Parametrising profile likelihoods with Neural Networks

J. Araz[a], **H. Reyes-Gonzalez[b]**, S. Kraml[f], Rafal Maselek[b], W. Waltenberger[e]
[a]JLab, [b]RWTH Aachen, [c]LPSC Grenoble, [d]HEPHY ÖAW

Collaborative Research Center TRR 257
Particle Physics Phenomenology after the Higgs Discovery

Abstract:

- Full statistical models[1] encapsulate the complete information of an experimental result, including the likelihood function given observed data.
- Since a few years ago ATLAS started publishing statistical models that can be reused via the pyhf framework[2].
- In LHC reinterpretation we are often mainly interested on the profile likelihood given a signal strength.
- Computations directly using pyhf's full likelihoods can take a significant amount of time.
- To fully leverage from the precision obtained from full statistical models without compromising speed, we propose to learn the profiled likelihood functions with Neural Networks (NNs).
- We show that such functions can be well described with simple NNs, published in the ONNX format, and easily used by different reinterpretation tools.

LHC likelihoods in a nutshell

From Bayes theorem, an statistical model (SM) is defined as:

$$\underbrace{P(\Theta, x)}_{} = \underbrace{P_x(x\,|\,\Theta)}_{\text{Likelihood}}\underbrace{\pi_\Theta(\Theta)}_{\text{Prior}} = \underbrace{P_\Theta(\Theta\,|\,x)}_{\text{Posterior}}\underbrace{\pi_x(x)}_{\text{Evidence}}$$

An LHC SM usually looks like:

$$P(\boxed{\mu}, \boxed{\theta} \,|\, \boxed{\text{data}}) = \Pi_{k=1}^{n_c} P[\boxed{n_k} \,|\, \boxed{\mu} c_{i,k}(\boxed{\vec{\theta}}) N_{S,i,k}(\boxed{\vec{\theta}}) + B_{i,k}(\boxed{\vec{\theta}})] \Pi_{j=1}^{n_{syst}} G(\boxed{\theta_j^{obs}}; \boxed{\theta_j}; 1)$$

signal/control yields
Nuisance parameters (uncertainties)
(Observed) data
(Auxiliary) data
Parameters of Interest (signal strength, observables, etc.)

With SMs we perform global fits, exclude BSM models, find upper limits, search for SM deviations, etc.

The Profile Likelihood

- In new physics searches reinterpretation, we are often interested in the **Profile Likelihood (PL)**. →
- The PL is defined as a function where the nuisance parameters are fixed such that the likelihood is maximised given a signal strength μ.
- In the case of positive signal μ=1, otherwise, if data is Standard Model like μ=0.
- The PL is a function of the signal yields (data), n_s.

$$L(x\,|\,\mu; \hat{\theta}(\mu))$$

- With the PL we construct **Log Likelihood Ratio (LLR) tests**. →
- Depending on how the PL is fitted and defined, we can derive upper limits, exclusion confidence levels and discoveries.

$$t(\mu) = -2\log\frac{L(\mu; \hat{\theta}(\mu))}{L(\hat{\mu}, \hat{\theta}(\hat{\mu}))}$$

Example likelihoods

ATLAS-SUSY-2018-04

- Search for direct stau production in events with two -leptons
- Number of SRs: 2.
- DOI: 10.1103/PhysRevD.101.032009

ATLAS-SUSY-2019-08

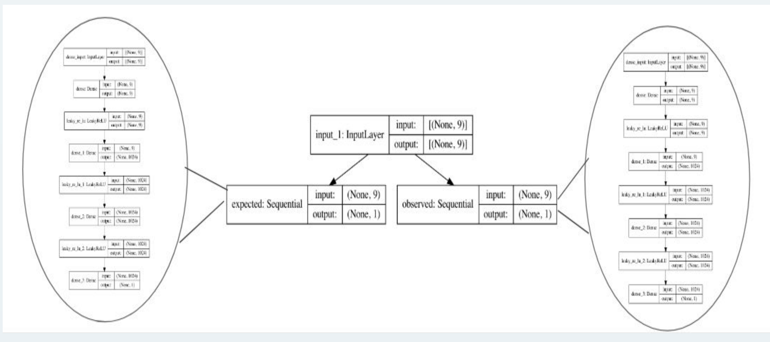
- Search for direct production of e-winos in final states with 1 lepton, MET and a Higgs boson decaying into 2 -jets
- Number of SRs : 9.
- DOI: https://doi.org/10.17182/hepdata.90607.v4

References.

[1]K. Cranmer, et. al. Publishing statistical models: Getting the most out of particle physics experiments. DOI:10.21468/SciPostPhys.12.1.037
[2]L. Heinrich, et. al. pyhf: pure-Python implementation of HistFactory statistical models. DOI: 10.21105/joss.02823
[3]J. Araz. Spey: smooth inference for reinterpretation studies. ArXiv: 2307.06996 .

General strategy

- **Sampling**: MCMC Metropolis-Hastings runs towards the min(L) and max(L), to cover the full parameter space.
- The **Input** is n_s and the **Output** is -log(L) .
- **Training:** All models were Multi-Layer Perceptrons (MPE) trained using Mean Squared Error loss function, ADAM optimiser and LeakyReLU activation functions. Data was divided as training-validation-test on a 60-20-20 scheme. Observed and Expected Likelihood functions are trained separately.
- **Testing:** The accuracy of the NN models was measured with the Mean (MAPE) and Max (MaxAPE) Absolute Percentage Error.
- **Saving:** After training, the best models for each analysis are ensemble together and saved as ONNX files.
- **Usage:** The NN likelihoods will be available for statistical studies via an Spey[3] backend.

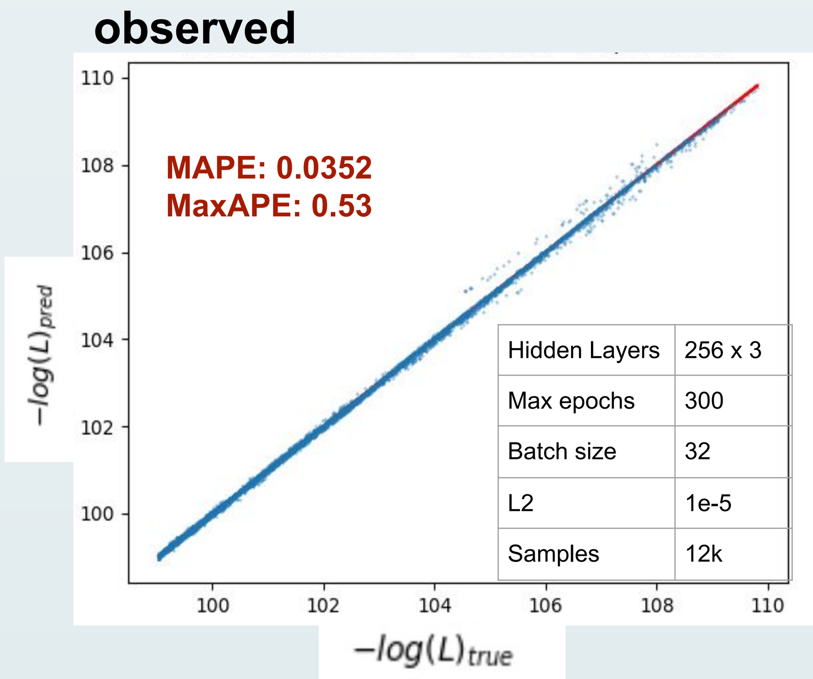


```python
from spey import BackendBase, ExpectationType
from spey.base.model_config import ModelConfig

import onnx, onnxruntime
import numpy as np


class MLlikelihoods(BackendBase):
    """
    Spey plug-in to evaluate machine learned likelihoods

    Args:
        signal_yields (``List[float]``): Signal yields
        network_path (``str``): Path to the network onnx file.
    """
```

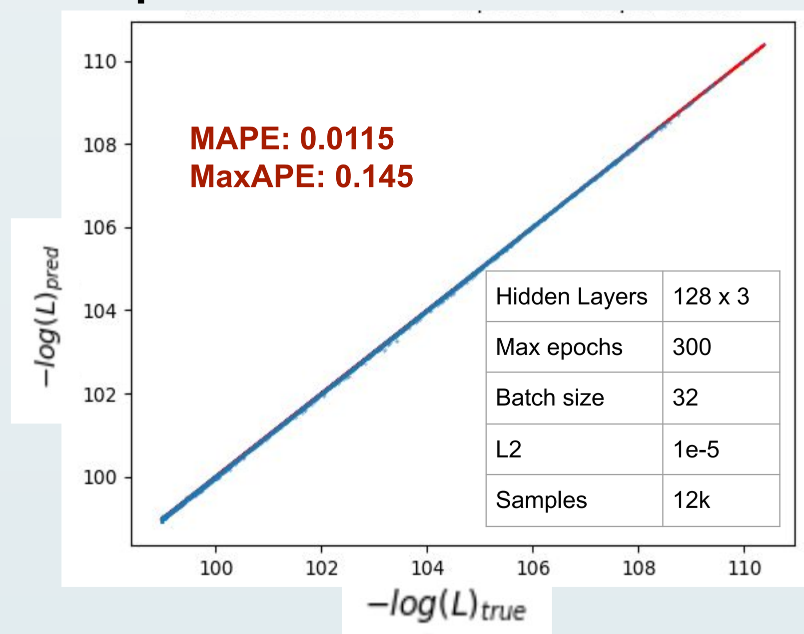
Results

ATLAS-SUSY-2018-04

observed

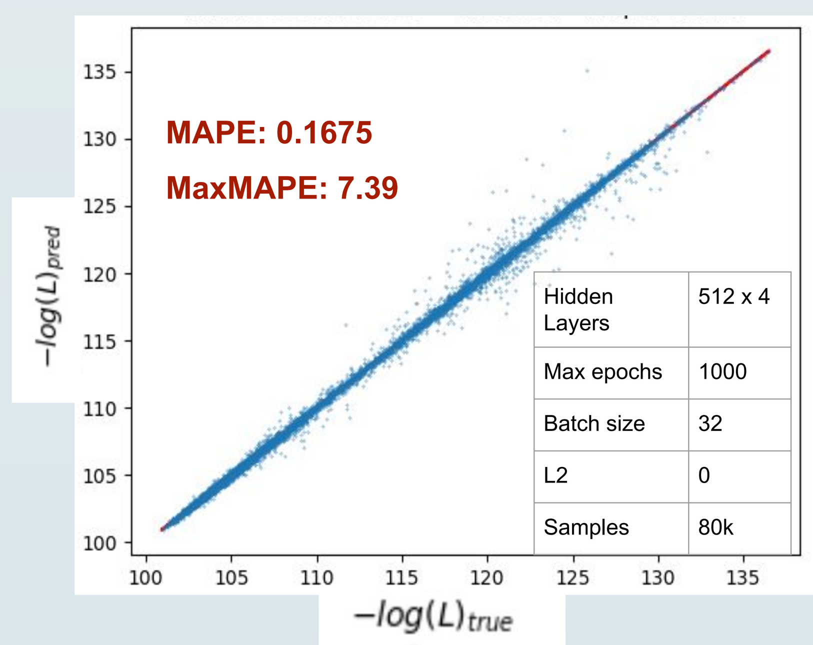


expected

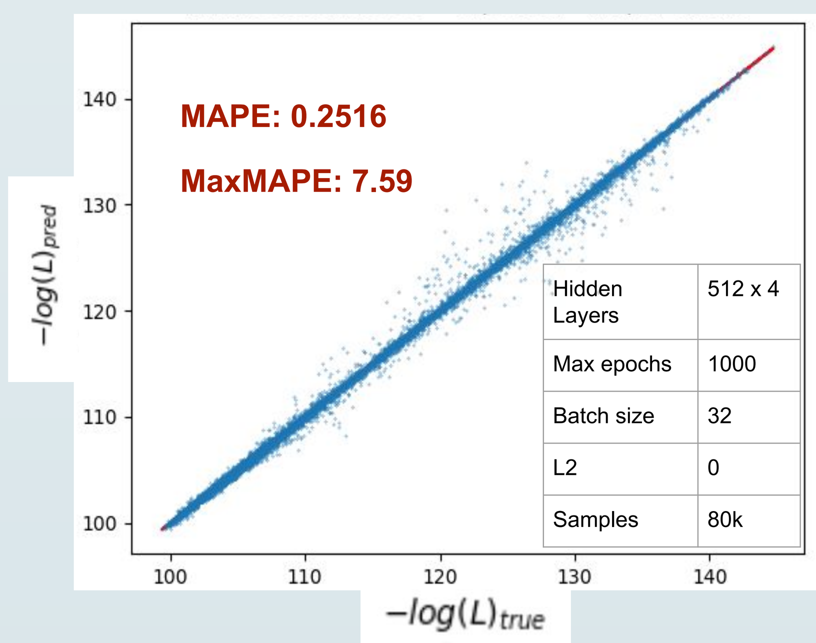


ATLAS-SUSY-2019-08

observed



expected



Conclusions

- Multivariate phenomenological studies require efficient handling of likelihoods.
- NNs provide an orders of magnitude faster alternative for LHC likelihood publication. **From several minutes to less than a second per point!**
- Profile likelihoods are easily learnable by NNs.
- They can easily be integrated into modern reinterpretation frameworks, e. g. SModelS MadAnalysis5
- As a plus, we can obtain max(L) directly from the gradients of the NNs (differential programming).