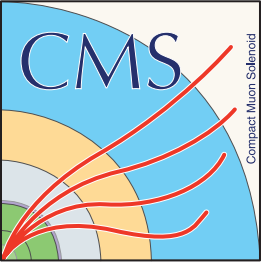


Event Classifier Transformer

Jaebak Kim (UCSB)

Hobby in Artificial Intelligence

- First introduced to AI by my PhD supervisor→ (E. Won).
- Feed-forward neural network (FFN) with one hidden layer.
  - $g_i = b'_i + \sum_j w'_{ij} \times \tanh\left(b_j + \sum_k w_{jk} \times x_k\right)$
- FFN can describe wide range of functions f(x).
  - $y = f(x) = 1 + \frac{x^2}{4000} - \cos(x)$
  - FNN with 4 nodes (13 parameters):

$$g(x) = b' + w'_1 \times \tanh(w_1 \times x + b_1) + w'_2 \times \tanh(w_2 \times x + b_2) + w'_3 \times \tanh(w_3 \times x + b_3) + w'_4 \times \tanh(w_4 \times x + b_4)$$

- Tried applying FFN to trigger for Belle II, but it had performance limitations.

Available online at www.sciencedirect.com
ScienceDirect
Nuclear Instruments and Methods in Physics Research A 581 (2007) 816–820
NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A
www.elsevier.com/locate/nima

A hardware implementation of artificial neural networks using field programmable gate arrays

2007 paper

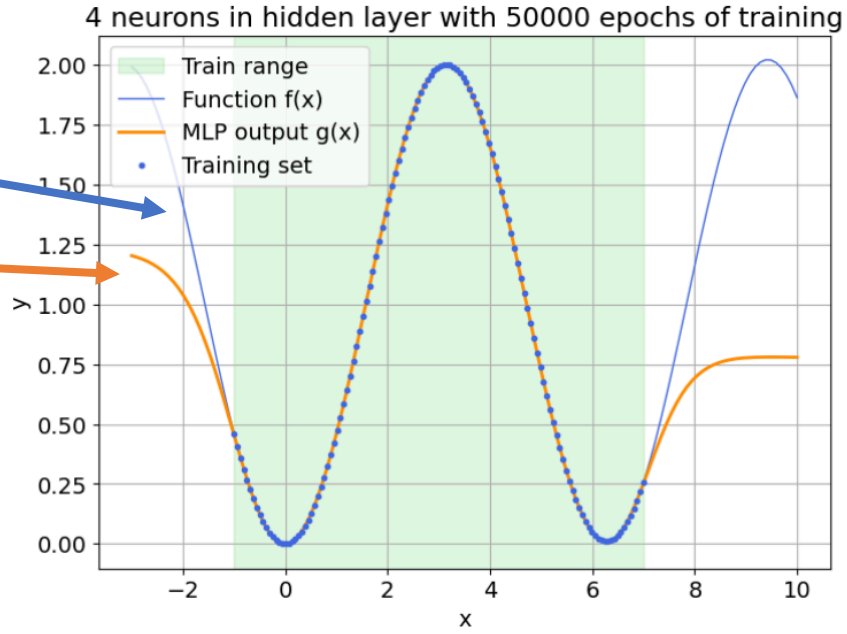
E. Won*
Department of Physics, Korea University, Seoul 136-713, Republic of Korea

Single author!

Received 5 March 2007; received in revised form 18 July 2007; accepted 9 August 2007
Available online 19 August 2007

Abstract

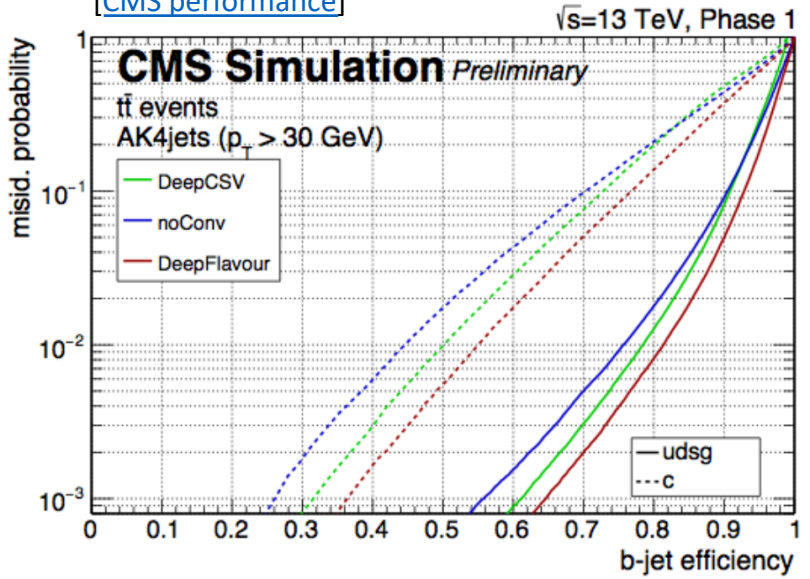
An artificial neural network algorithm is implemented using a low-cost field programmable gate array hardware. One hidden layer is used in the feed-forward neural network structure in order to discriminate one class of patterns from the other class in real time. In this work, the training of the network is performed in the off-line computing environment and the results of the training are configured to the hardware in order to minimize the latency of the neural computation. With five 8-bit input patterns, six hidden nodes, and one 8-bit output, the implemented hardware neural network makes decisions on a set of input patterns in 11 clock cycles, or less than 200 ns with a 60 MHz clock. The result from the hardware neural computation is well predictable based on the off-line computation. This implementation may be used in level 1 hardware triggers in high energy physics experiments.



Hobby in Artificial Intelligence (AI)

- In 2016, AlphaGo AI defeated well known professional player. So I was curious and studied it.
  - AI based UCT (Upper Confidence bound applied to Trees) algorithm that utilized CNN (Convolutional Neural Networks).
  - Learned about CNN, RNN (Recursive NN), LSTM (long short term memory) networks.
  - Interesting but wasn't easily applicable for a "Search for CP violation in $D^0 \to K^+K^-\pi^+\pi^-$ at Belle" analysis.
- In 2020, utilized "DeepCSV" to tag b-jets in the $H(b\bar{b})H(b\bar{b}) + p_T^{\text{miss}}$ analysis at CMS.
  - DeepCSV: FFN with 4 hidden layers, each with 100 nodes and using RELU (rectified linear units).
  - Recently more CMS analyses starting to use DeepFlavor, which consists of CNN layers, LSTM layers, and FFN layers.

[CMS performance]

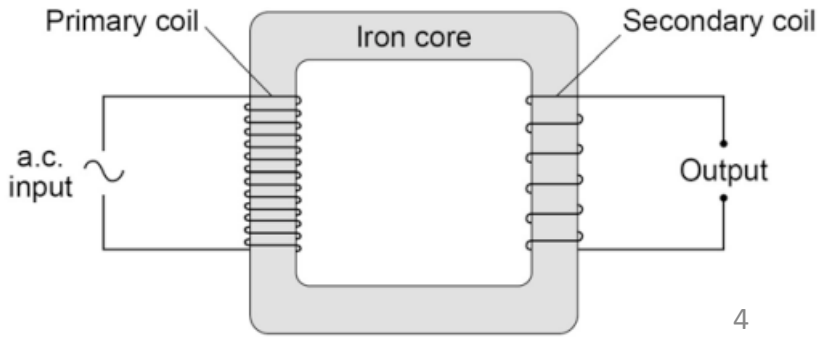


Hobby in Artificial Intelligence (AI)

- After playing tennis with a computer science PhD student at UCSB in summer of 2022, I talked about what research he was doing.
- He said he was researching "transformers" in the natural language processing field.
  - ➢He came to UCSB, quitting Naver.
  - ➢He did internships at Apple, Microsoft, ...
  - ➢He said transformers were not an RNN.
- I got curious in what transformers were...

UCSB is a nice place to play tennis.
Good weather.

My understanding of a "transformer" at the time...



Transformer neural network

- What is the transformer neural network (NN)?
  - Developed for translation of English to German (English to French) by a Google team in 2017.
  - "Randomly" combined many concepts in the natural language field according the intern on the team.
    - Encoder/decoder structure
    - Attention
    - Residual connections
    - Layer normalization
    - Warmup training
    - Label smoothing
    - …
  - Encoder decoder structure.
    - Encoder: Transforms English to abstract language space.
    - Decoder: Transforms abstract language space to German.

Google Paper (2017): [Attention Is All You Need]

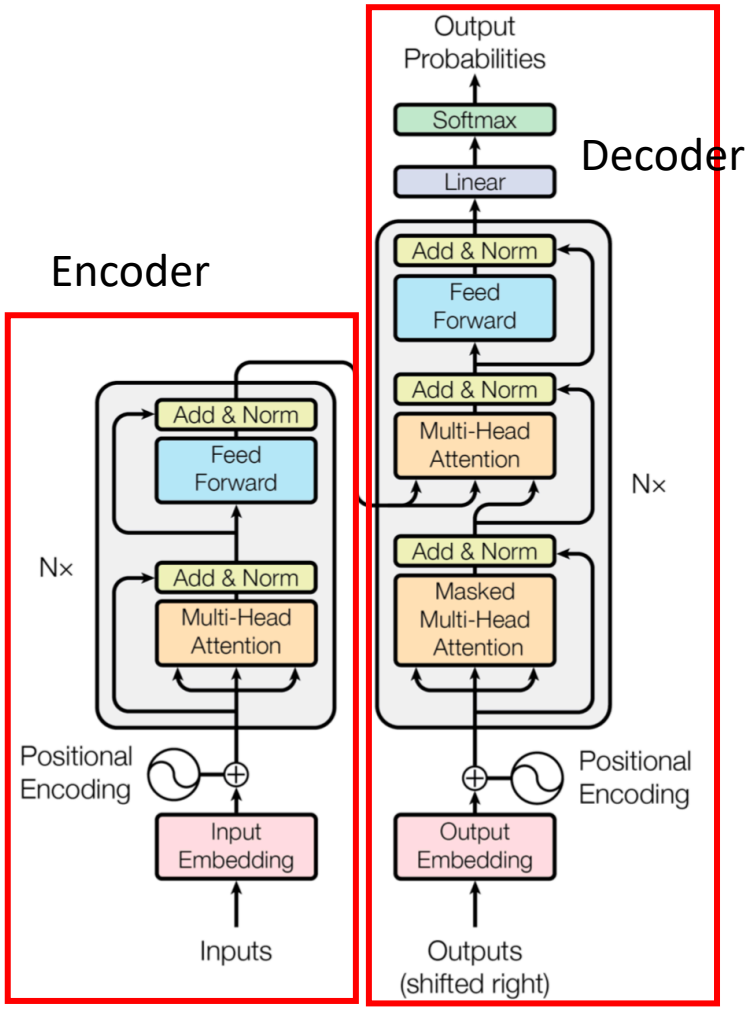


Figure 1: The Transformer - model architecture.

Transformer neural network

Google Paper (2017): [Attention Is All You Need]

- Why is the transformer good?
  - For some unknown reason, when increasing number of transformer NN weights, performance also increases (No limit??)
  - Normal NNs have a limit of increasing number of weights, because NN has a hard time finding the optimum in training.
  - However requires a large data sample for training.

Table 3: Variations on the Transformer architecture. Unlisted values are identical to those of the base model. All metrics are on the English-to-German translation development set, newstest2013. Listed perplexities are per-wordpiece, according to our byte-pair encoding, and should not be compared to per-word perplexities.

Number of weights

	N	d_{model}	d_{ff}	h	d_k	d_v	P_{drop}	ϵ_{ls}	train steps	PPL (dev)	BLEU (dev)	params $\times 10^6$
base	6	512	2048	8	64	64	0.1	0.1	100K	4.92	25.8	65
(A)				1	512	512				5.29	24.9	
				4	128	128				5.00	25.5	
				16	32	32				4.91	25.8	
				32	16	16				5.01	25.4	
(B)					16					5.16	25.1	58
					32					5.01	25.4	60
(C)	2									6.11	23.7	36
	4									5.19	25.3	50
	8									4.88	25.5	80
		256			32	32				5.75	24.5	28
		1024			128	128				4.66	26.0	168
			1024							5.12	25.4	53
			4096							4.75	26.2	90
(D)							0.0			5.77	24.6	
							0.2			4.95	25.5	
								0.0		4.67	25.3	
								0.2		5.47	25.7	
(E)	positional embedding instead of sinusoids									4.92	25.7	
big	6	1024	4096	16			0.3		300K	**4.33**	**26.4**	213

Translation score

Transformer neural network

Google Paper (2017): Attention Is All You Need

- What are the input and output of the network?
  - Input: English sentence where each word(s) is converted to a N-dimensional vector.
    - Example: My name is Jaebak.
      - My: Converted to a N-dimensional vector (1.1, 0.2, 2.1, 3.4, ...)
  - Output: Prediction of "next" Korean word(s). Output sentence is made by an iterative processes.

Iteration	Pre-output	Output
1	"Start of sentence" (Start)	(Start) 제
2	(Start) 제	(Start) 제 이름은
3	(Start) 제 이름은	(Start) 제 이름은 김재박입니다.

  - Output gives probabilities of word(s), where a N-dimensional vector is converted to probability of word(s)
    - Example: Probability output for iteration 1.
      - 제: 50%
      - 내: 45%
      - ...: 5%

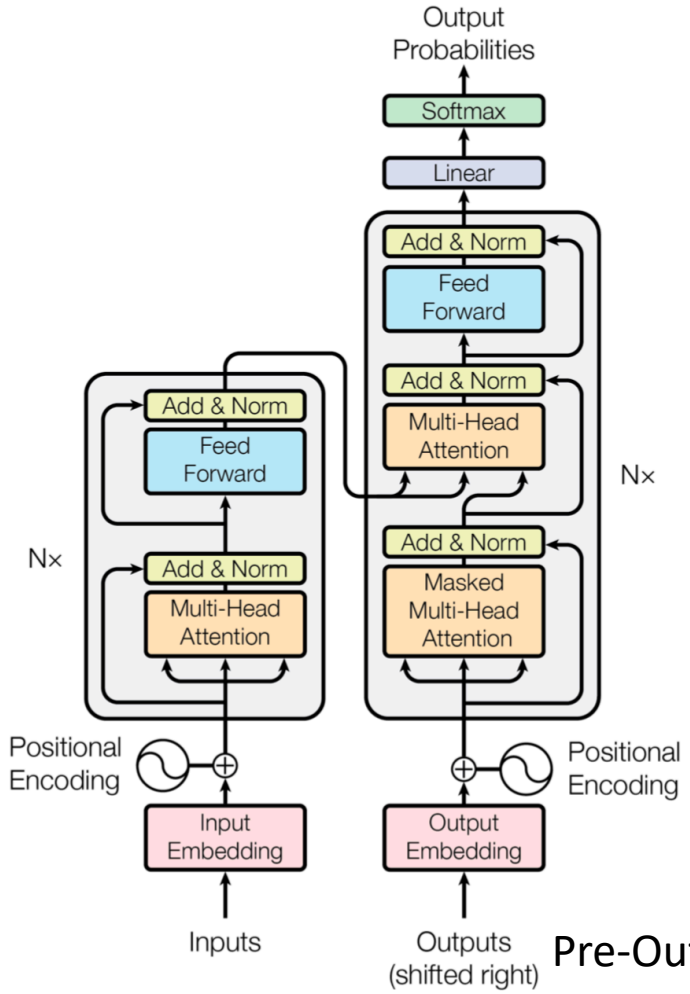
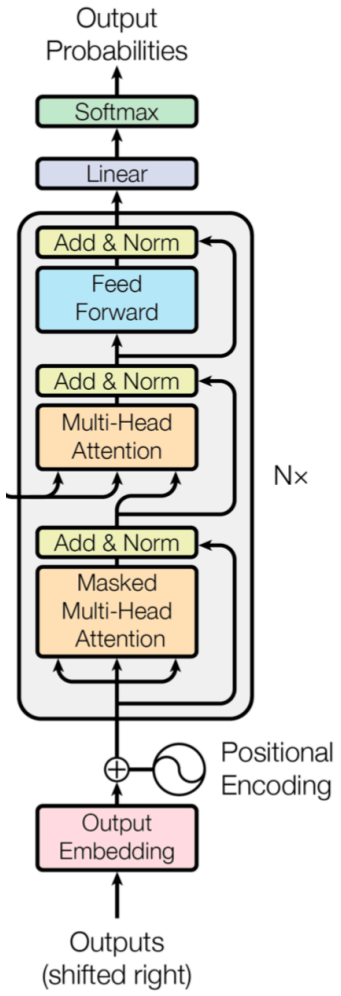


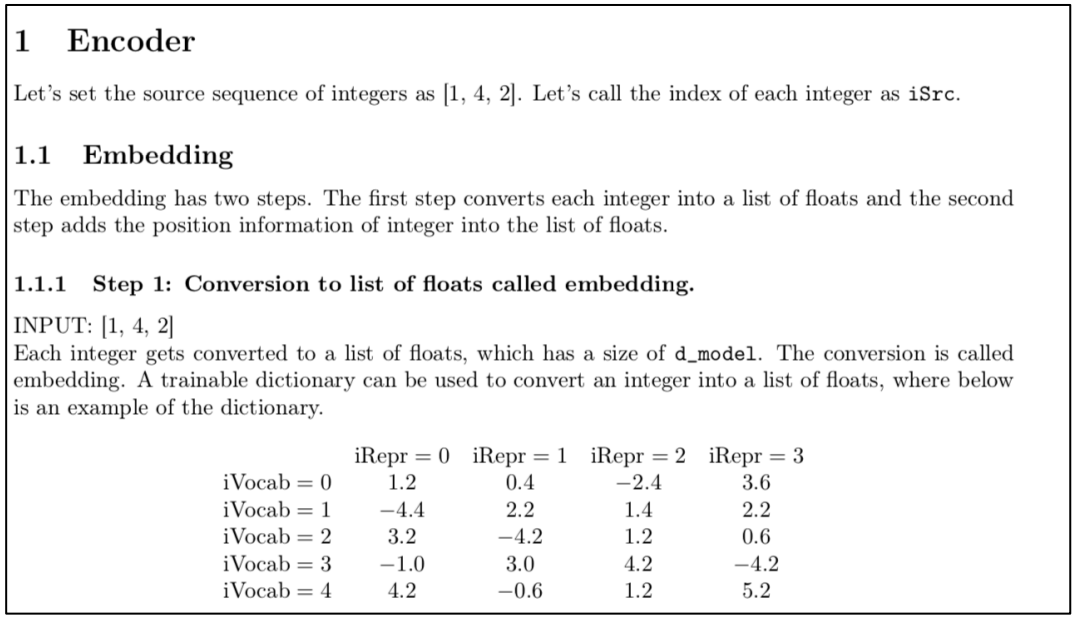
Figure 1: The Transformer - model architecture.

ChatGPT (Generative Pre-trained Transformer)

- Based on the transformer, OpenAI made ChatGPT.
- Just used the decoder part of the transformer.
- Input: Words of a sentence.
  - Example: One plus two is
- Output: Prediction of the next word
  - Example: three.
- Used a huge dataset from internet and huge amount of weights (GPT3 has 175×10^9 weights...)
  - If each weight is 64 bits, weight is 1.5 Terra-Byte of data...
  - When training the network GPUs are used.
  - A normal GPU has few GB of memory. High-end gaming GPUs have 16 GB of memory...
- Because more weights could mean more performance, there is a large investment in bigger GPUs. Nvidia...
  - Many GPUs can be chained together to train transformers.



How did I study the transformer?



1 Encoder

Let's set the source sequence of integers as [1, 4, 2]. Let's call the index of each integer as `iSrc`.

1.1 Embedding

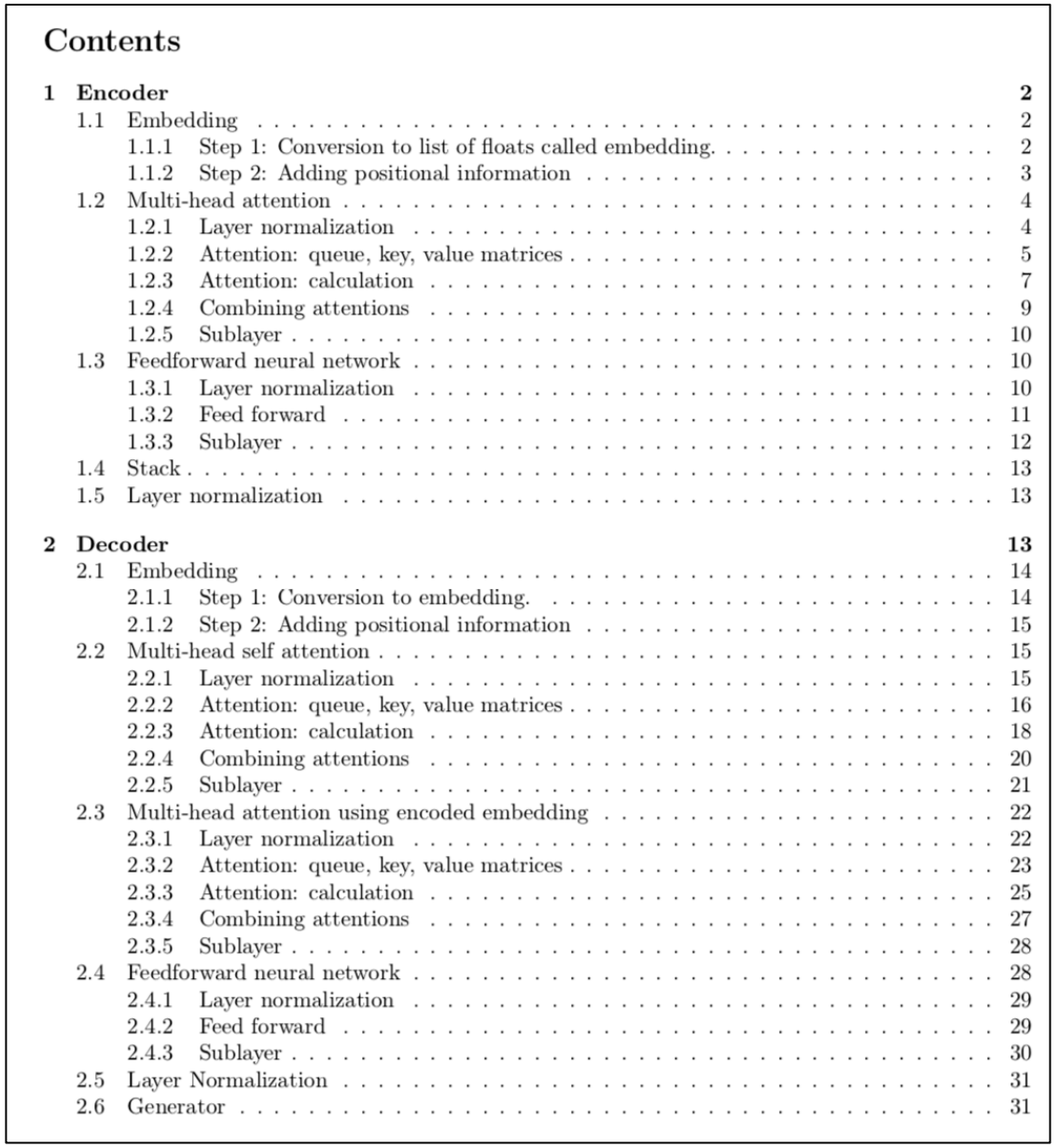
The embedding has two steps. The first step converts each integer into a list of floats and the second step adds the position information of integer into the list of floats.

1.1.1 Step 1: Conversion to list of floats called embedding.

INPUT: [1, 4, 2]
Each integer gets converted to a list of floats, which has a size of `d_model`. The conversion is called embedding. A trainable dictionary can be used to convert an integer into a list of floats, where below is an example of the dictionary.

	iRepr = 0	iRepr = 1	iRepr = 2	iRepr = 3
iVocab = 0	1.2	0.4	−2.4	3.6
iVocab = 1	−4.4	2.2	1.4	2.2
iVocab = 2	3.2	−4.2	1.2	0.6
iVocab = 3	−1.0	3.0	4.2	−4.2
iVocab = 4	4.2	−0.6	1.2	5.2

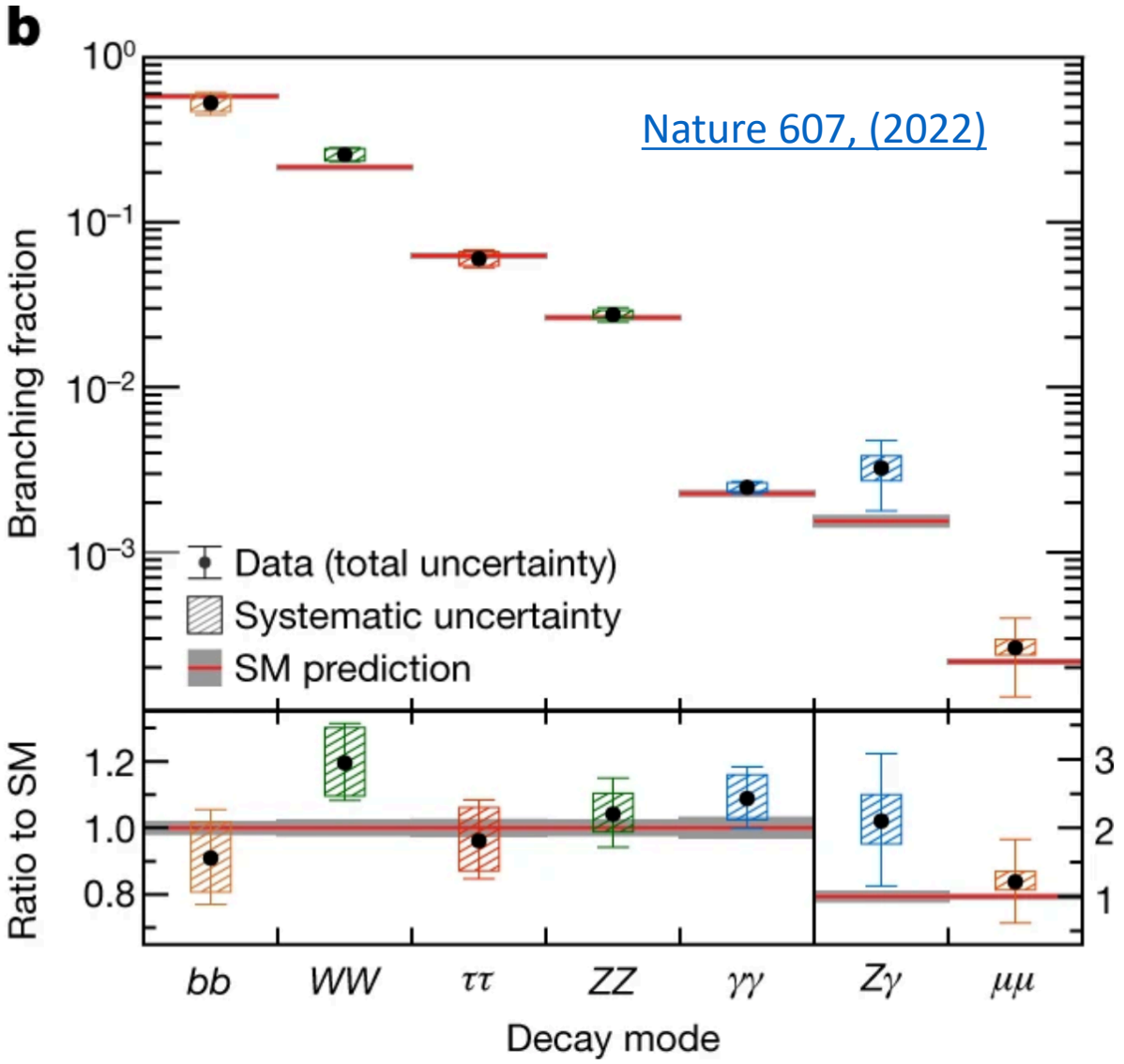
- Made a document that has an example of calculations for each step in network.
- Have a Excel sheet that has all calculations of transformer that were matched up with a transformer example.
- Gained better understanding and intuition of transformers.
- Took quite a bit of time because I was studying it during my free time (weekends...)



Contents

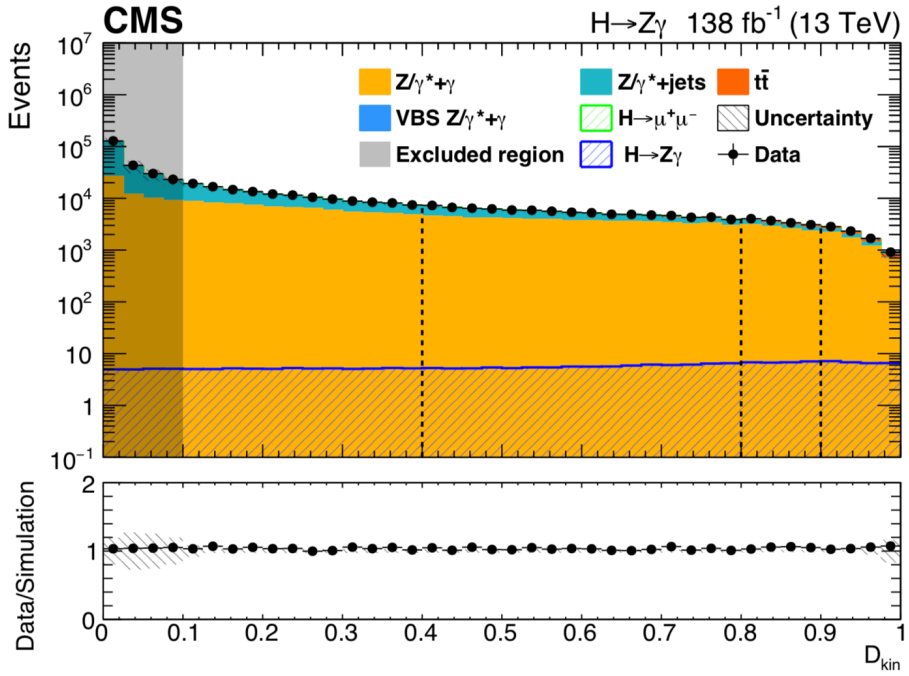
$H \to Z\gamma$ analysis

- As more data is being collected at LHC, becoming possible to see $H \to Z\gamma$.
- Run-2 showed an excess for both CMS and ATLAS
  - Run-2 CMS: Observed (expected) significance is 2.7σ (1.2σ).
  - Run-2 ATLAS: Observed (expected) significance is 2.2σ (1.2σ).
  - With Run-2 CMS+ATLAS, observed significance above 3 σ.

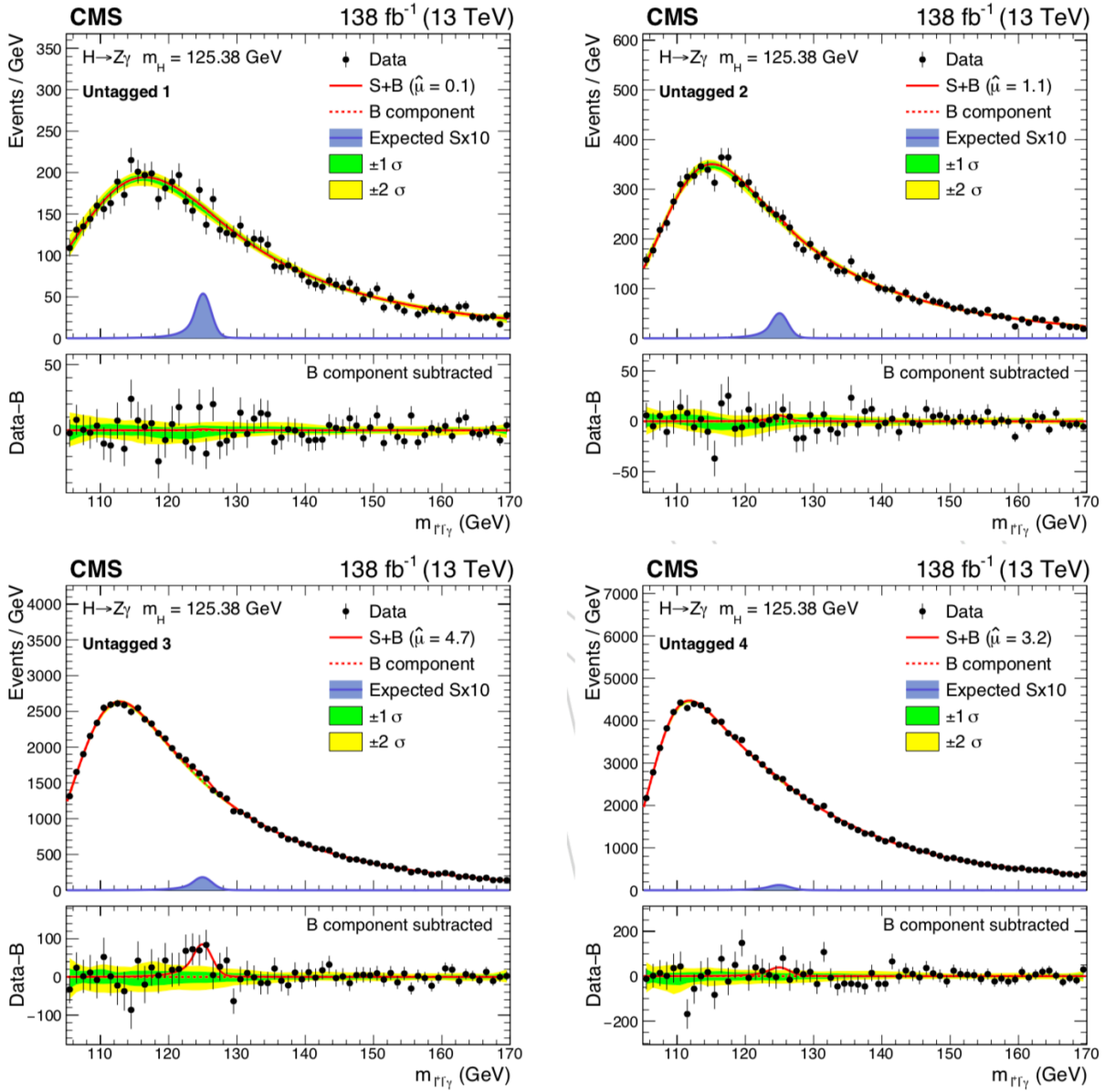


$H \to Z\gamma$ analysis

- Analysis uses BDT score to divide categories and then fits the categories.
- BDT is trained as a signal/background classifier.

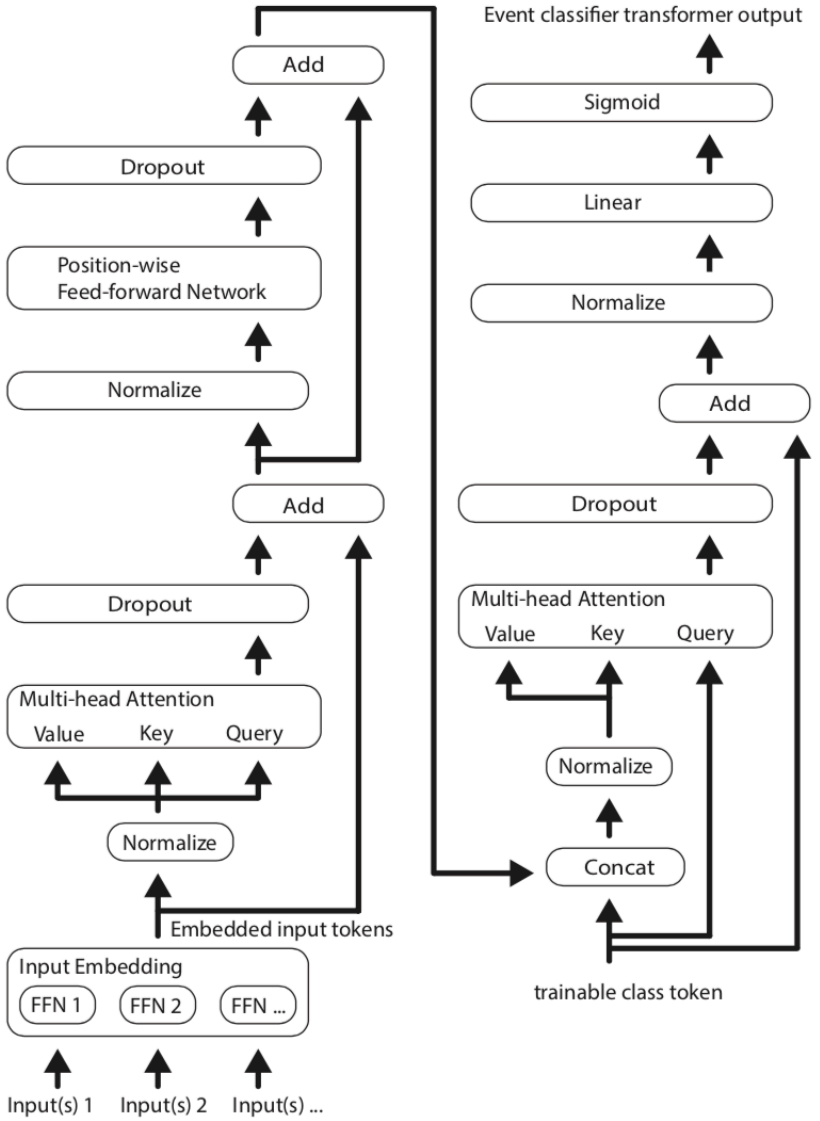


- Could a transformer be used?
  - ➢Started to look into it in 2023.



Event Classifier Transformer network (ETN)

- Tried out a **minimal** version of transformer to classify between signal and background.
  - ➢Because more weights means more training data....
- Each input feature of event is converted to N-vector through feed-forward network "embedding".
- Class token is given to decoder that "tells" network to predict signal probability.
- Output is signal probability.



Special techniques for training network

- Loss are used to train networks, where networks are trained to lower loss.
- Common loss is binary cross-entropy (BCE) $\hat{y}$: ML output y: Truth

$$\mathrm{BCE}(\hat{y}, y) = -y \ln(\hat{y}) - (1-y) \ln(1-\hat{y}),$$

Minimum when $\dfrac{\partial \mathrm{BCE}}{\partial \hat{y}} = 0 \longrightarrow \dfrac{\hat{y}-y}{\hat{y}\,(1-\hat{y})} = 0 \longrightarrow$ Minimum when $\hat{y} = y$

- Use "extreme" loss instead of commonly used binary cross-entropy loss for training network

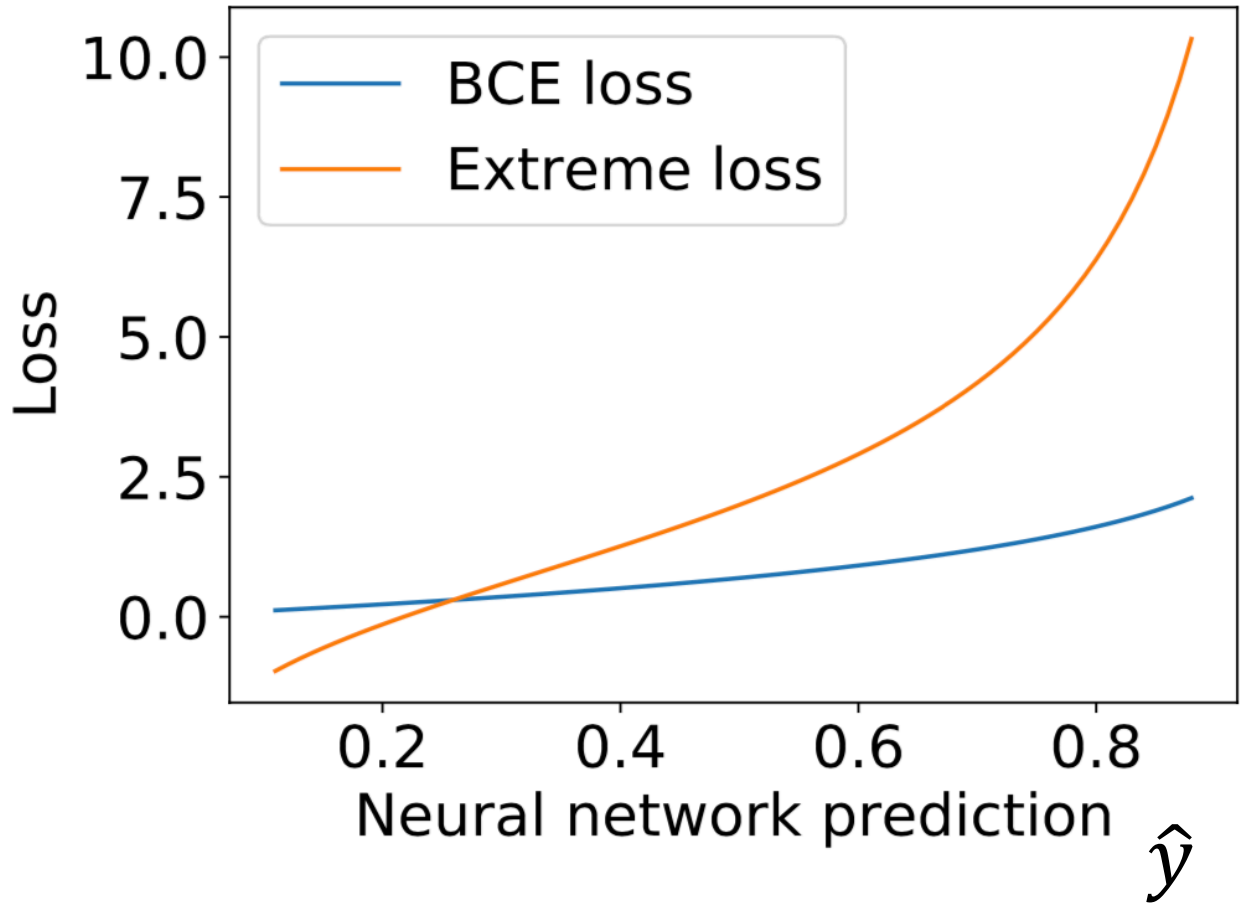
$$\begin{aligned} E\,(\hat{y}, y) &= \int \frac{\hat{y}-y}{\hat{y}^2\,(\hat{y}-1)^2} d\hat{y} \\ &= -y \left[-\frac{1}{\hat{y}} - \ln\,(1-\hat{y}) + \ln\,(\hat{y}) \right] \\ &\quad -(1-y) \left[\frac{1}{\hat{y}-1} + \ln\,(1-\hat{y}) - \ln\,(\hat{y}) \right]. \end{aligned}$$

Special techniques for training network

$$E(\hat{y}, y) = \int \frac{\hat{y} - y}{\hat{y}^2 (\hat{y} - 1)^2} d\hat{y}$$
$$= -y \left[-\frac{1}{\hat{y}} - \ln(1 - \hat{y}) + \ln(\hat{y}) \right]$$
$$- (1 - y) \left[\frac{1}{\hat{y} - 1} + \ln(1 - \hat{y}) - \ln(\hat{y}) \right].$$

- Extreme loss penalizes more than BCE loss for background events with high ML score.
  - Can be helpful when one is sensitive to backgrounds that have high ML score.
  - Significance: ~$N_{\text{signal}}/\sqrt{N_{\text{background}}}$
    - High significance means less of a probability that signal is from background statistical fluctuation.
    - If N_{signal} is fixed, need to work hard in reducing background.

When y=0



Special techniques for training network

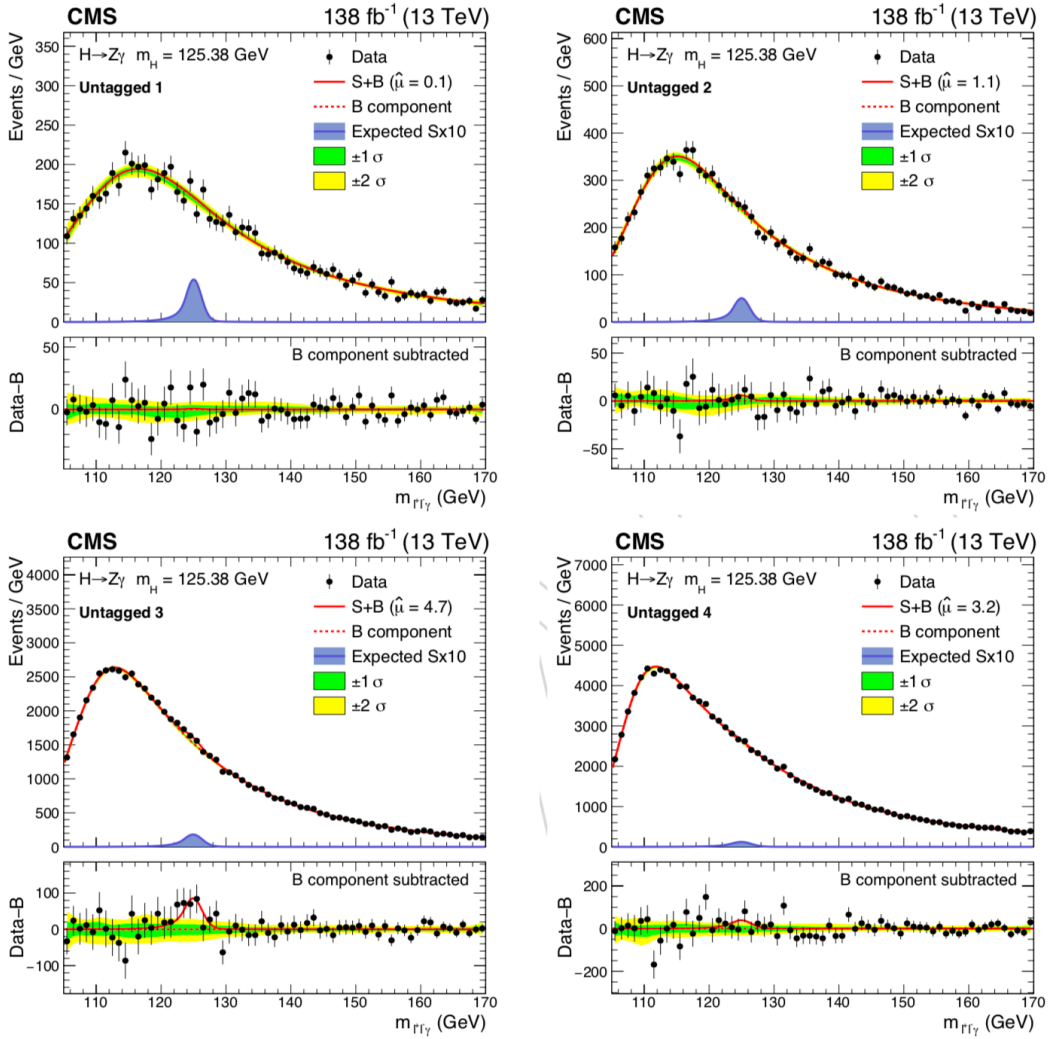
- It is preferred that ML output is not be correlated with $m_{\ell\ell\gamma}$.
  - ➢If not, a ML bin could have background peak at the Higgs mass!

https://arxiv.org/abs/2001.05310

- Can use Disco (Distance Correlation regularization) in training to decorrelate network output with $m_{\ell\ell\gamma}$.
  - ➢Disco measures the dependence between ML output and $m_{\ell\ell\gamma}$, where value of Disco is zero if and only if ML output and are $m_{\ell\ell\gamma}$ independent.
- Method of using Disco

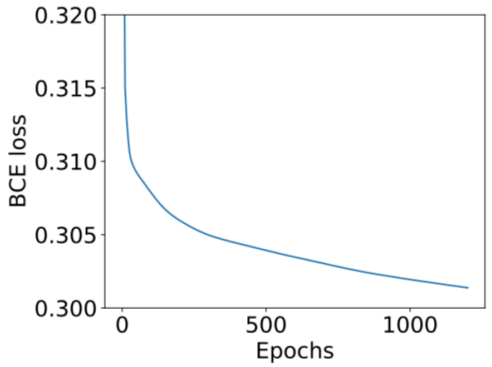
$$\text{Loss} = \text{Loss}_{\text{classifier}}\left(\hat{y}, y\right) + \lambda \cdot \text{DisCo}\left(\text{mass}, \hat{y}\right).$$

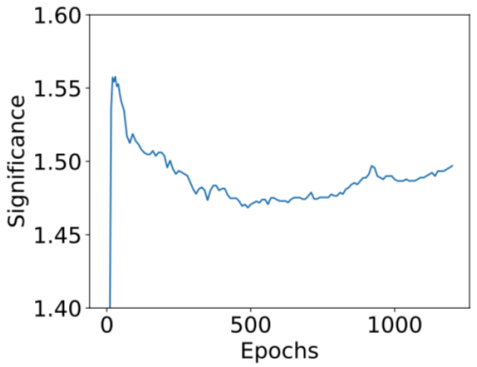
BCE loss or Extreme loss



Special techniques for training network

- Training is done with many epochs
  - Train NN with data (epoch 1)
  - Train NN with data again (epoch 2)
  - Train NN with data again (epoch 3)
  - …



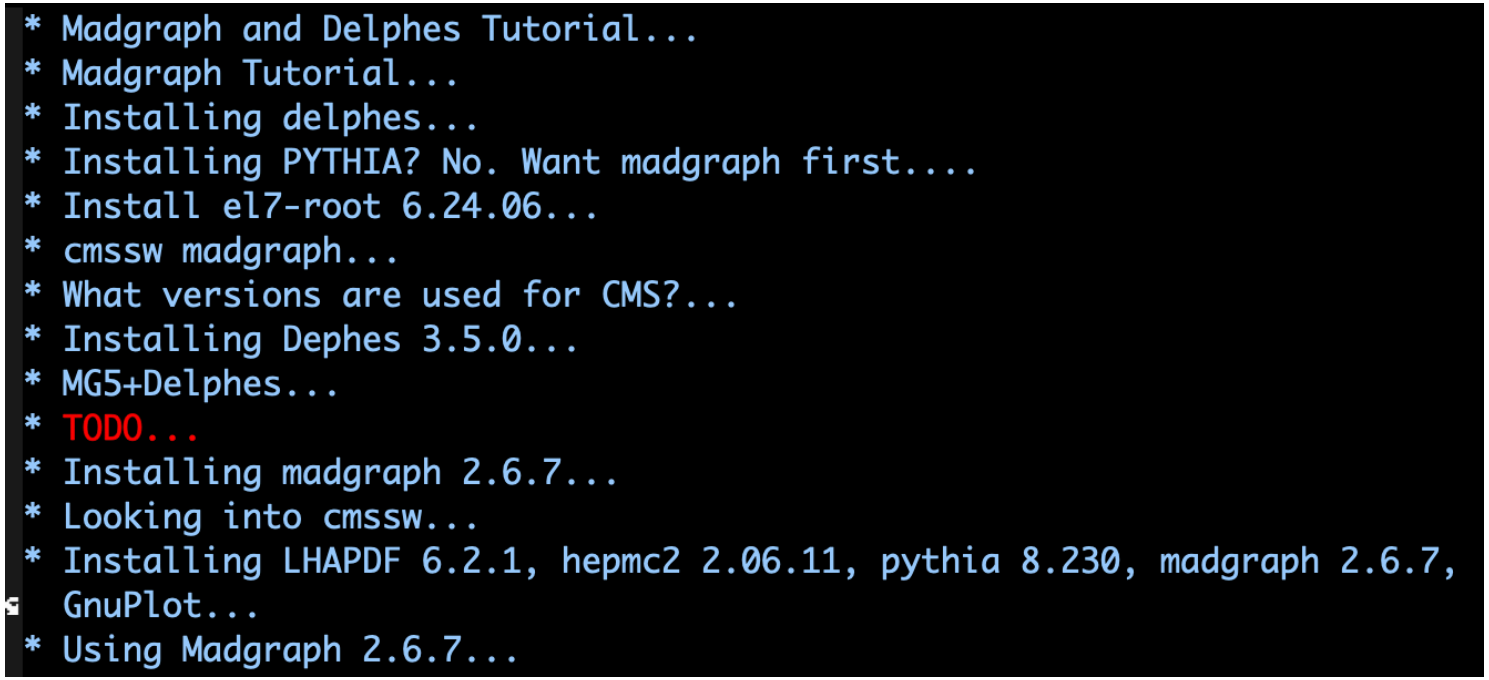


- Choosing best NN model based on significance during epoch.
  - For each epoch, evaluate significance.
  - Best NN model is selected by epoch which had highest significance.
  - Normally best NN model is selected by lowest loss. But highest significance and lowest loss do not always agree.
- Calculation of combined significance
  - Divide sample into bins based on ML output, where each bin has equal number of signal events.
  - Calculate significance for each bin: $$\text{Significance} = \sqrt{2\left[(N_S + N_B)\ln\left(1 + \frac{N_S}{N_B}\right) - N_S\right]},$$
  - Combine significances: $$\text{Total significance} = \sqrt{\sum_i^n (\text{Significance}_i)^2},$$

Another way in calculating significance that works event with small number of events

MC Dataset

- Generated MC using Madgraph, Pythia8, and Delphes3
  - Found a python2 bug for Madgraph-pythia8, where pythia8 command sequence would become mixed. Fixed Madgraph code to resolve issue. Shouldn't be an issue with python3.
- Needed to learn how to install Madgraph, Pythia8, and Delphes3...
  - Took a bit of time...



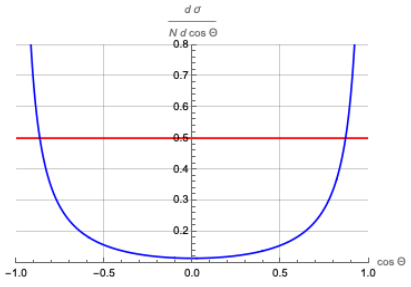
My Emacs Org document on figuring out how to install Madgraph, Pythia8, and Delphes3...

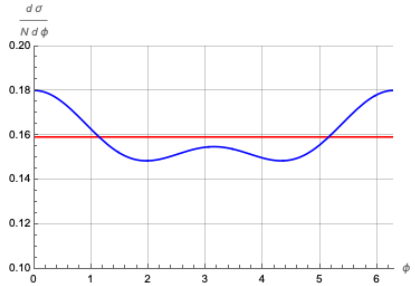
Learning how to analyze LHE, HEP3, and Delphes files

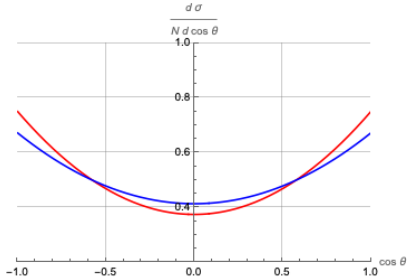
- Learn how to analyze LHE files, HEP3 files, and Delphes files...
- Tried comparing theory predictions with generated files.

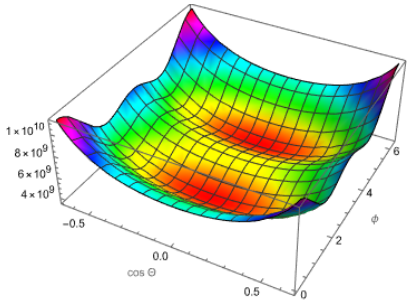
Used Wolfram Engine with Jupyter

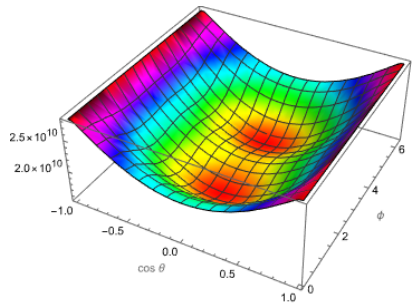
- Integrate to get angular distributions
  - Used $\sqrt{\hat{s}} = 125$ GeV and $Z \to d\bar{d}$ couplings
  - Limit Θ to 0.999415 (p_T^γ =1GeV) $\cos\Theta = \sqrt{1 - 4p_{\gamma T}^2 \hat{s}/(\hat{s} - m_Z^2)^2}$

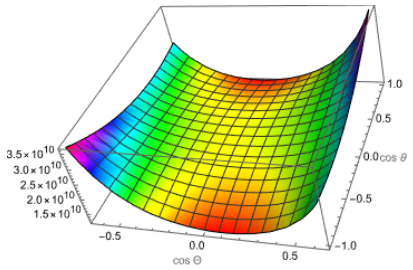












Drell-Yan (+ISR) process

[arXiv: J. Gainer]

$$\frac{1}{N'}\frac{d\sigma}{d\cos\Theta\, d\cos\theta\, d\phi} = (g_r^2+g_\ell^2)(g_R^2+g_L^2)\,\mathcal{G}_1 + (g_r^2-g_\ell^2)(g_R^2-g_L^2)\mathcal{G}_2\,,$$

with

$$\begin{aligned}\mathcal{G}_1 &= \Big[(m_{12}^4+\hat{s}^2)(3+\cos 2\theta)(4\csc^2\Theta-2) \\ &\quad +8m_{12}^2\,\hat{s}\,\sin^2\theta(2+\cos 2\phi) \\ &\quad +8\,m_{12}\sqrt{\hat{s}}\,\left(m_{12}^2+\hat{s}\right)\cot\Theta\,\sin 2\theta\,\cos\phi\Big]\,, \\ \mathcal{G}_2 &= 16\csc\Theta\,\Big[(m_{12}^4+\hat{s}^2)\cos\theta\cot\Theta \\ &\quad +\,m_{12}\sqrt{\hat{s}}\,\left(m_{12}^2+\hat{s}\right)\,\sin\theta\cos\phi\Big]\,,\end{aligned}$$

Gluon fusion production of Higgs

$$\frac{1}{N}\frac{d\sigma}{d\cos\Theta\, d\cos\theta\, d\phi} = (1+\cos^2\theta)\,,$$

Generation of Drell-Yan (+ISR) [$d\bar{d}$] sample

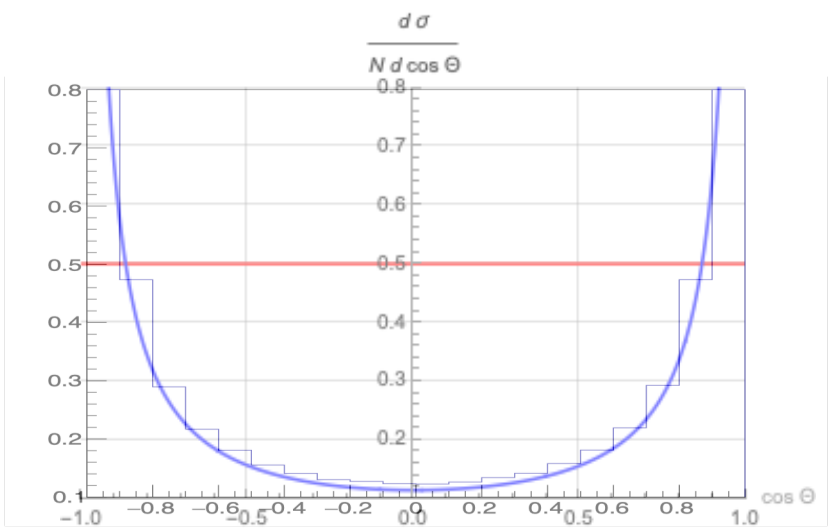
- Generated samples to customize production and modify generator cuts
  - ➢MadGraph5 2.9.11, Pythia 8.307, LHAPDF 6.5.2, Delphes 3.5.0

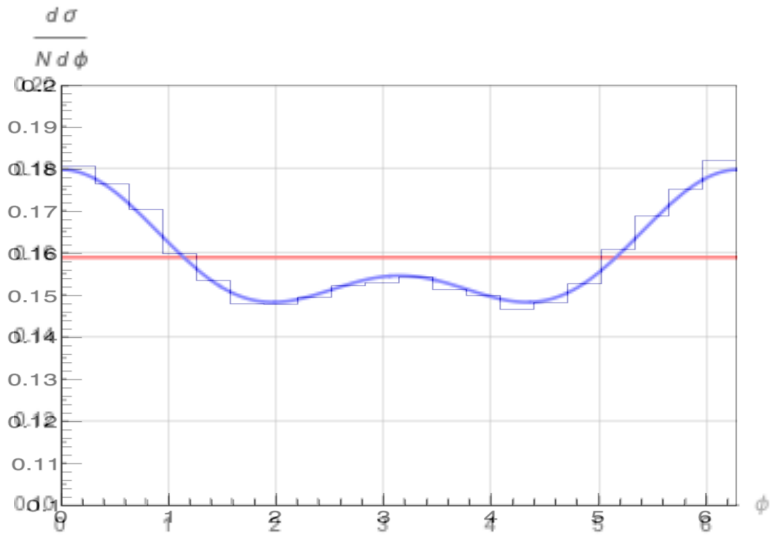
(Loosened default (γ, ℓ) pT, eta, $\Delta R(\gamma, \ell)$ generator cuts)

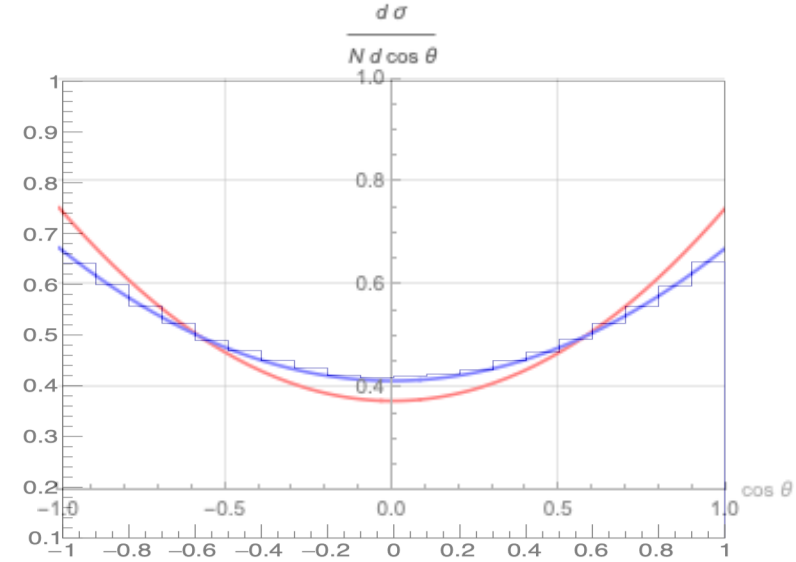
- Generated Drell-Yan (+ISR) from $d\bar{d}$ production with $\sqrt{\hat{s}} = 125$ GeV, $p_T^{\gamma} \geq 1$GeV.

```
generate d d~ > z a, z > l+ l-
```

  - ➢Calculated angles using generator information (gen. info). [Before showering with Pythia]







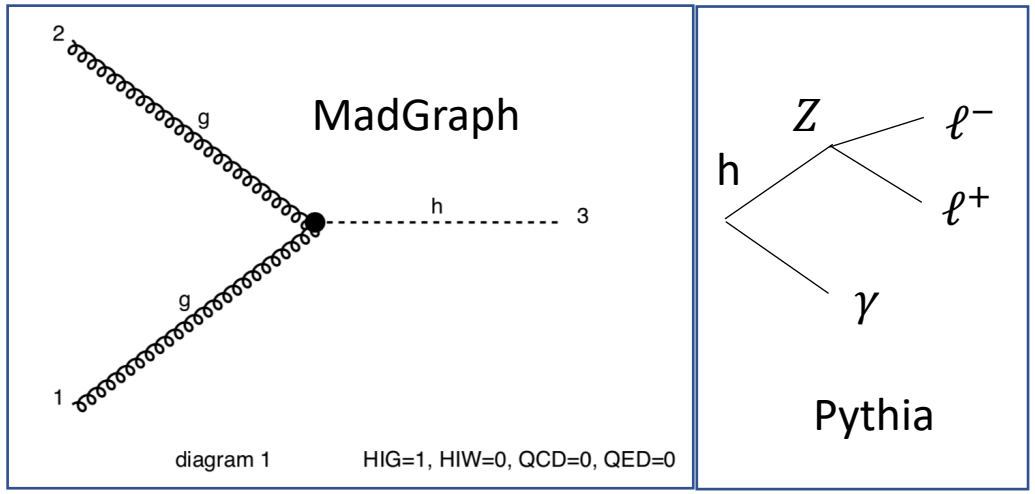
- Agrees well with integration calculations.

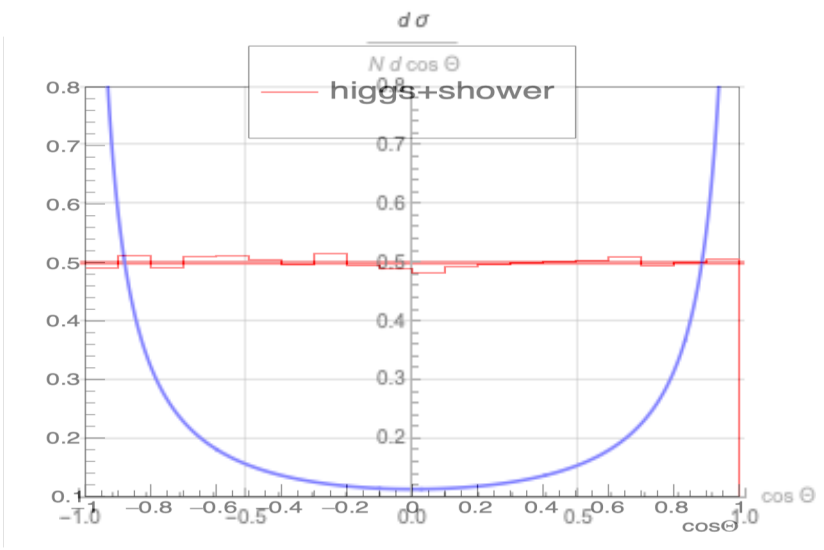
Generation of gluon fusion production of Higgs sample

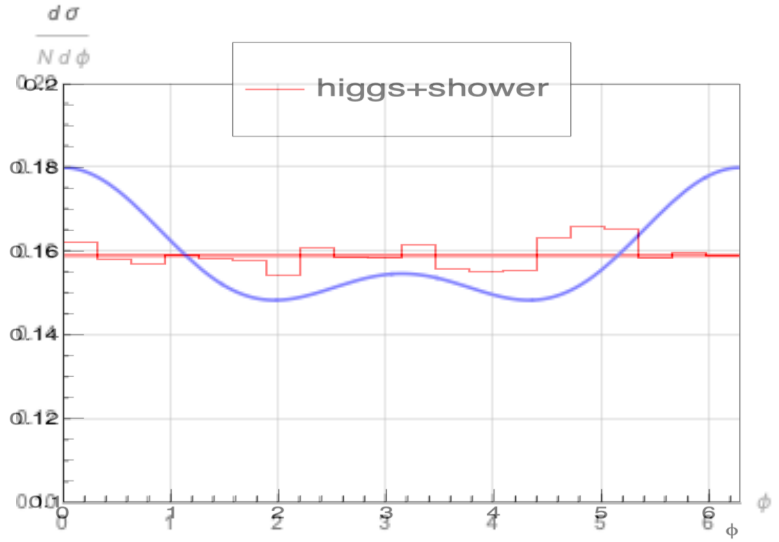
- Gluon fusion produced from Madgraph

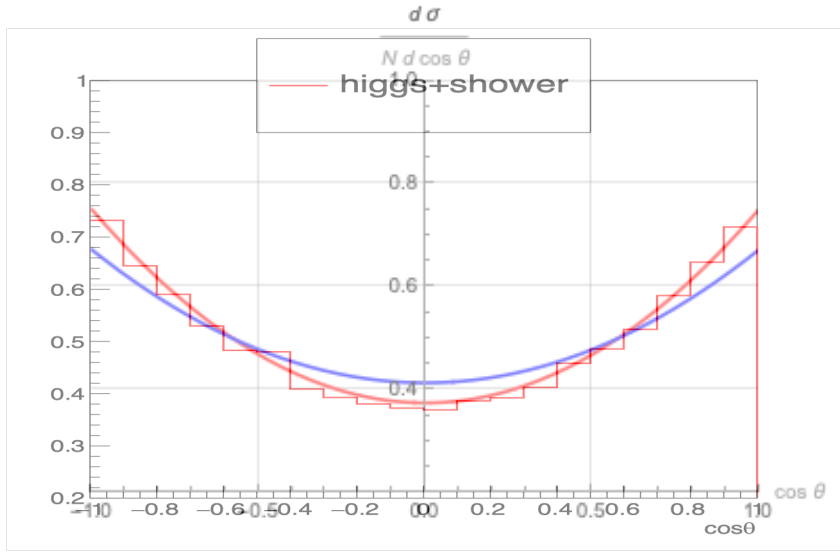
```
import model heft; generate p p > h
```

- Higgs decayed by Pythia.
- Showers included by Pythia.
- Calculated angles using gen. info.



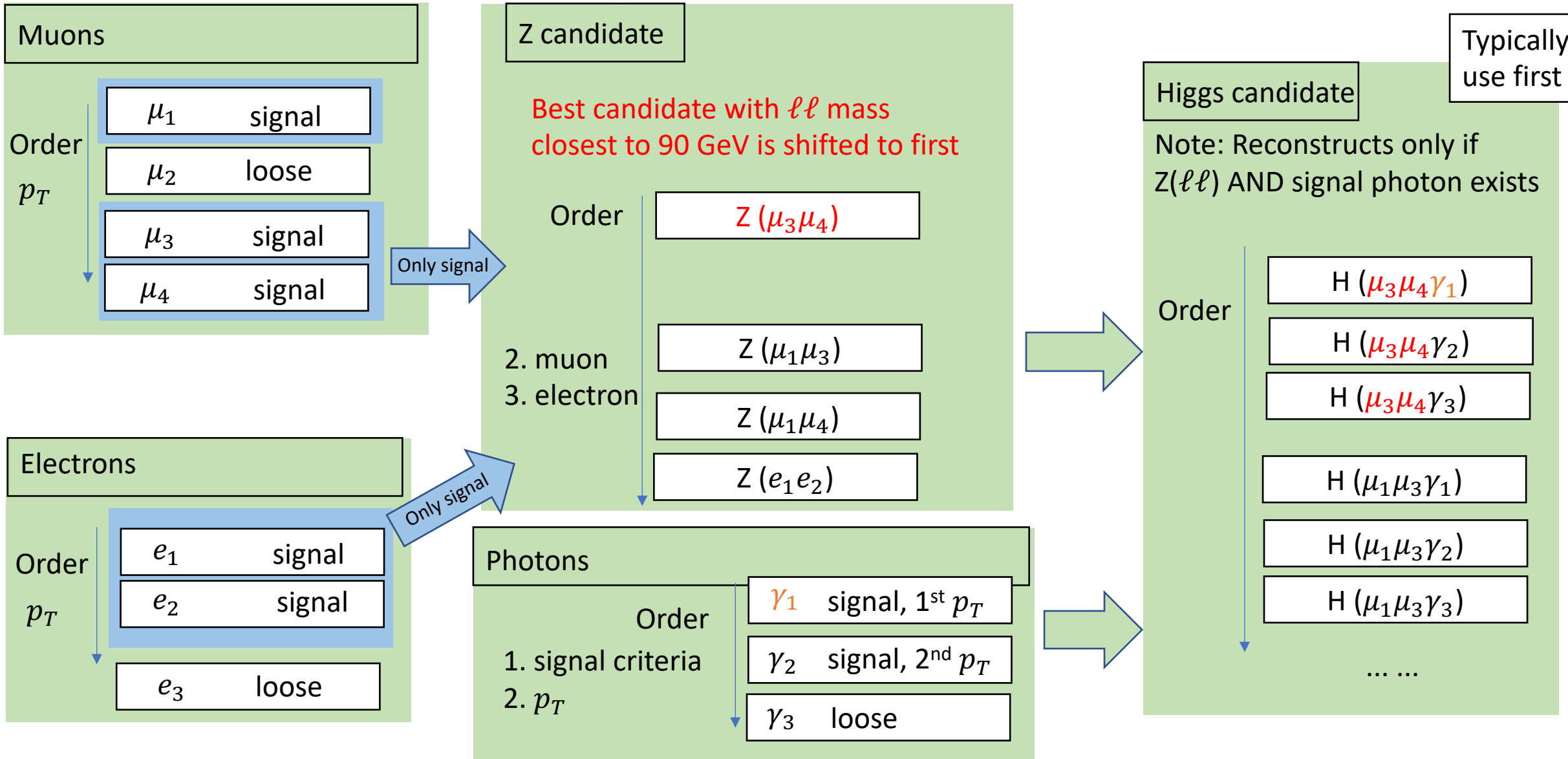






- Agrees well with integration calculations.

Implementing $H \to Z\gamma$ reconstruction



Implementing input features for ML input

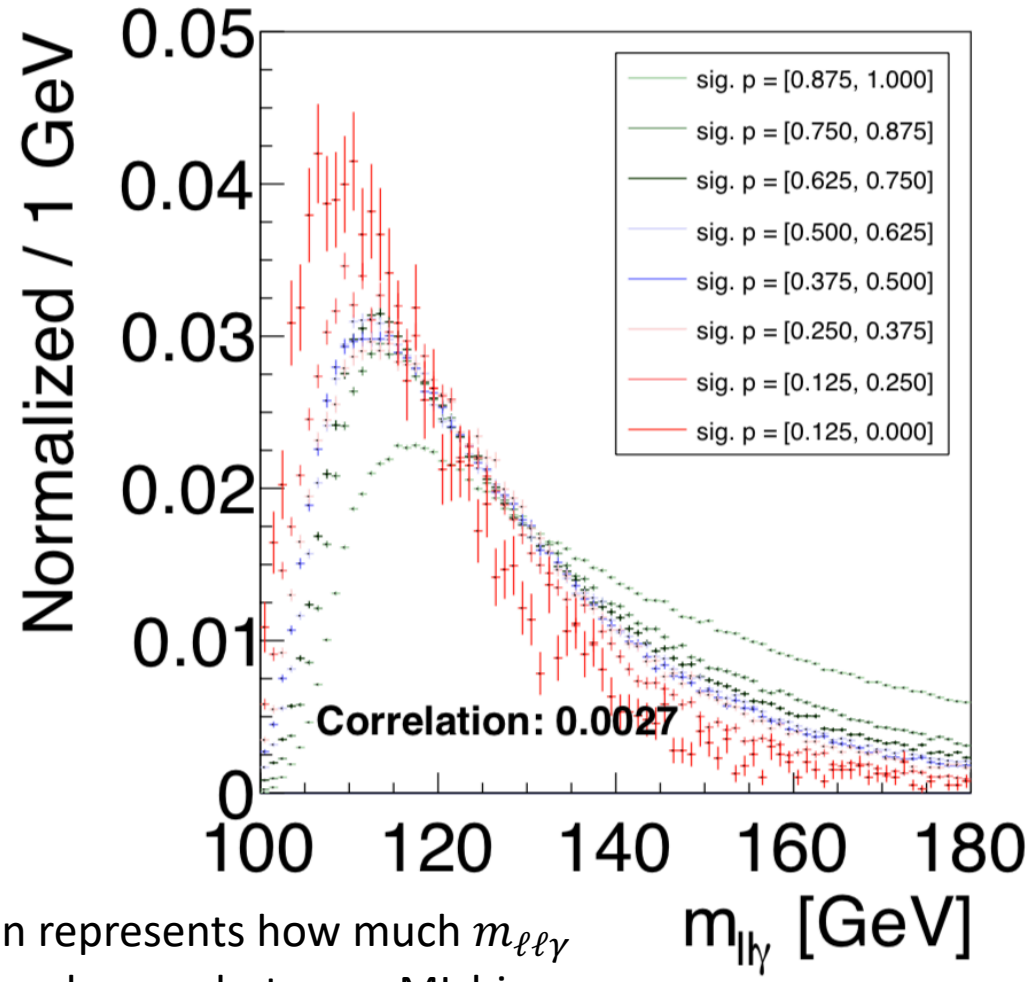
- η_γ, $\eta_\ell^{\text{leading }\ell}$, $\eta_\ell^{\text{subleading }\ell}$: Pseudorapidity angle of the photon and leptons.
- Minimum $\Delta R\,(\ell^\pm, \gamma)$, Maximum $\Delta R\,(\ell^\pm, \gamma)$: Minimum and maximum ΔR between leptons and the photon.
- Flavor of ℓ: Flavor of lepton used to reconstruct the Z boson, either being an electron or a muon.
- $p_T^{\ell\ell\gamma}/m_{\ell\ell\gamma}$: p_T of the reconstructed Higgs boson candidate divided by the mass.
- $p_{Tt}^{\ell\ell\gamma}$: Projection of the reconstructed Higgs boson p_T to the di-lepton thrust axis [29].
- $\sigma_m^{\ell\ell\gamma}$: Mass reconstruction error of the $\ell\ell\gamma$ candidate estimated by binning signal events in η and p_T for the photon and leptons, and measuring the signal's mass width for each bin.
- $\cos\Theta$, $\cos\theta$, ϕ: Production and decay angles that determine the differential cross section of $H \to Z\,(\ell^+\ell^-)\,\gamma$ and $qq \to Z\,(\ell^+\ell^-)\,\gamma$ [30, 31].
- $p_T^\gamma/m_{\ell\ell\gamma}$, $p_T^{\text{leading }\ell}$, $p_T^{\text{subleading }\ell}$: Momenta of the photon and leptons.

← Delphes3 does not have photon resolution...
Made a custom photon resolution variable.

Background $m_{\ell\ell\gamma}$ distribution for BDT

- Trained BDT with 12 below features
  - η_γ, $\eta_\ell^{\text{leading }\ell}$, $\eta_\ell^{\text{subleading }\ell}$: Pseudorapidity angle of the photon and leptons.
  - Minimum $\Delta R(\ell^\pm, \gamma)$, Maximum $\Delta R(\ell^\pm, \gamma)$: Minimum and maximum ΔR between leptons and the photon.
  - Flavor of ℓ: Flavor of lepton used to reconstruct the Z boson, either being an electron or a muon.
  - $p_T^{\ell\ell\gamma}/m_{\ell\ell\gamma}$: p_T of the reconstructed Higgs boson candidate divided by the mass.
  - $p_{Tt}^{\ell\ell\gamma}$: Projection of the reconstructed Higgs boson p_T to the di-lepton thrust axis [29].
  - $\sigma_m^{\ell\ell\gamma}$: Mass reconstruction error of the $\ell\ell\gamma$ candidate estimated by binning signal events in η and p_T for the photon and leptons, and measuring the signal's mass width for each bin.
  - $\cos\Theta$, $\cos\theta$, ϕ: Production and decay angles that determine the differential cross section of $H \to Z(\ell^+\ell^-)\gamma$ and $qq \to Z(\ell^+\ell^-)\gamma$ [30, 31].

When binning events by BDT score with equal number of signal events, background $m_{\ell\ell\gamma}$ distributions look like below.



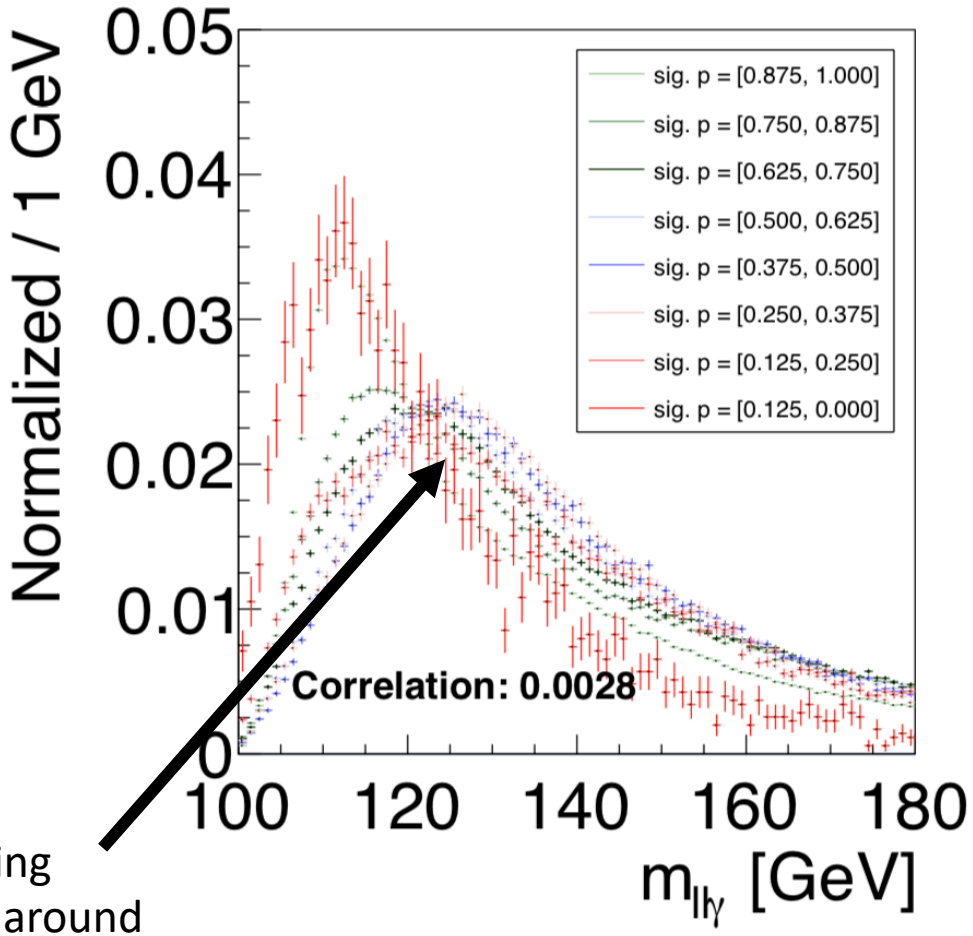
Correlation represents how much $m_{\ell\ell\gamma}$ distribution changes between ML bins.
Smaller means less of a change.

Background $m_{\ell\ell\gamma}$ distribution for BDT

- Trained BDT with 15 below features

  - η_γ, $\eta_\ell^{\text{leading }\ell}$, $\eta_\ell^{\text{subleading }\ell}$: Pseudorapidity angle of the photon and leptons.
  - Minimum $\Delta R(\ell^\pm, \gamma)$, Maximum $\Delta R(\ell^\pm, \gamma)$: Minimum and maximum ΔR between leptons and the photon.
  - Flavor of ℓ: Flavor of lepton used to reconstruct the Z boson, either being an electron or a muon.
  - $p_T^{\ell\ell\gamma}/m_{\ell\ell\gamma}$: p_T of the reconstructed Higgs boson candidate divided by the mass.
  - $p_{Tt}^{\ell\ell\gamma}$: Projection of the reconstructed Higgs boson p_T to the di-lepton thrust axis [29].
  - $\sigma_m^{\ell\ell\gamma}$: Mass reconstruction error of the $\ell\ell\gamma$ candidate estimated by binning signal events in η and p_T for the photon and leptons, and measuring the signal's mass width for each bin.
  - $\cos\Theta$, $\cos\theta$, ϕ: Production and decay angles that determine the differential cross section of $H \to Z(\ell^+\ell^-)\gamma$ and $qq \to Z(\ell^+\ell^-)\gamma$ [30, 31].
  - $p_T^\gamma/m_{\ell\ell\gamma}$, $p_T^{\text{leading }\ell}$, $p_T^{\text{subleading }\ell}$: Momenta of the photon and leptons.

When binning events by BDT score with equal number of signal events, background $m_{\ell\ell\gamma}$ distributions look like below.



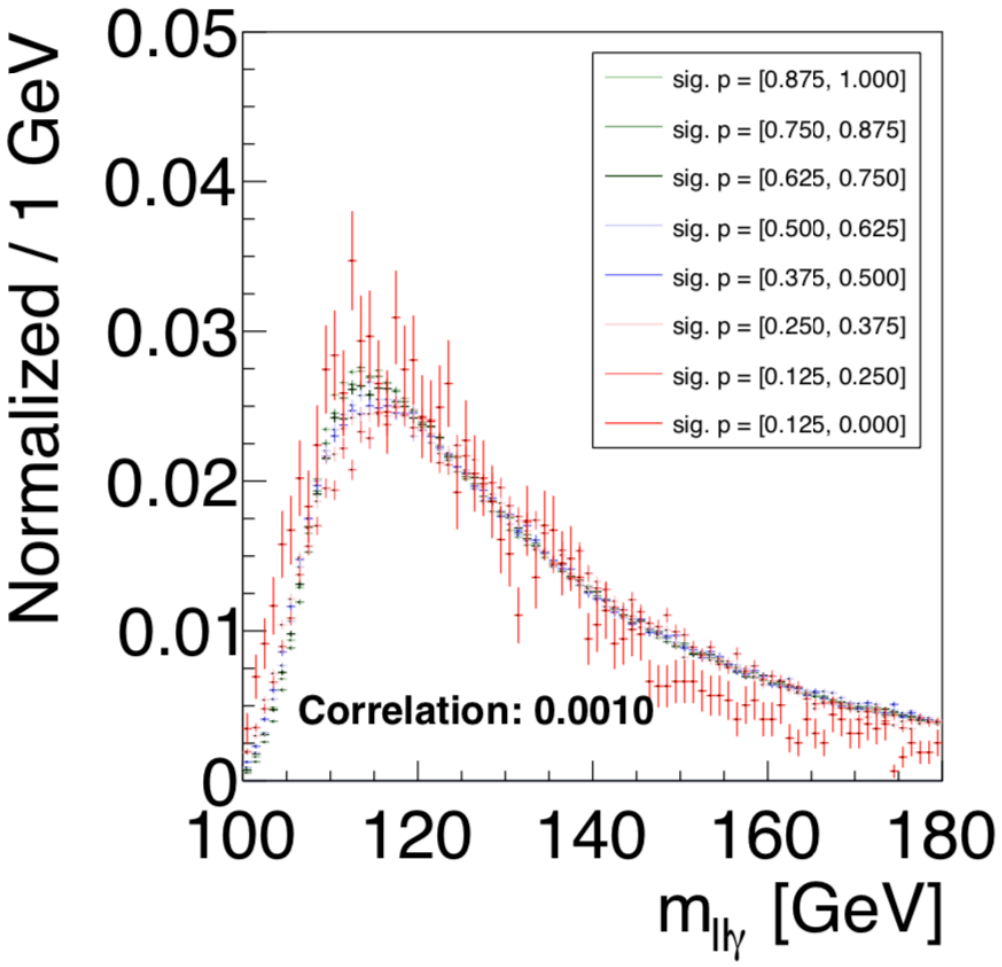
Photon pT is making background peak around Higgs mass....

Background $m_{\ell\ell\gamma}$ distribution for ETN with Exteme+Disco loss training.

- Trained ETN with 15 below features

  - η_γ, $\eta_\ell^{\text{leading }\ell}$, $\eta_\ell^{\text{subleading }\ell}$: Pseudorapidity angle of the photon and leptons.
  - Minimum $\Delta R(\ell^\pm, \gamma)$, Maximum $\Delta R(\ell^\pm, \gamma)$: Minimum and maximum ΔR between leptons and the photon.
  - Flavor of ℓ: Flavor of lepton used to reconstruct the Z boson, either being an electron or a muon.
  - $p_T^{\ell\ell\gamma}/m_{\ell\ell\gamma}$: p_T of the reconstructed Higgs boson candidate divided by the mass.
  - $p_{Tt}^{\ell\ell\gamma}$: Projection of the reconstructed Higgs boson p_T to the di-lepton thrust axis [29].
  - $\sigma_m^{\ell\ell\gamma}$: Mass reconstruction error of the $\ell\ell\gamma$ candidate estimated by binning signal events in η and p_T for the photon and leptons, and measuring the signal's mass width for each bin.
  - $\cos\Theta$, $\cos\theta$, ϕ: Production and decay angles that determine the differential cross section of $H \to Z(\ell^+\ell^-)\gamma$ and $qq \to Z(\ell^+\ell^-)\gamma$ [30, 31].
  - $p_T^\gamma/m_{\ell\ell\gamma}$, $p_T^{\text{leading }\ell}$, $p_T^{\text{subleading }\ell}$: Momenta of the photon and leptons.

When binning events by BDT score with equal number of signal events, background $m_{\ell\ell\gamma}$ distributions look like below.



Comparison of ML techniques

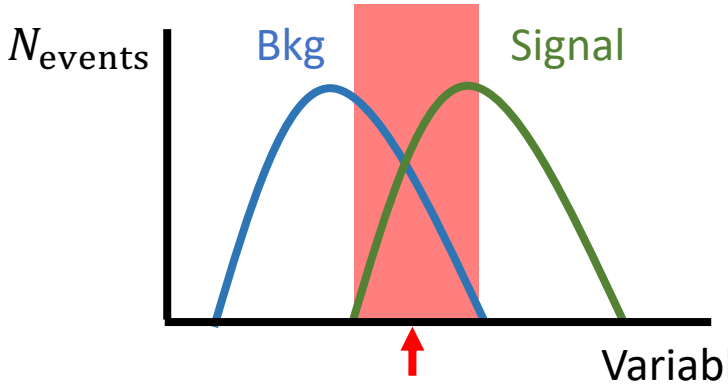
- ETN generally shows best performance.
- DFNN (Deep feed-forward network) also good when using 11 inputs.
  - ➢(N, 3N, 9N, 3N, 1) network
- Extreme loss can be helpful in certain cases: FNN, Ext+Disco
- Disco training shows lowest background correlation.
- ETN probably working better than DFNN with 15 inputs because
  - ➢ETN needs 265 additional weights (5% increase)
  - ➢DFNN needs 4,671 additional weights (55% increase)

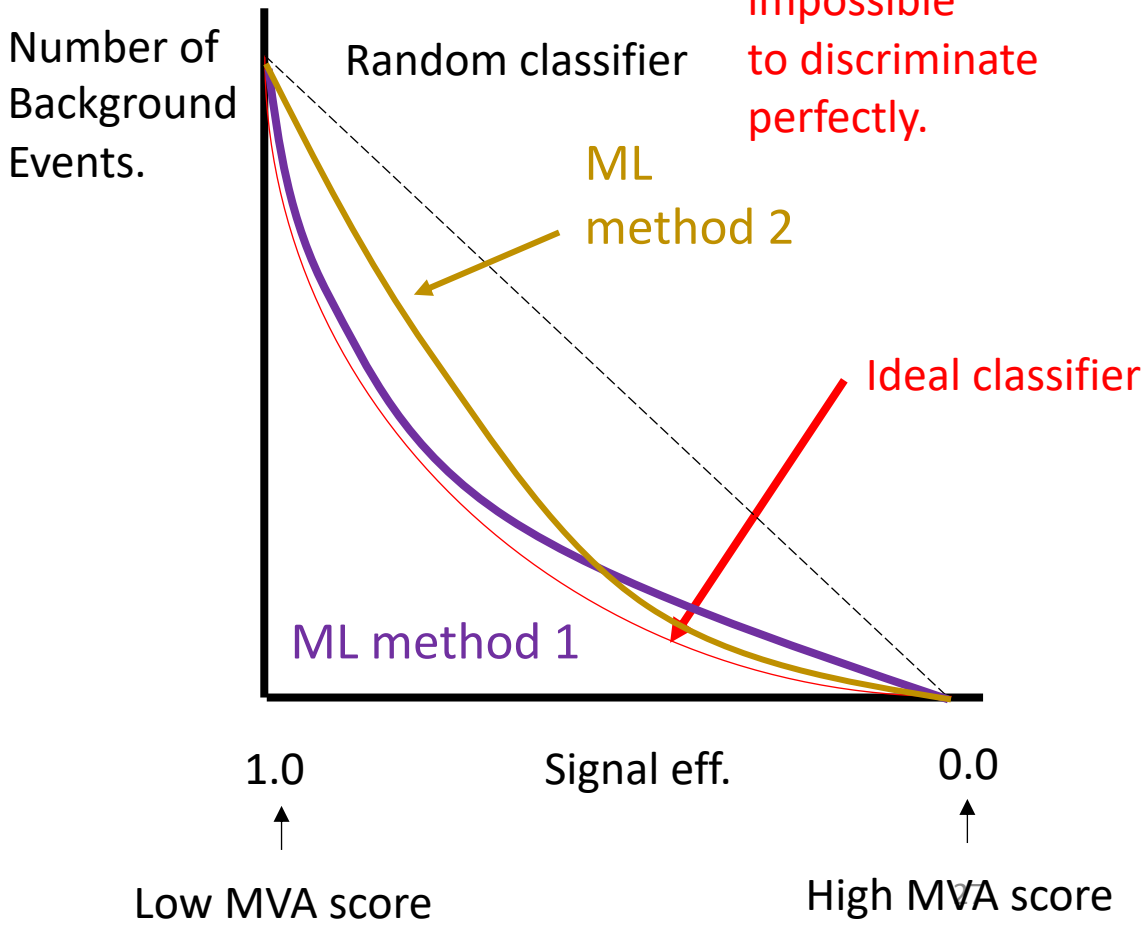
Might be possible to tune DFNN structure to get similar performance with ETN

	Technique	Loss	Signi.	AUC (%)	Bkg. Corr. (10^{-3})	Best epoch
	Random	N.A	0.50	50.0	0	N.A.
11 inputs	BDT	N.A	1.17	75.6	3.3	N.A.
11 inputs	XGBoost	N.A	1.49	75.8	2.7	N.A.
11 inputs	FNN	BCE	1.43	75.2	2.8	22
11 inputs	FNN	Ext	1.45	75.4	2.7	107
11 inputs	DFNN	BCE	**1.50**	**76.0**	2.0	907
11 inputs	DFNN	Ext	1.46	75.7	2.0	487
11 inputs	ETN	BCE	**1.51**	**76.2**	2.3	570
11 inputs	ETN	Ext	1.48	75.7	2.3	307
15 inputs	FNN	BCE+DisCo	1.35	75.0	**0.7**	633
15 inputs	FNN	Ext+DisCo	1.46	74.7	**0.8**	327
15 inputs	DFNN	BCE+DisCo	1.46	75.1	**1.0**	550
15 inputs	DFNN	Ext+DisCo	1.47	75.7	**1.0**	273
15 inputs	ETN	BCE+DisCo	**1.52**	75.6	**1.0**	670
15 inputs	ETN	Ext+DisCo	**1.50**	75.4	**1.0**	623

What is making the difference?

- Machine learning (ML) methods try to approach the ideal classifier limit.
- However depending on the "optimization/fitting", different ML methods can be closer to the limit in different regions.
  - ML method 1: Closer to limit for low ML scores.
  - ML method 2: Closer to limit for high ML scores.
- Two considerations for ML
  1. Which ML can generally get closest to the ideal limit? (=AUC metric)
  2. Which ML can get closest to the ideal limit in the the region that is important. (In this case, significance metric)
- Can try to find good structures that scale well with number of inputs. Transformers are good at this.
- Can try to tweak ML to get good performance in important region. Extreme loss can help.





Summary

Training towards significance with the decorrelated event classifier transformer neural network

Accepted by PRD on April 24, 2024

Jaebak Kim*
Department of Physics, University of California, Santa Barbara, CA, U.S.A.
(Dated: April 22, 2024)

Experimental particle physics uses machine learning for many of tasks, where one application is to classify signal and background events. The classification can be used to bin an analysis region to enhance the expected significance for a mass resonance search. In natural language processing, one of the leading neural network architectures is the transformer. In this work, an event classifier transformer is proposed to bin an analysis region, in which the network is trained with special techniques. The techniques developed here can enhance the significance and reduce the correlation between the network's output and the reconstructed mass. It is found that this trained network can perform better than boosted decision trees and feed-forward networks.

- Turned a hobby/interest in neural networks to a paper!
- Needed to learn couple of things on the way...
  - Transformer neural network calculations
  - Installing Madgraph, Pythia8, Delphes3
  - LHE, HEP3, Delphes3 data formats
  - Installing GPU drivers
  - Pytorch, XGBoost, TMVA software
  - BCE loss, Disco loss

ACKNOWLEDGMENTS

I am grateful to Gyuwan Kim in the UCSB Computer Science and Natural Language Processing group for introducing me to the transformer neural network and for revising the paper. I am also grateful to Jeff Richman in

- This paper started from asking "What are you researching?" to a grad student after playing tennis together. Thankful for the grad student! Being curious can be good.

Acknowledgments

- The material is based upon work supported by the U.S. Department of Energy Office of Science, Office of High Energy Physics under Award Number DE-SC0011702.