.5
Totals	3350	3350	0	69,360,000	46,471,200,000	34:45:53	554.5918	

CANDLE UQ Utilities



uq_utils module — CANDLE do ×

https://ecp-candle.github.io/Candle/html/candle_lib/uq_utils.html

CANDLE

Search docs

CONTENTS:

What is CANDLE?

Tutorials

CANDLE Library API

candle_keras package

data_utils module

default_utils module

file_utils module

generic_utils module

keras_utils module

solr_keras module

uq_utils module

How to Contribute

Docs » CANDLE Library API » uq_utils module

View page source

uq_utils module

`uq_utils.compute_limits`(*numdata, numblocks, blocksize, blockn*) [source]

Generates the limit of indices corresponding to a specific block. It takes into account the non-exact divisibility of numdata into numblocks letting the last block to take the extra chunk.

Parameters:

- **numdata** (*int*) – Total number of data points to distribute
- **numblocks** (*int*) – Total number of blocks to distribute into
- **blocksize** (*int*) – Size of data per block
- **blockn** (*int*) – Index of block, from 0 to numblocks-1

Returns:

- **start** (*int*) – Position to start assigning indices
- **end** (*int*) – One beyond position to stop assigning indices

`uq_utils.fill_array`(*blocklist, maxsize, numdata, numblocks, blocksize*) [source]

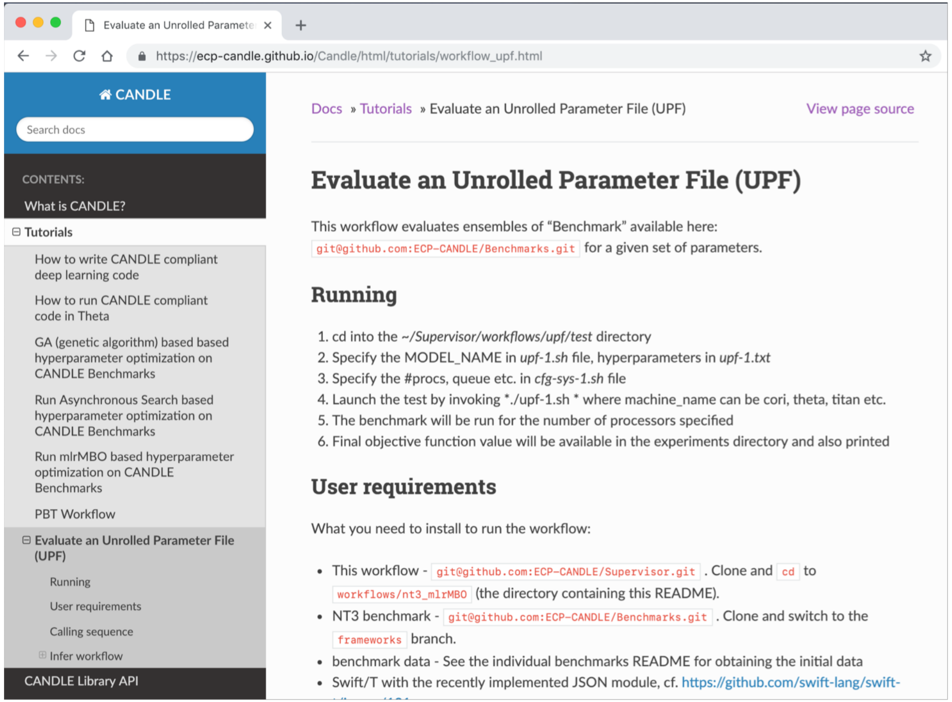
Fills a new array of integers with the indices corresponding to the specified block structure.

Parameters:

- **blocklist** (*list*) – List of integers describen the block indices that go into the array
- **maxsize** (*int*) – Maximum possible length for the partition (the size of the common block size plus the remainder, if any).
- **numdata** (*int*) – Total number of data points to distribute
- **numblocks** (*int*) – Total number of blocks to distribute into
- **blocksize** (*int*) – Size of data per block

https://ecp-candle.github.io/Candle/html/candle_lib/uq_utils.html

CANDLE UPF Workflow


Evaluate an Unrolled Paramete

https://ecp-candle.github.io/Candle/html/tutorials/workflow_upf.html

CANDLE

Search docs

CONTENTS:

What is CANDLE?

Tutorials

- How to write CANDLE compliant deep learning code
- How to run CANDLE compliant code in Theta
- GA (genetic algorithm) based based hyperparameter optimization on CANDLE Benchmarks
- Run Asynchronous Search based hyperparameter optimization on CANDLE Benchmarks
- Run mlrMBO based hyperparameter optimization on CANDLE Benchmarks
- PBT Workflow
- Evaluate an Unrolled Parameter File (UPF)
  - Running
  - User requirements
  - Calling sequence
  - Infer workflow

CANDLE Library API

Docs » Tutorials » Evaluate an Unrolled Parameter File (UPF) View page source

Evaluate an Unrolled Parameter File (UPF)

This workflow evaluates ensembles of "Benchmark" available here: `git@github.com:ECP-CANDLE/Benchmarks.git` for a given set of parameters.

Running

1. cd into the *~/Supervisor/workflows/upf/test* directory
2. Specify the MODEL_NAME in *upf-1.sh* file, hyperparameters in *upf-1.txt*
3. Specify the #procs, queue etc. in *cfg-sys-1.sh* file
4. Launch the test by invoking *./upf-1.sh * where machine_name can be cori, theta, titan etc.
5. The benchmark will be run for the number of processors specified
6. Final objective function value will be available in the experiments directory and also printed

User requirements

What you need to install to run the workflow:

- This workflow - `git@github.com:ECP-CANDLE/Supervisor.git` . Clone and `cd` to `workflows/nt3_mlrMBO` (the directory containing this README).
- NT3 benchmark - `git@github.com:ECP-CANDLE/Benchmarks.git` . Clone and switch to the `frameworks` branch.
- benchmark data - See the individual benchmarks README for obtaining the initial data
- Swift/T with the recently implemented JSON module, cf. https://github.com/swift-lang/swift-

https://ecp-candle.github.io/Candle/html/tutorials/workflow_upf.html

Steps for Dropout UQ

- Add Permanent Dropout Layers to DNN
- Add Permanent Dropout Layers to Inference Script
- Modify the JSON Model Representation

- Set Up the UPF Workflow
- Run the Experiment

Demo

THANKS