Machine Learning for the precision determination of Parton Distribution Functions

Jesús Urtasun Elizari

Supervised by Prof. Stefano Forte and Dr. Stefano Carrazza

IFAE Seminar - Barcelona, January 2020

UNIVERSITÀ
DEGLI STUDI
DI MILANO

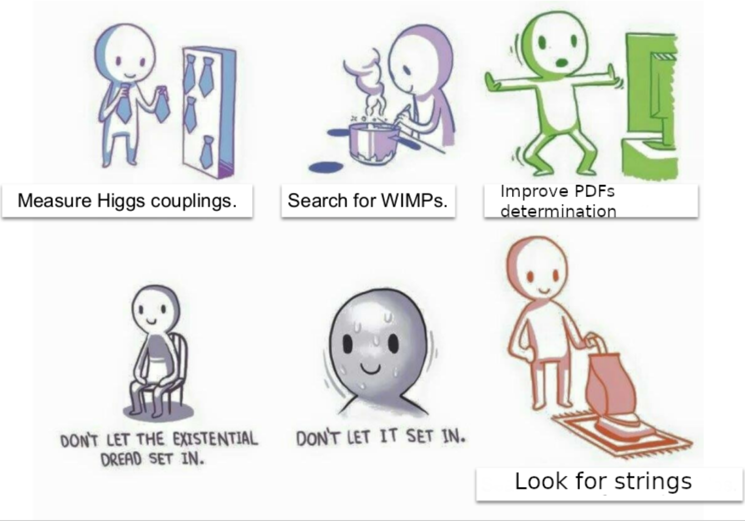
Machine Learning • PDFs • QCD

European
Research
Council

How to adult…


SHOP FOR A NEW TIE.
MAKE MACARONI
DO CARDIO.
DON'T LET THE EXISTENTIAL DREAD SET IN.
DON'T LET IT SET IN.
VACUUM THE RUG.

How to be a mature physicist…



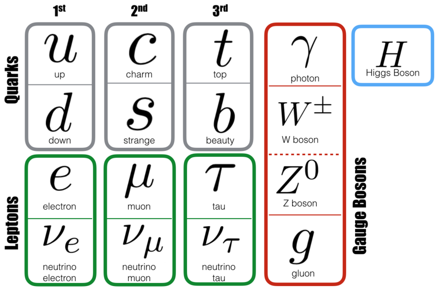
Outline

1. QCD in a nutshell
   - The Standard Model & strong interactions
   - Parton Distribution Functions
   - Factorization theorem
2. Machine Learning
   - Motivation for Machine Learning
   - Neural Networks & general strategy
3. The N3PDF project
   - The NNPDF methodology
   - Operator implementation in TensorFlow
   - Results & Conclusions

Quantum Chromodynamics in a nutshell

QCD in a nutshell

The Standard Model



Quantum Field Theory describing physics at the TeV scale

1. Fermions composing matter
2. Bosons mediating interactions
3. Scalar Higgs generating mass

QCD in a nutshell

Explore the strong interactions

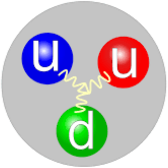
How to explore proton's inner structure?

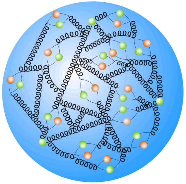
- Point-like projectile on the object $\longrightarrow$ DIS
- Smash the two objects $\longrightarrow$ LHC physics

"A way to analyze high energy collisions is to consider any hadron as a composition of point-like constituents $\longrightarrow$ **partons"** R.Feynman, 1969

QCD in a nutshell

Parton Distribution Functions





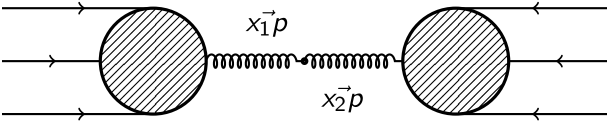
- Hadrons made of partonic objects $\longrightarrow$ non perturbative physics
- Interactions take place only at partonic level

Parton Distribution Functions: probability distribution of finding a particular parton (u, d, ..., g) carrying a fraction x of the proton's momentum

QCD in a nutshell

Factorization theorem

Observables in hadronic events $\longrightarrow$ σ is hard to compute



Factorize the problem $\longrightarrow$ Convolute the PDFs with the partonic $\hat{\sigma}_{ij}$

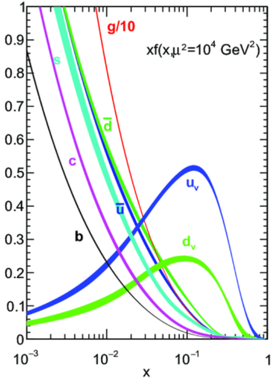
$$\sigma = \int_0^1 dx_1\, dx_2\, f_\alpha(x_1, \mu_F) * f_\beta(x_2, \mu_F) * \hat{\sigma}_{\alpha\beta}(\alpha_s(\mu_R), \mu_F)$$

- Partonic $\hat{\sigma}$ can be computed as perturbative series in α_s
- PDFs absorb the non perturbative effects, evaluated at μ_F

QCD in a nutshell

What PDFs look like

- Each parton has a different PDF $u(x), d(x), ..., g(x)$
- PDFs are not predicted, and can not be measured
- PDFs are extracted from data



Machine Learning

Machine learning

A hot topic at IFAE

TITLE: Searches for new phenomena at the LHC using machine-learning techniques

TITLE: Design and performance study of machine learning techniques for the ATLAS experiment event selection at the High-Luminosity LHC

TITLE: Enhanced ATLAS Level-1 trigger capabilities with Artificial-Intelligence regression on Field-Programmable Gate Array architecture.

TITLE: Unraveling New Physics effects in rare B decays using Neural Networks

Machine learning

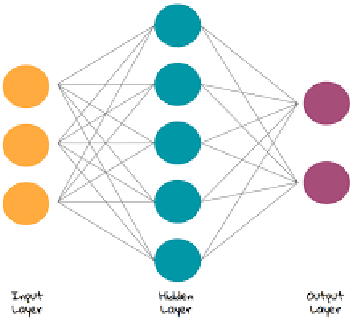
What is Machine Learning?

1. A subset of Artificial Intelligence (AI) algorithms
2. Used to solve *complex* tasks like classification and regression
3. Rely on comparison with data $\longrightarrow$ Learning



Machine learning

Building a Ml model

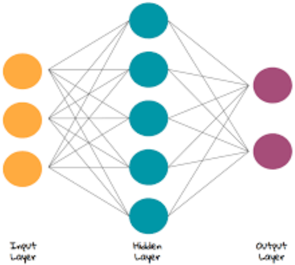


1. Turn the input set into an array and build a random prediction
2. Compare with truth and compute a Loss function
3. Update the parameters in a specific way

Machine learning

Building a ML model

$\longrightarrow$

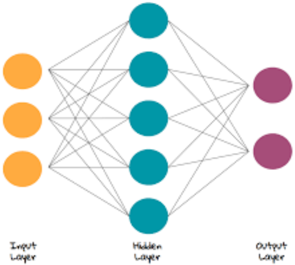


$\longrightarrow$ Tree ✓

Machine learning

Building a ML model

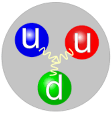
$\longrightarrow$



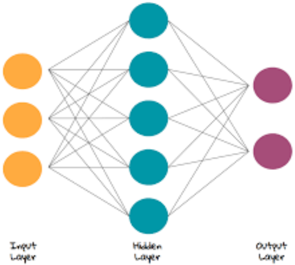
$\longrightarrow$ Tree ✓

Machine learning

Building a ML model



$\longrightarrow$

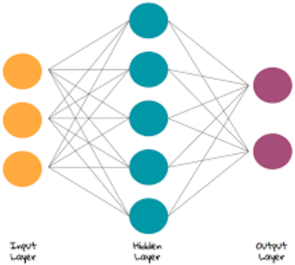


$\longrightarrow$ Not tree X

Machine learning

Building a ML model

$\longrightarrow$



$\longrightarrow$ Tree ✓

Loss function

$$Loss = \left\{\begin{matrix} y_1^{true} \\ y_2^{true} \\ \vdots \\ y_n^{true} \end{matrix}\right\} - \left\{\begin{matrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{matrix}\right\}$$

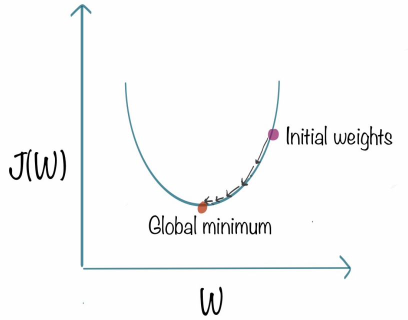
y_{true} $\qquad$ y_{pred}

1. Compute a loss function

$$L = \sum_{i=1}^{N} \left(y_i^{true} - y_i^{pred}\right)^2$$

2. Perfect prediction will mean $L = 0$

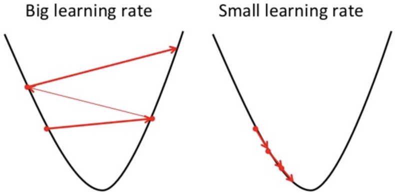
1. Loss is a function of weights and bias $L = L(w, b)$
2. Compute gradient $\nabla_{w_{ij}} L$ to look for the minimum of L



Update the parameters of the network following the gradient descend

$$w_{ij} \longrightarrow w_{ij} - \alpha \nabla L$$
$$b_i \longrightarrow b_i - \alpha \nabla L$$

Where α is the learning rate



The N3PDF project

Machine Learning for the precision determination of PDFs

Factorize the problem $\longrightarrow$ Convolute the PDFs with the partonic $\hat{\sigma}_{ij}$

$$\sigma = \int_0^1 dx_1 \; dx_2 \; f_\alpha(x_1, \mu_F) * f_\beta(x_2, \mu_F) * \hat{\sigma}_{\alpha\beta}(\alpha_s(\mu_R), \mu_F)$$

- Partonic $\hat{\sigma}_{\alpha\beta}$ is computed perturbatively. Hadronic σ is measured.
- Use a Neural Networks to generate (*fit*) the PDFs
- Generate a vector of observables σ_N to be compared with data

$$\sigma_N = \sum_{i,j,\alpha,\beta} f_\alpha(x_i) \; f_\beta(x_j) \; \hat{\sigma}_{Nij\alpha\beta}$$

The N3PDF project

General structure of n3fit

- Use TensorFlow and Keras to determine the PDFs
- See paper by S.Carraza - J.Cruz-Martinez
  "Towards a new generation of parton densities with deep learning models",
  https://arxiv.org/abs/1907.05075

The N3PDF project

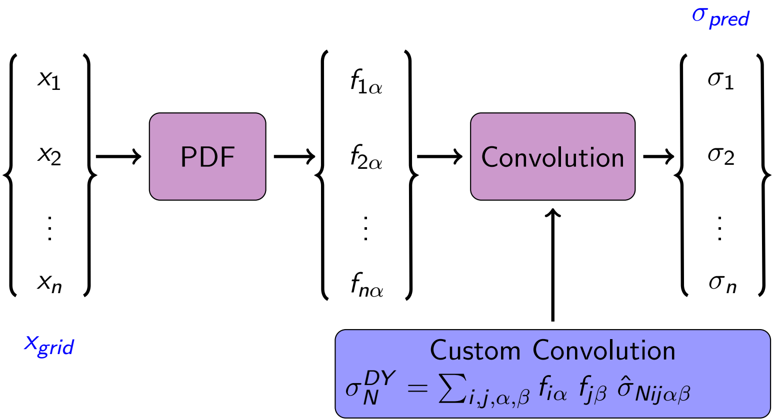
General structure of n3fit

$$\underset{x_{grid}}{\begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix}} \longrightarrow \boxed{\text{PDF}} \longrightarrow \begin{Bmatrix} f_{1\alpha} \\ f_{2\alpha} \\ \vdots \\ f_{n\alpha} \end{Bmatrix} \longrightarrow \boxed{\text{Convolution}} \longrightarrow \overset{\sigma_{pred}}{\begin{Bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_n \end{Bmatrix}}$$

1. Build a NN to compute σ_{pred} observables from a grid x_i
2. Compute χ^2 loss function by comparing with data
3. Update values of PDF $\longrightarrow$ Fit

The N3PDF project

Operator implementation



1. TF relies in symbolic computation $\longrightarrow$ High memory usage
2. Implement c++ operator replacing the convolution

Results
Checking computation

DIS:

	TensorFlow	Custom	Ratio
Convolution	1.9207904	1.9207904	1.0000000
	2.4611666	2.4611664	0.9999999
	1.3516952	1.3516952	1.0000000
Gradient	1.8794115	1.8794115	1.0000000
	1.505316	1.505316	1.0000000
	2.866085	2.866085	1.0000000

Hadronic:

	TensorFlow	Custom	Ratio
Convolution	8.142365	8.142366	1.0000001
	8.947762	8.947762	1.0000000
	7.4513326	7.4513316	0.9999999
Gradient	18.525095	18.525095	1.0000000
	19.182995	19.182993	0.9999999
	19.551006	19.551004	0.9999999

Hadronic only:

	TensorFlow	Custom Convolution	Diff
Virtual	17.7 GB	13.8 GB	3.9 GB
RES	12.1 GB	8.39 GB	3.2 GB

Global:

	TensorFlow	Custom Convolution	Diff
Virtual	23.5 GB	19.7 GB	3.8 GB
RES	18.4 GB	12.5 GB	5.9 GB

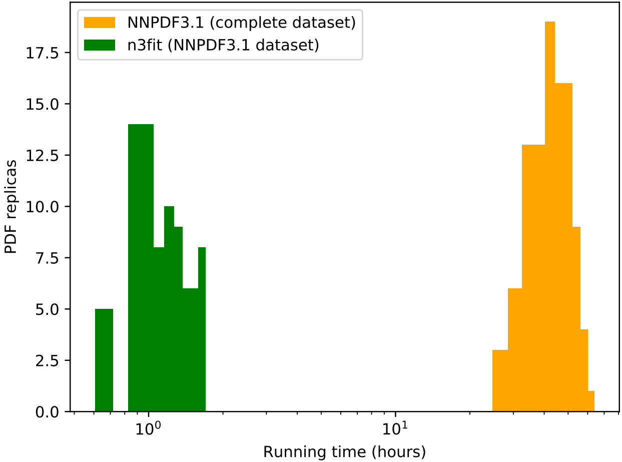
"Towards hardware acceleration for parton densities estimation", https://arxiv.org/abs/1909.10547

Summary & Conclusions

1. PDFs are required to have accurate predictions in high energy physics
2. ML provides a new way of determine the PDFs
3. Operator implementation leads to memory saving by taking full control on the computation

Thank you!

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 740006.


NNPDF3.1 (complete dataset)
n3fit (NNPDF3.1 dataset)
PDF replicas
Running time (hours)
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
10^0
10^1

Back up

TF computation of the gradient

1. Computing gradient of the χ^2 loss with respect to all parameters of the network.

$$\nabla\chi^2 \longrightarrow \frac{\partial\chi^2}{\partial x_i}$$

2. TF requires the gradient with respect to any operation in the model.

$$\frac{\partial\chi^2}{\partial x_i} = \frac{\partial\chi^2}{\partial\mathbf{Op}}\frac{\partial\mathbf{Op}}{\partial f_{\mu\nu}}\cdots\frac{\partial f_{\mu\nu}}{\partial x_i}$$

3. TF does not know the structure of **Op**. Compute manually the gradient $\frac{\partial\mathbf{Op}}{\partial f_{\mu\nu}}$ and implement it in the TF framework.

Back up

Manual computation of the gradient

1. From the expression of the output:

$$\mathbf{Op} \equiv y_N = \sum_{i,j,\alpha,\beta} f_\alpha(x_i) f_\beta(x_j)\, \hat{\sigma}_{Nij\alpha\beta}$$

2. Gradient of the output with respect ot the PDFs:

$$\begin{aligned}
\frac{\partial y_N}{\partial f_{\mu\nu}} &= \frac{\partial}{\partial f_{\mu\nu}} \sum_{i,j,\alpha,\beta} f_{i\alpha} f_{j\beta}\, \hat{\sigma}_{Nij\alpha\beta} \\
&= \sum_{i,j,\alpha,\beta} \left(\delta_{\mu i}\delta_{\nu\alpha} f_{\beta j} + \delta_{\mu j}\delta_{\nu\beta} f_{\alpha i}\right) \hat{\sigma}_{Nij\alpha\beta} \\
&= \sum_{j,\beta} f_{j\beta}\, \hat{\sigma}_{N\mu j\nu\beta} + \sum_{i,\alpha} f_{i\alpha}\, \hat{\sigma}_{Ni\mu\alpha\nu}
\end{aligned}$$