

Extraction of Drell-Yan Angular Coefficients using Neural Network-based Classifiers

Dinupa Nawarathne

New Mexico State University
Representing the E906/SeaQuest Collaboration
FERMILAB-SLIDES-23-124-PPD

New Perspectives Meeting - June 27, 2023



1 Methodology

Likelihood Ratio Test
Gaussian Example

2 Drell-Yan Angular Coefficients

Drell-Yan Cross Section
FermiLab E906/SeaQuest Experiment
Applying the Fitting Algorithm to E906 MC Data

3 Summary

Likelihood Ratio Test


- The likelihood ratio test is a highly effective method for assessing the goodness of fit.
- Let $X_1, X_2, X_3, \dots, X_n$ be a random sample from a distribution with a parameter θ . Suppose that we have observed $X_1 = x_1, X_2 = x_2, \dots, X_n = x_n$. We define the the likelihood function as the joint probability of the observed samples as a function of θ ;

$$L(x_1, x_2, \dots, x_n; \theta) = P(X_1 = x_1, X_2 = x_2, \dots, X_n = x_n; \theta)$$

- To decide between two simple hypotheses $H_0 : \theta = \theta_0$ and $H_1 : \theta = \theta_1$, we define the likelihood ratio:

$$\lambda(x_1, x_2, \dots, x_n) = \frac{L(x_1, x_2, \dots, x_n; \theta_0)}{L(x_1, x_2, \dots, x_n; \theta_1)}$$

- To perform a likelihood ratio test, we choose a constant c . We reject H_0 if $\lambda < c$ and accept it if $\lambda \geq c$. The value of c can be chosen based on the desired significance level α .
- Neural networks excel at approximating non-linear functions, making them ideal for use as higher-dimensional likelihood functions.
- Our goal is to train the neural network to classify samples accurately. Specifically, we aim to classify samples $\omega_{0i} \in \Omega_0$ as $y = 0$ and $\omega_{1i} \in \Omega_1$ as $y = 1$, regardless of the parameter θ .
- Subsequently, we can utilize the trained neural network to estimate any unknown parameter θ_{unknown} by employing the gradient descent algorithm.¹

¹A. Andreassen, B. Nachman, *Phys. Rev. D* **101**, 091901, arXiv: 1907.08209 (hep-ph) (2020). 

- Suppose we have a Gaussian distribution $\mathcal{N}(\mu_{\text{unknown}}, 1)$, where μ_{unknown} is the unknown parameter.
- Our training strategy is to train the neural network to classify samples drawn from two Gaussian distributions: $x_0 \sim \mathcal{N}_0(0, 1)$ with the label $y = 0$, and $x_1 \sim \mathcal{N}_1(\mu, 1)$ with the label $y = 1$, where μ can take any value within a specified range.
- We construct the neural network with three hidden layers, each layer consisting of 50 nodes. The layers are activated using the Rectified Linear Unit (ReLU), while the final layer is activated by the Sigmoid function.
- The neural network was trained for 200 epochs, employing early stopping with a patience of 10, to minimize the binary cross-entropy loss.

Gaussian Example

Extraction of
Drell-Yan
Angular
Coefficients
using Neural
Network-based
Classifiers

Dinupa
Nawarathne

Methodology

Likelihood Ratio
Test

Gaussian Example

Drell-Yan
Angular
Coefficients

Drell-Yan Cross
Section

FermiLab
E906/SeaQuest
Experiment

Applying the
Fitting Algorithm
to E906 MC Data

Summary

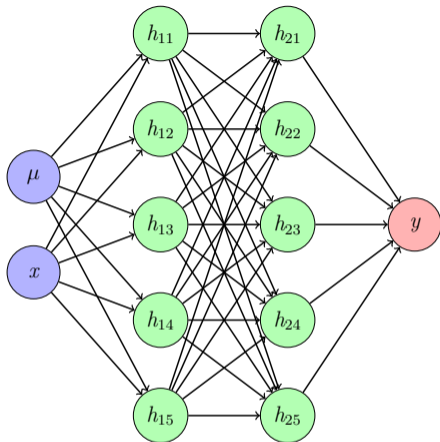


Figure 1: Example of a neural network architecture: The input layer is depicted with blue nodes, the hidden layers are represented by green nodes, and the output layer is indicated by a red node.

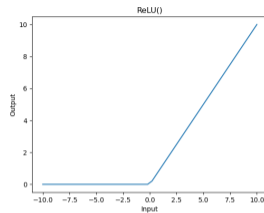


Figure 2: ReLU activation function.

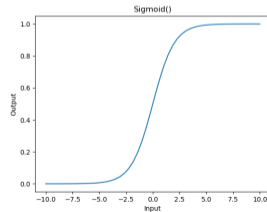


Figure 3: Sigmoid activation function.

Gaussian Example

- To perform the fitting algorithm, we fix the weights and biases of the trained neural network. Then, we optimize the μ parameter by using the gradient descent algorithm to find the optimal value μ_{opt} .

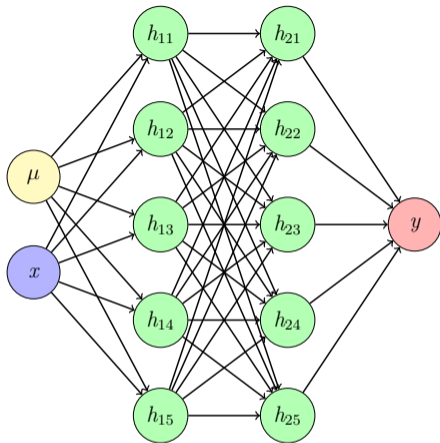


Figure 4: In the fitting step, the only trainable parameter is the yellow node representing μ .

Gaussian Example

Extraction of
Drell-Yan
Angular
Coefficients
using Neural
Network-based
Classifiers

Dinupa
Nawarathne

Methodology

Likelihood Ratio
Test

Gaussian Example

Drell-Yan
Angular
Coefficients

Drell-Yan Cross
Section

FermiLab
E906/SeaQuest
Experiment

Applying the
Fitting Algorithm
to E906 MC Data

Summary

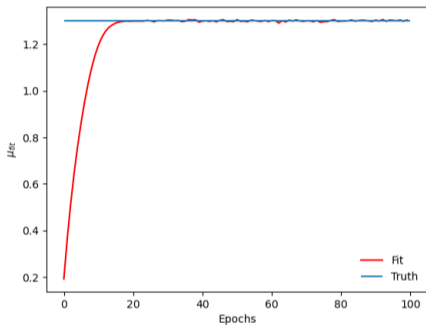


Figure 5: We tested the algorithm with $\mu_{\text{unknown}} = 1.3$. As depicted in the plot, the optimal value for μ converges to the correct value as the epochs progress.

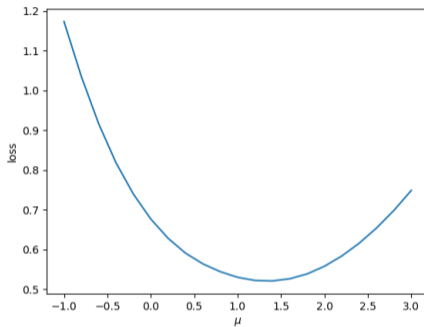
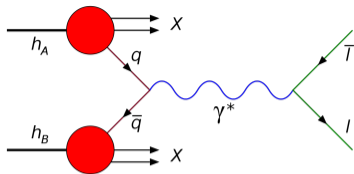


Figure 6: The loss reaches its minimum value at the optimal value of μ .

Drell-Yan Cross Section



$$\frac{d\sigma}{d\Omega} \propto 1 + \lambda \cos^2\theta + \mu \sin 2\theta \cos\phi + \frac{\nu}{2} \sin^2\theta \cos 2\phi$$

Figure 7: Drell-Yan process.

- In the “naive” Drell-Yan model, where we ignore the transverse momentum of the quark and assume no gluon emission, we have $\lambda = 1$, $\mu = \nu = 0$.
- Measuring the $\cos 2\phi$ angular dependence can be used to extract the Boer-Mulders (BM) function.
- BM function describes the transverse-polarization asymmetry of quarks within an unpolarized hadron and results from the coupling between the transverse momentum and transverse spin of the quarks inside the hadron.

FermiLab E906/SeaQuest Experiment

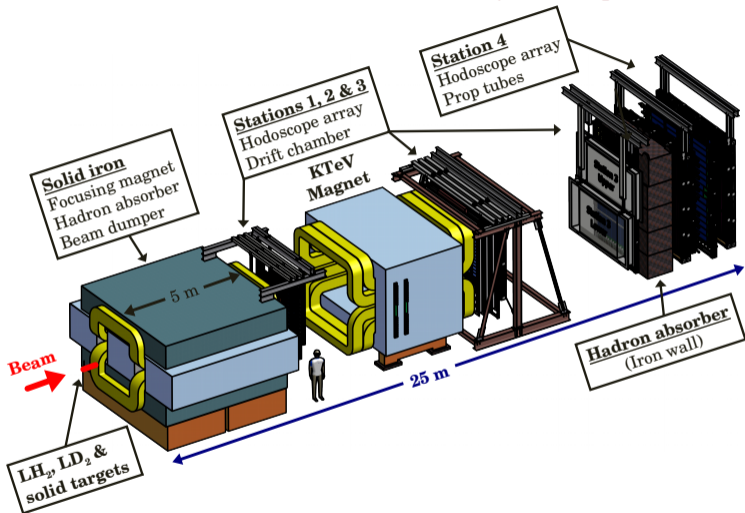


Figure 8: FermiLab E906/SeaQuest spectrometer.

Extraction of
Drell-Yan
Angular
Coefficients
using Neural
Network-based
Classifiers

Dinupa
Nawarathne

Methodology

Likelihood Ratio
Test
Gaussian Example

Drell-Yan
Angular
Coefficients

Drell-Yan Cross
Section

FermiLab
E906/SeaQuest
Experiment

Applying the
Fitting Algorithm
to E906 MC Data

Summary

Applying the Fitting Algorithm to E906 MC Data

- The neural network was trained for 200 epochs, employing early stopping with a patience of 20, to minimize the binary cross-entropy loss.
- We conducted a sanity check using four different combinations of λ , μ , and ν . The fitted values are presented in table 1.
- In the four test samples, we were able to extract the injected parameters within a ± 1.5 standard deviation (σ) interval. The number of events included in one test sample is close to the number of events in the real data.

Applying the Fitting Algorithm to E906 MC Data

Combination	Coefficient	Injected	Fitted
1	λ	0.84	0.876 ± 0.208
	μ	0.26	0.234 ± 0.054
	ν	-0.34	-0.299 ± 0.052
2	λ	1.33	1.134 ± 0.151
	μ	0.17	0.146 ± 0.050
	ν	-0.34	-0.281 ± 0.043
3	λ	1.12	1.242 ± 0.181
	μ	-0.27	-0.211 ± 0.088
	ν	-0.24	-0.236 ± 0.071
4	λ	0.62	0.888 ± 0.282
	μ	-0.32	-0.232 ± 0.091
	ν	0.18	0.147 ± 0.055

Table 1: Table showing the fitted values of λ , μ , and ν using the gradient descent algorithm.

- Neural networks can be used as multi-dimensional likelihood functions, and we can utilize likelihood ratio test to extract the optimal parameters for the Drell-Yan angular distribution.
- Measuring the $\cos 2\phi$ angular dependence with higher accuracy is important for extracting the Boer-Mulders (BM) functions.
- BM function describes the transverse-polarization asymmetry of quarks within an unpolarized hadron and results from the coupling between the transverse momentum and transverse spin of the quarks inside the hadron. This function serves as a useful tool for unraveling the structure of hadrons.
- Our plan is to use this high-dimensional fitting algorithm to extract the Drell-Yan angular coefficients from the E906/SeaQuest data with higher accuracy.
- Acknowledgement: This work was funded by the DOE Office of Science, Medium-Energy Nuclear Physics Program.

2D Example

Extraction of
Drell-Yan
Angular
Coefficients
using Neural
Network-based
Classifiers

Dinupa
Nawarathne

Methodology

Likelihood Ratio
Test

Gaussian Example

Drell-Yan
Angular
Coefficients

Drell-Yan Cross
Section

FermiLab
E906/SeaQuest
Experiment

Applying the
Fitting Algorithm
to E906 MC Data

Summary

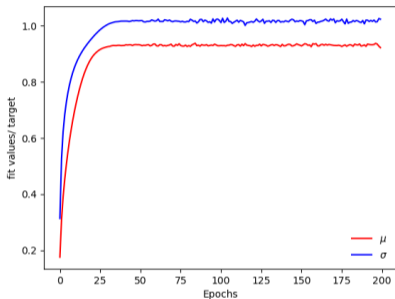


Figure 9: We tested the algorithm with $\mu_{\text{unknown}} = 1.2$ and $\sigma_{\text{unknown}} = 0.7$. As depicted in the plot, the optimal value for μ and ν converges to the correct value as the epochs progress.

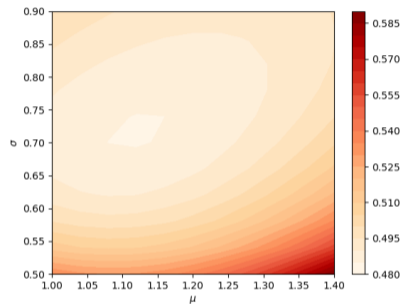


Figure 10: The loss reaches its minimum value at the optimal value of μ and ν .

- Think of a neural network as a network of interconnected nodes, called neurons, organized in layers. The input layer receives information, such as numerical data, images, or text. This information is then passed through the network, layer by layer, where each neuron performs a simple calculation based on its input and a set of weights.
- During training, the neural network adjusts these weights based on the provided input and the desired output. It learns by continuously comparing its predictions to the actual results and updating the weights accordingly, aiming to minimize the difference between them. This process, known as backpropagation, allows the network to learn and improve its performance over time.

- Once trained, the neural network can make predictions or classify new, unseen data based on the patterns it has learned. It can handle a wide range of tasks, including image and speech recognition, natural language processing, and predictive modeling.
- Learning Rate: The learning rate is a hyperparameter that determines how quickly a neural network adapts its weights during training. It controls the step size taken in the direction of minimizing the loss function. A higher learning rate can lead to faster convergence, but it may also cause overshooting and instability. On the other hand, a lower learning rate can ensure more stable updates, but it may slow down the convergence. Finding an appropriate learning rate is crucial for effective training.

- **Early Patience (Early Stopping):** Early patience, also known as early stopping, is a technique used to prevent overfitting and improve generalization in a neural network. During training, the model's performance is monitored on a validation set. If the validation loss stops improving or starts to worsen after a certain number of epochs, early patience is applied. It involves stopping the training process early and using the model with the best validation performance as the final model. Early patience helps to prevent the model from overfitting to the training data and ensures it generalizes well to unseen data.

- The binary cross entropy loss is calculated as follows;

$$\mathcal{L}(y, \hat{y}) = -(y * \log(\hat{y}) + (1 - y) * \log(1 - \hat{y}))$$

where y represents the true binary label (either 0 or 1) of the training example. \hat{y} represents the predicted probability of the positive class (typically obtained by passing the input through a sigmoid activation function).