Application of Machine Learning to Global Fits of PDFs

CHRIS COCUZZA

(TEMPLE UNIVERSITY)

Introduction

In our work (W. Melnitchouk, N. Sato, A. Metz), we use a Monte Carlo simulation to fit parton distribution functions (PDFs) to data from experiment.

1. The PDFs are parameterized as $Nx^a(1-x)^b$.
2. Data points are varied randomly within their uncertainties at the start.
3. Difference of the data and theoretical results from the PDFs is calculated.
4. The parameters of the PDFs are then varied randomly and the difference between data and theory is recalculated.
5. If the difference decreases, the new parameters are kept. This process repeats until the difference no longer decreases.

Complications

The observable measured by the data is not actually the PDFs, it is for structure functions (SFs) which are found through measurement of the cross section. PDFs and SFs are related through a convolution.

Things are further complicated by the fact that some of the data is for the deuteron (one proton + one neutron) and not just a single hadron.

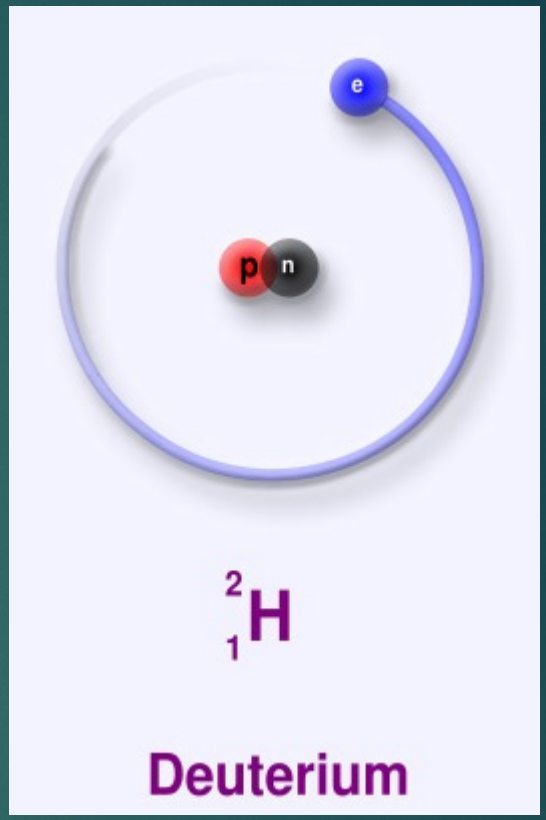


Image from https://energyeducation.ca/encyclopedia/Deuterium

Experimental Measurement —allows calculation→ Cross Section —related to→ SF —convolution→ PDF

The Deuteron

Due to nuclear smearing effects (the wavefunctions of the proton and neutron overlapping), the deuteron structure function is not just the addition of the proton and neutron structure functions. Instead it is a convolution of the hadronic structure functions with a "smearing function:"

$$F_2^D = \sum_{N=p,d} \int dy * f_{22}^N(y) * F_2^N\left(\frac{x}{y}, Q^2\right)$$

Here F_2 denotes a structure function, while f_{22} denotes a smearing function.

The convolution is handled by expressing F_2^N through its Mellin moments and then integrating over a contour in the complex plane.

The Smearing Function

This Mellin moments technique will lead to factors like this:

$$\int_x^{y_{max}} dy\, f_{22}^N(y) \left(\frac{x}{y}\right)^{-N}$$

where N is the contour in the complex plane.

Fortunately, these factors do not depend on the fitting parameters (they do not involve the SFs and thus do not involve the PDFs). This allows us to calculate and save these factors before the fits begin. Then the fit can load them and pull the values as needed.

Table Generation

In practice, sufficient accuracy is reached if the integral is evaluated at 68 points along the contour N.
This must be done for every (x, Q^2) deuteron data point to be used during the fit (up to 1,000+ data points depending on the fit). If we want to make a plot, we must also generate the necessary data points for that plot.
These tables in total end up to be about ~10 MB.

So if we want to submit 1,000 runs, that is:

$$10\ MB\ * 1{,}000 = 10\ GB\ of\ memory$$

Not too bad! The real issue arises when we need to do a ***double*** Mellin transform.

The Dreaded Double Mellin

The double Mellin transform arises when one wants to take into account the fact that the nucleons inside of the deuteron may not be on shell, i.e:

$$p^2 \neq M_n^2$$

It also arises in other contexts. One will then run into factors like this:

$$\int_x^{y_{max}} dy\, f_{22}^N(y) \left(\frac{x}{y}\right)^{-M-N} \qquad \int_x^{y_{max}} dy\, f_{22}^N(y) \left(\frac{x}{y}\right)^{-M^*-N}$$

where M is another contour in the complex plane and M* is its complex conjugate.

Let's see how much memory we need now…

Table Generation for Double Mellin

We must now calculate the 68 points along the M contour for every value of N, and the same for the M* contour.

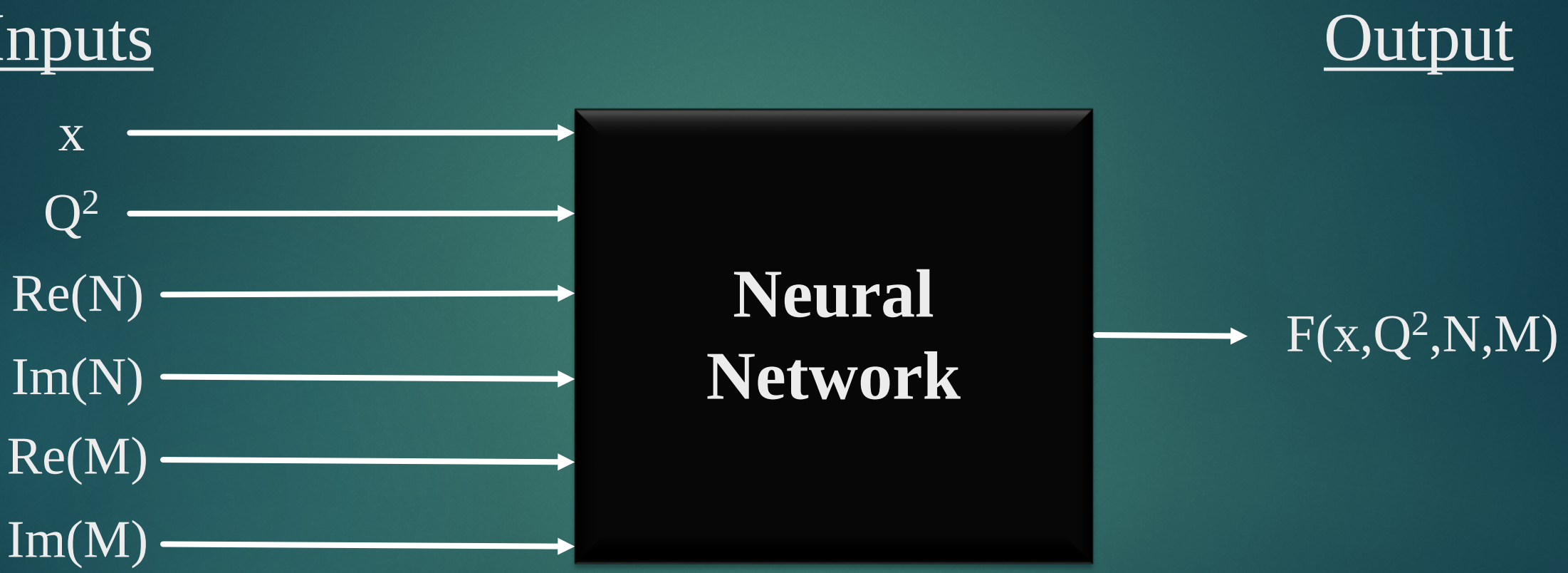
So if we want to submit 1,000 runs, that is:

$$10\ MB * 68 * 2 * 1{,}000 = 1360\ GB\ of\ memory$$

- Memory for single Mellin transform
- Points along M contour
- Must calculate for both M and M*
- Number of runs

That's a lot worse! This means that memory becomes the main limitation of how many fits we can run simultaneously.

Machine Learning to the Rescue!



The inputs are used to train the neural network. The output is a function that takes the inputs and can predict the output if the model is trained well.

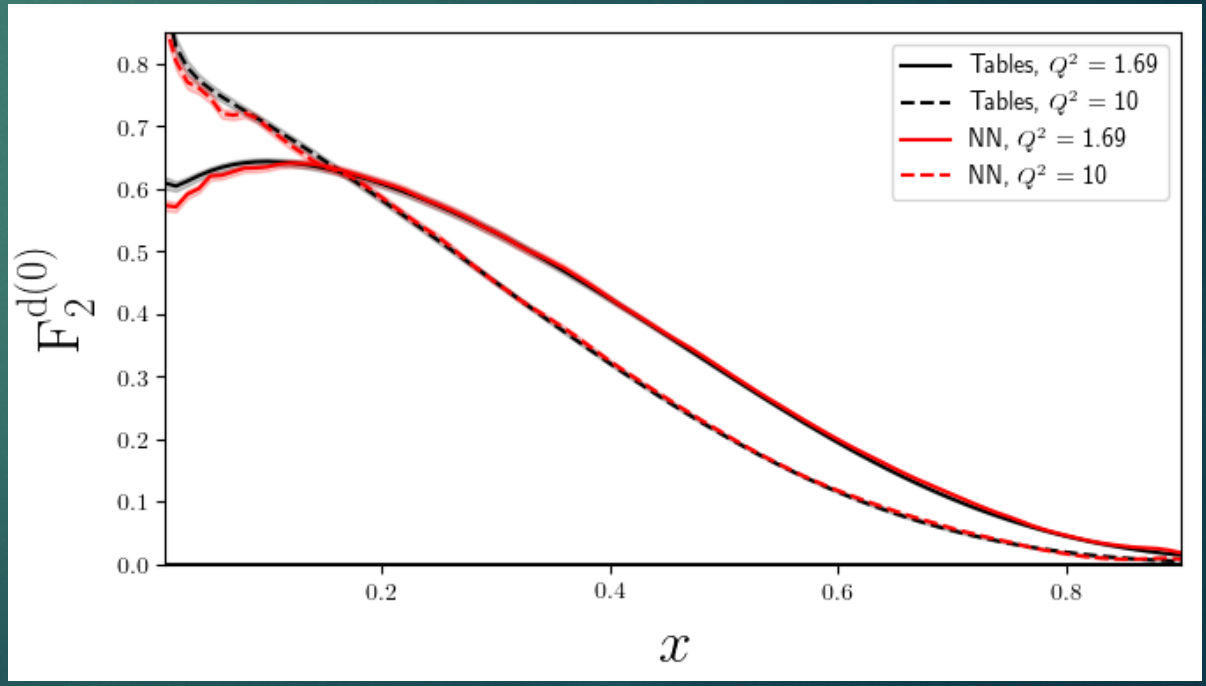
Advantages/Potential Issues of ML

Advantages

- The resulting function from the neural network is very small; less than 1 MB. Thus, for 1,000 runs, the amount of memory required is less than 1 GB. So we went from 1,360 GB to 1 GB using neural networks!!
- If we want to make a new plot or include new data into our fit, we do not need to generate new tables (which take up time and memory)

Issues

- Training the models is quick, but nontrivial. We still need to figure out how to do it well.



THE END

Thank you to Nobuo Sato, Patrick Barry, Yiyu Zhou, and Yaohang Li for their help in learning neural networks

Supported by the NSF