The use of adversaries of optimal Neural Net training

(https://travellingbuzz.com)

Anton Hawthorne and Martin Sevior

University of Melbourne and Belle(II) collaboration

Computing in High Energy Physics

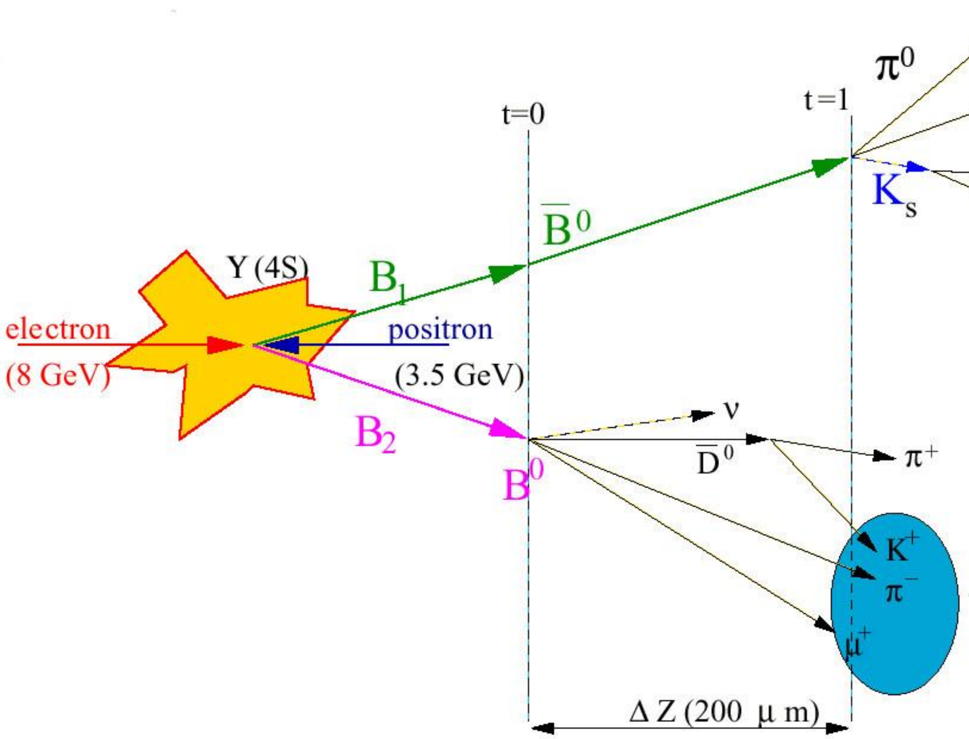
Sophia, Bulgaria, July, 2018

Introduction

This presentation summarizes the M.Phil.of Anton Hawthorne
Detailed write up in arXiv:1712.07790

- Belle
- CP-violation in $\overline{B^0} \to K_s\pi^0$ decays
- Analysis and Backgrounds to $\overline{B^0} \to K_s\pi^0$
- Deep Neural Network vs Shallow and BDT
- Data normalisation
- Performance of Deep Net
- Background sculpting
- Adversarial Neural Net
- Performance of Adversarial Net
- Conclusions

CP-violation in $\overline{B^0} \to K_s\pi^0$ decays

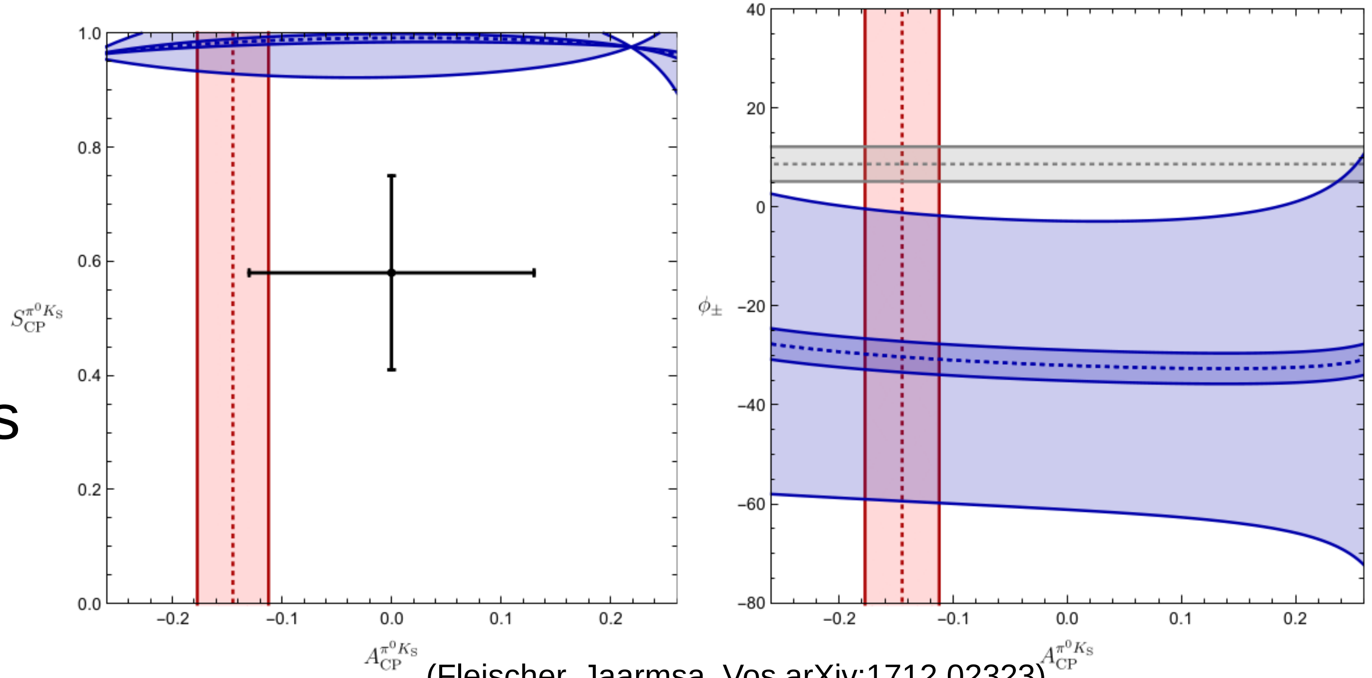


Flavour Tag

Define: $$A(t) = \frac{N\left(\overline{B^0}\to K_S\pi^0\right)(t) - N\left(B^0\to K_S\pi^0\right)(t)}{N\left(\overline{B^0}\to K_S\pi^0\right)(t) + N\left(B^0\to K_S\pi^0\right)(t)}$$

$$A(t) = S_{CP}\sin(\Delta m t) + A_{CP}\cos(\Delta m t)$$

- SM prediction 2.2σ from measurements
- Measurements are statistically limited

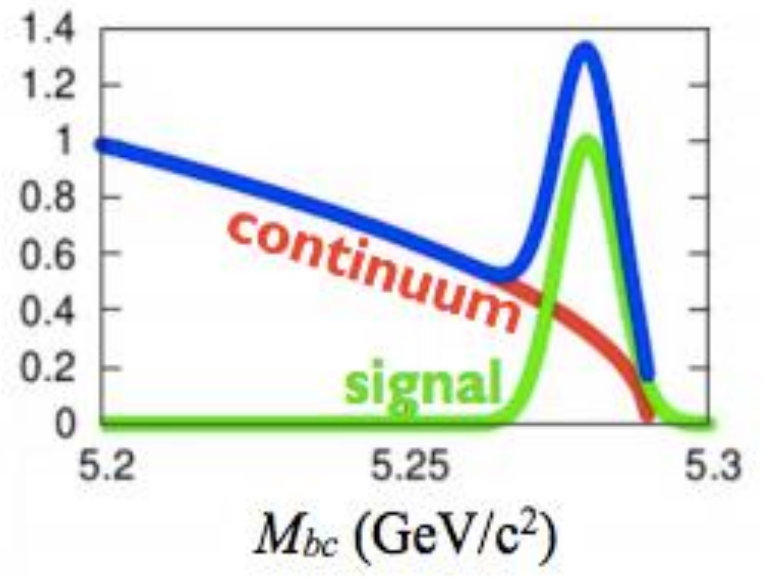


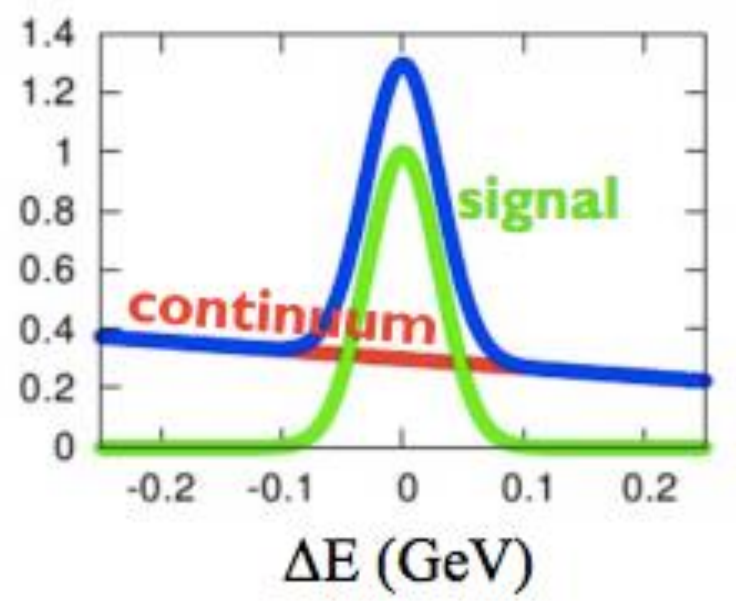
(Fleischer, Jaarsma, Vos arXiv:1712.02323)

Kinematic Variables in B-Factory measurements

$$M_{\rm bc} = \sqrt{E^{*2}_{beam} - p^{*2}_{B}}$$

$$\Delta E = E^*_B - E^*_{beam}$$



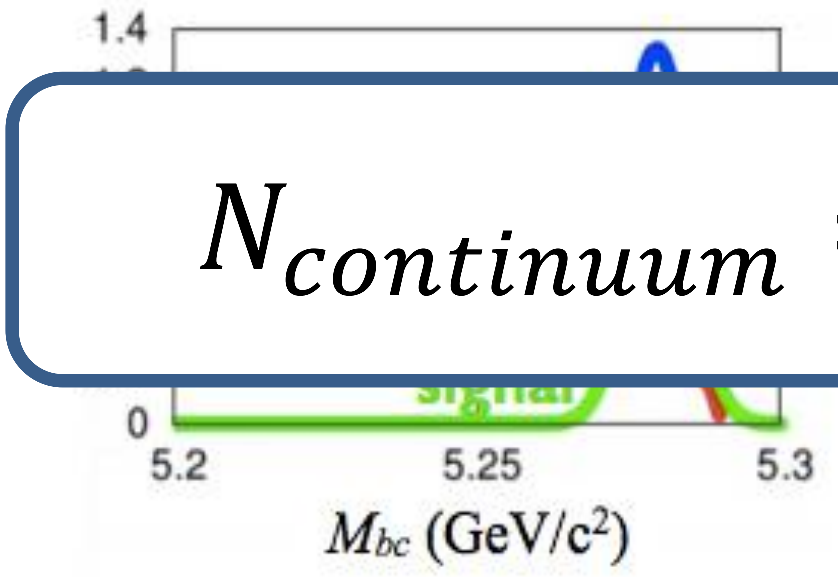


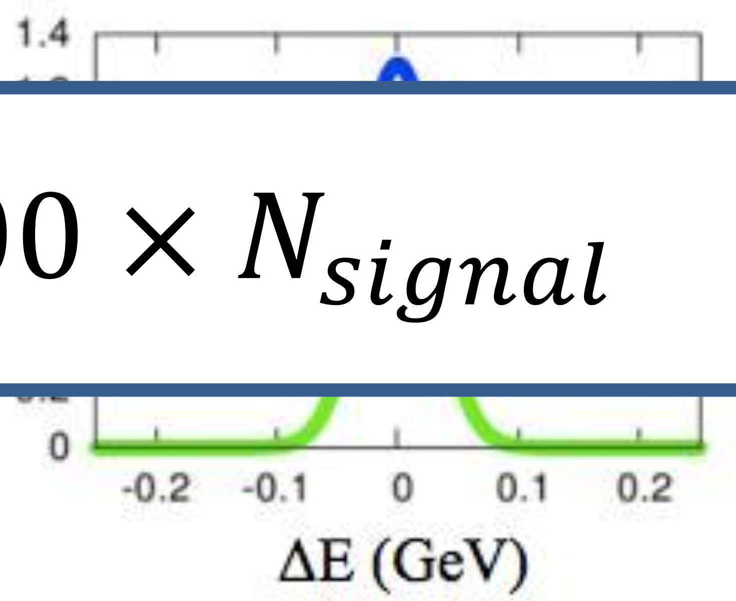
M_{bc} peaks at B mass for fully reconstructed signal
ΔE peaks at zero for fully reconstructed signal

Kinematic Variables in B-Factory measurements

$$M_{\rm bc} = \sqrt{E^{*2}_{beam} - p^{*2}_{B}}$$

$$\Delta E = E^{*}_{B} - E^{*}_{beam}$$



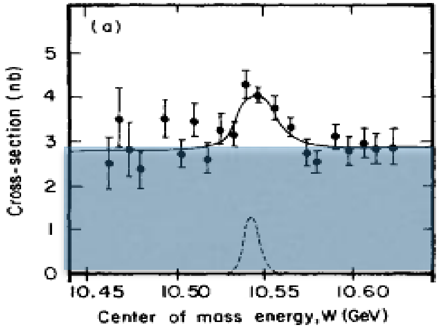


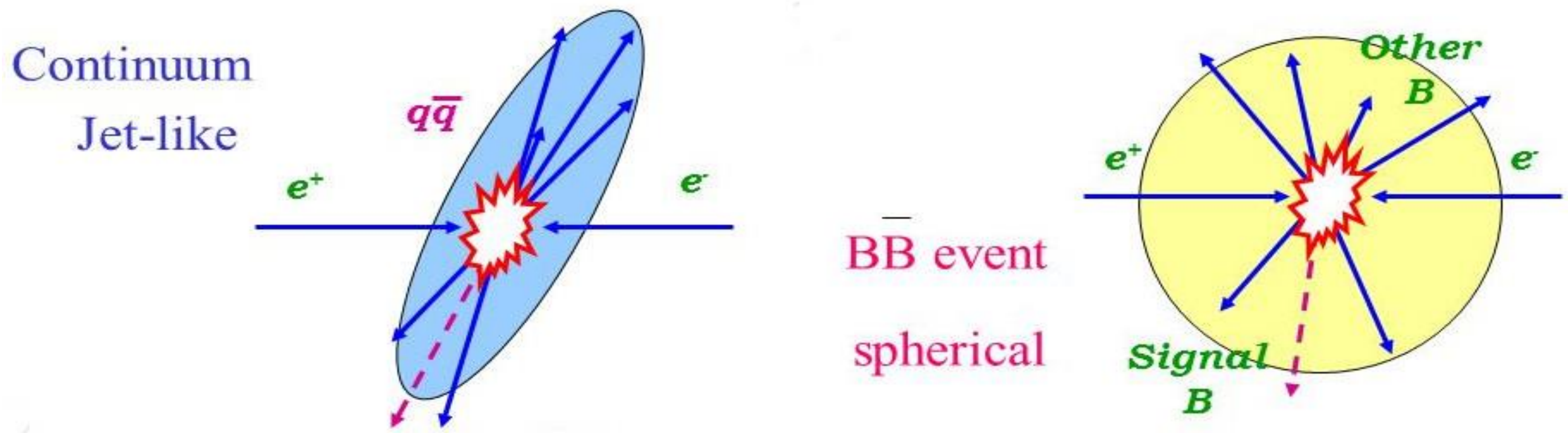
$$N_{continuum} = 400 \times N_{signal}$$

M_{bc} peaks at B mass for fully reconstructed signal
ΔE peaks at zero for fully reconstructed signal

Continuum Background

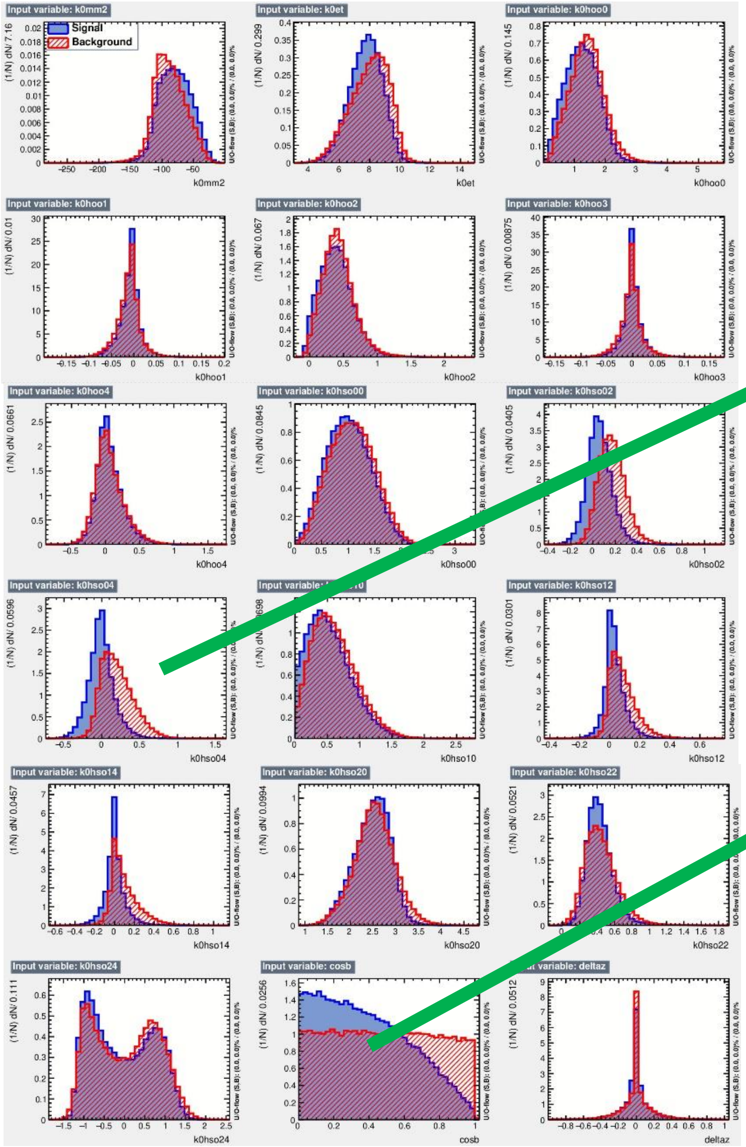
- Continuum background $e^+e^- \rightarrow q\bar{q}(u,d,s,c)$
  - Dominant background
  - Event topology differs from BB decays

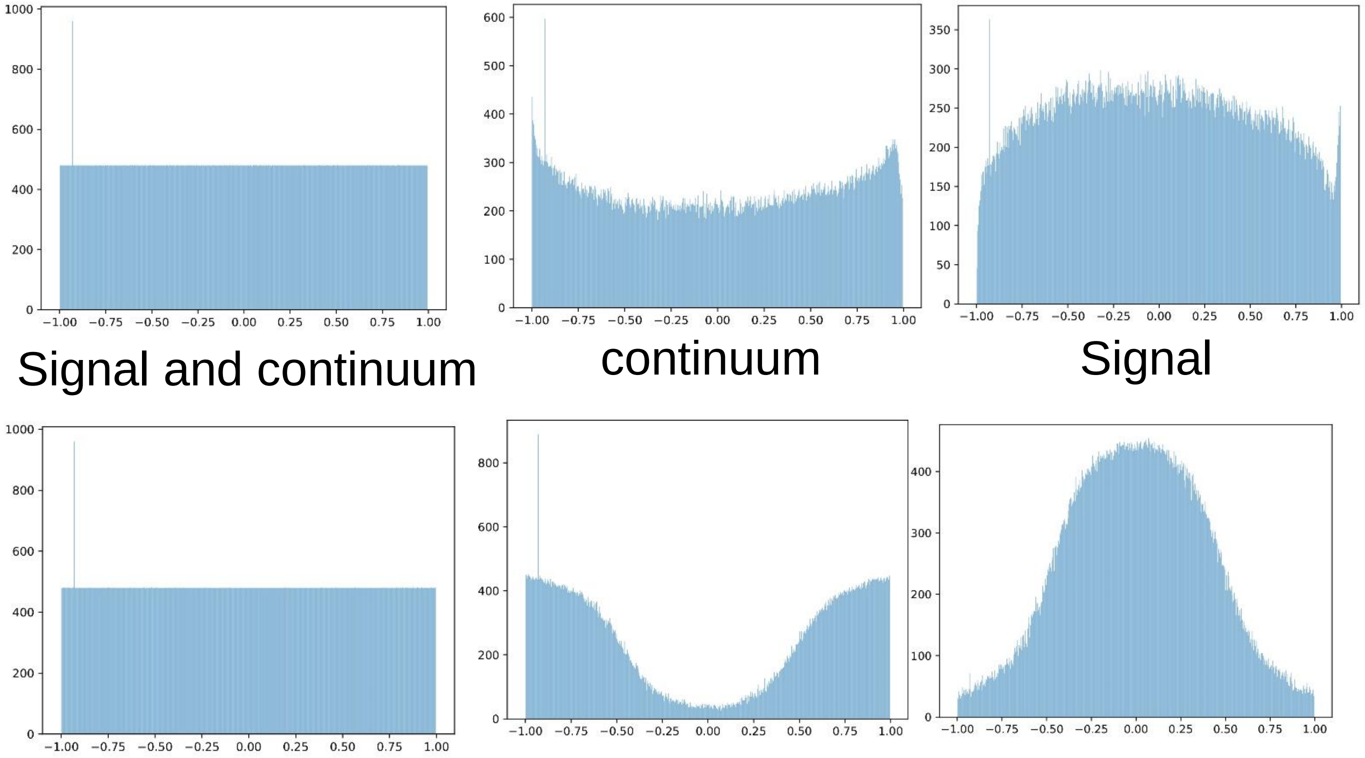




  - Combine variables describing the event topology in a Multi-Variate analysis.
  - Investigate a Deep Neural Net for improved performance

Normalization of continuum fighting variables





Signal and continuum

continuum

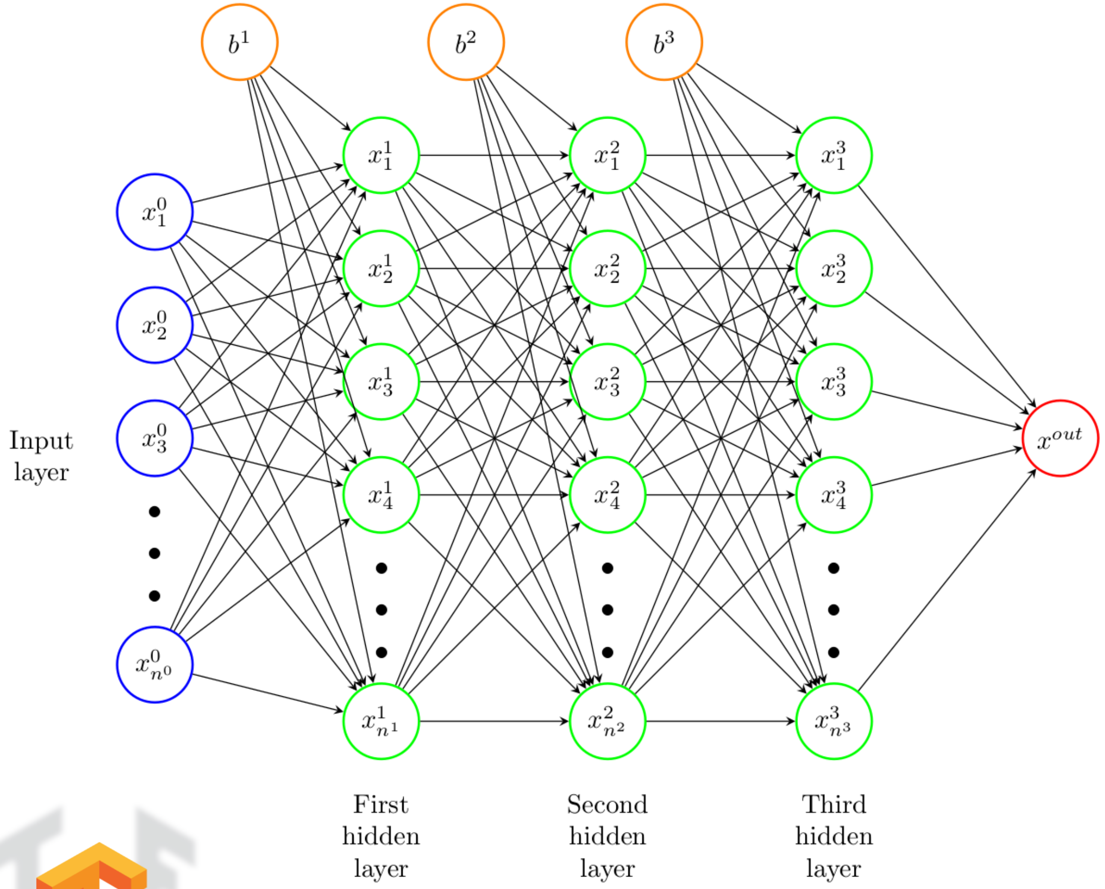
Signal

Signal

continuum

Equal frequency binning used to map into range -1.0 to +1.0

Implementation of Deep Neural Network (TensorFlow)

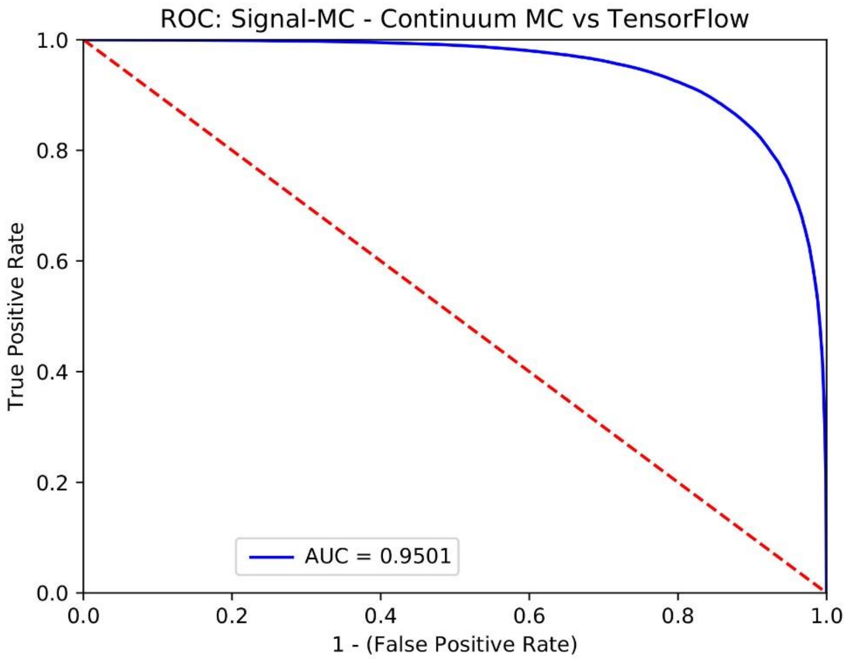


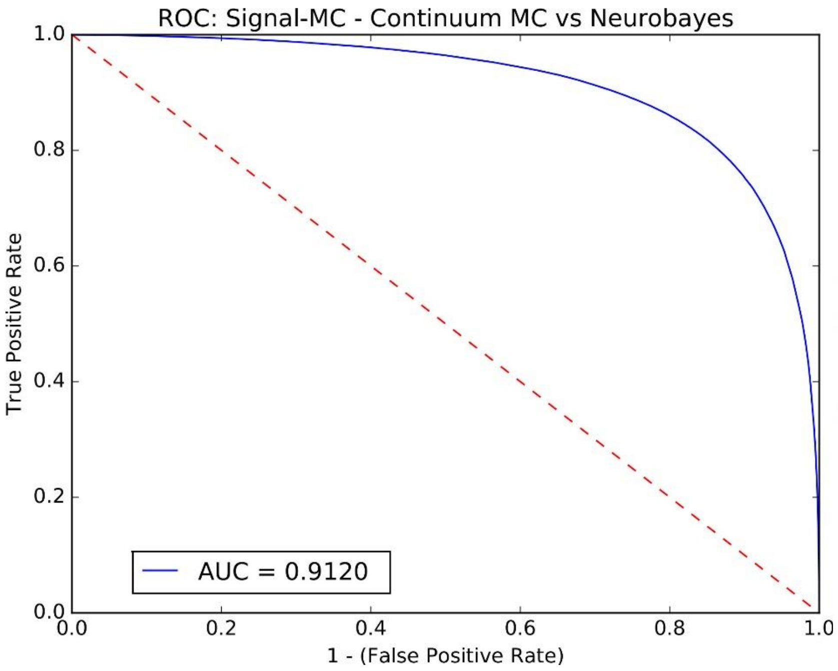
- Deep Neural Net built from the ground up in TensorFlow
- Employed Hyperband to search for best hyper-parameters

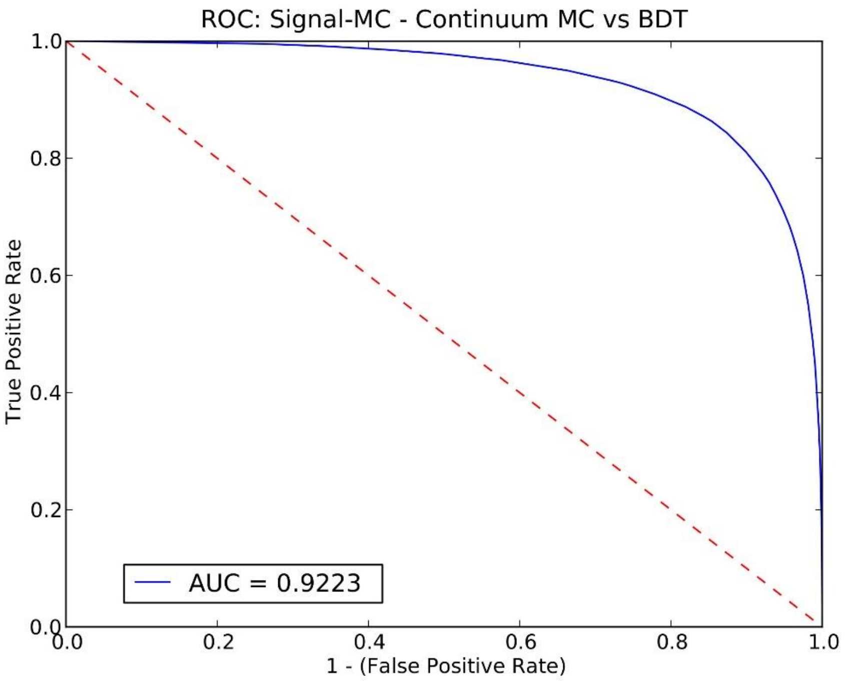
- Trained with 125000 signal and continuum events
- Validated with 125000 signal and continuum events
- Tested with 62500 signal and continuum events
- Employed ADAM algorithm for training
- $L_{class}(\vec{x}, \hat{y}) = -\hat{y} \cdot \log\big(y(\vec{x})\big) - (1 - \hat{y}) \cdot \log(1 - y(\vec{x}))$

- A maximum number of epochs 600.
- 50 events per batch
- Learning rate of 0.0001.
- Six hidden layers.
- 47 nodes per hidden layer.
- Exponential linear unit activation function.

Performance of Deep Neural Network





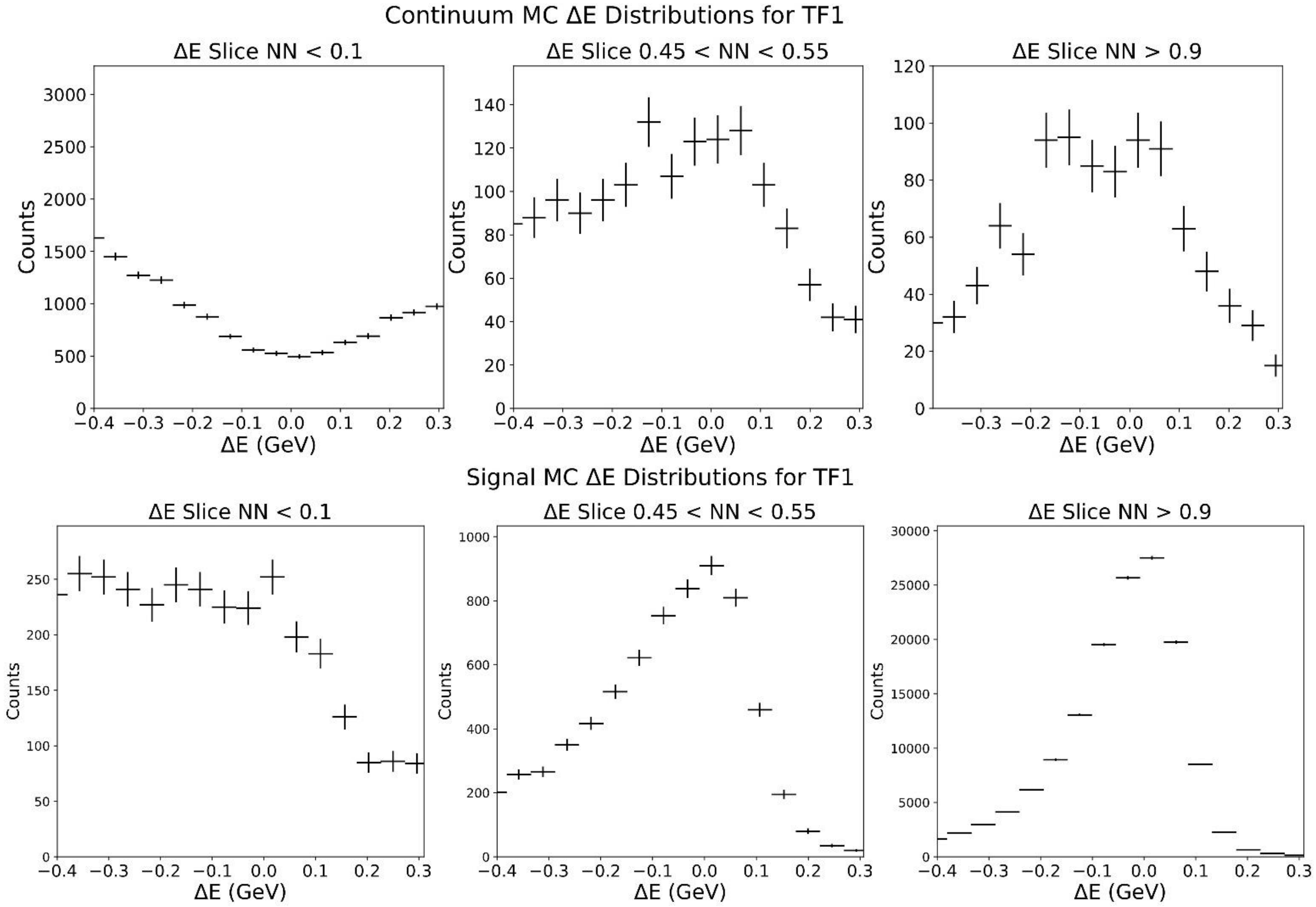


TensorFlow
Deep Neural Network
AUC = 0.9501
70% sig. eff. => 3.3% Continuum

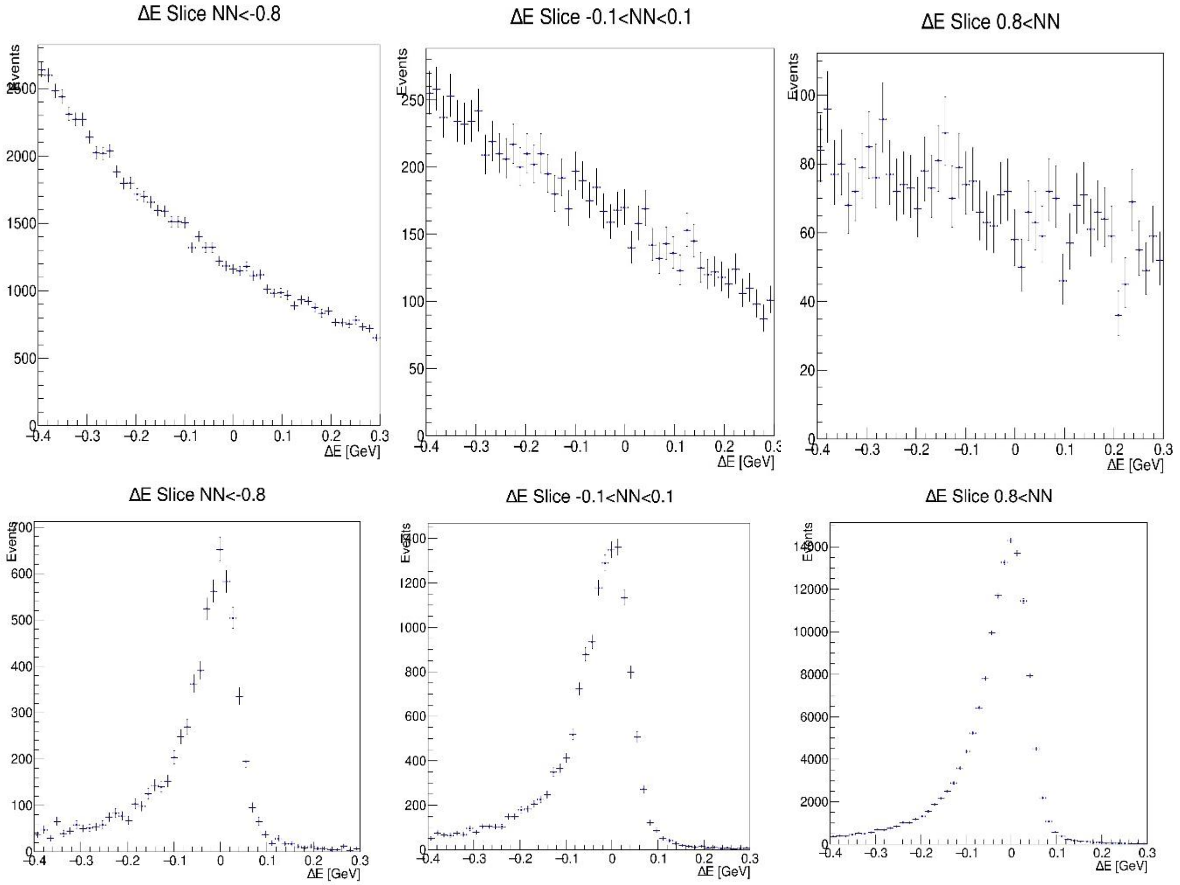
Neurobayes
Shallow Neural Network
AUC = 0.912
70% sig. eff => 7.6% continuum

TMVA
Boosted Decision Trees
AUC = 0.922
70% sig. eff => 7.0% continuum

Sculpting in ΔE distribution

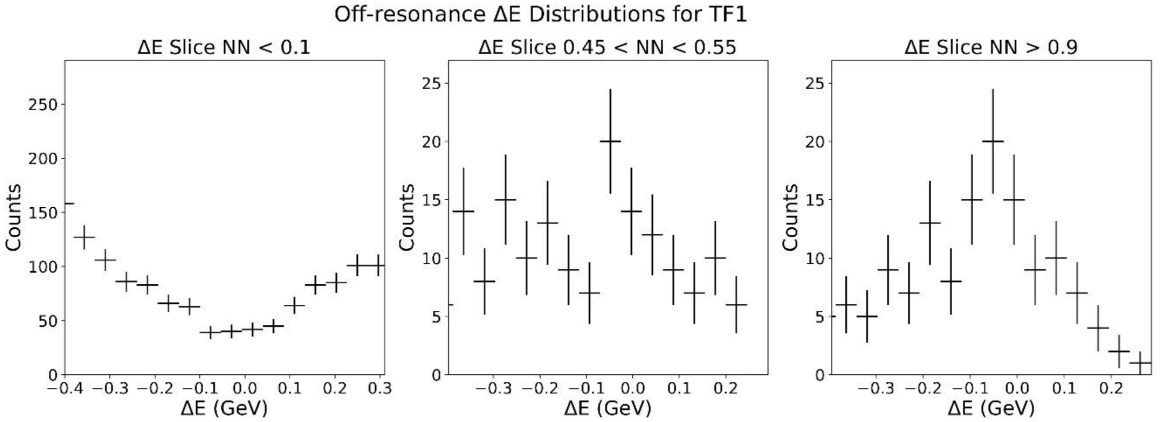


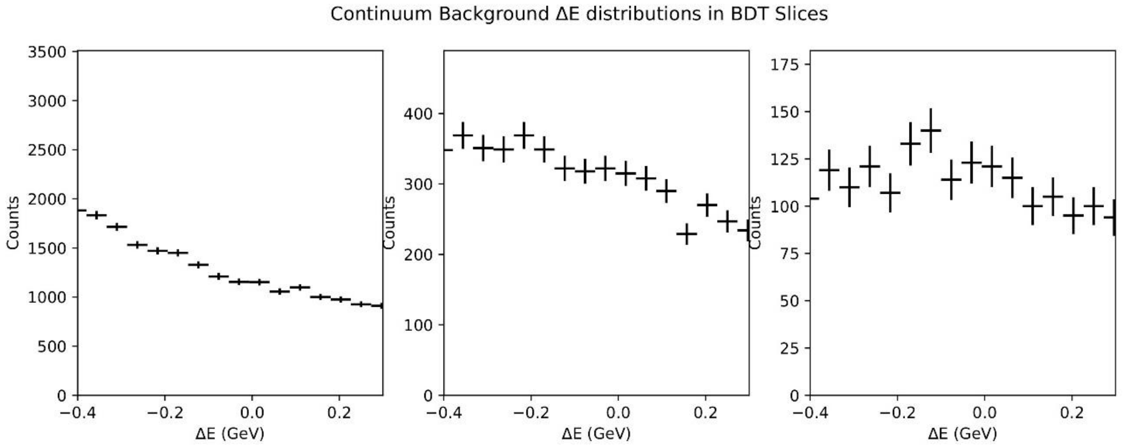
Deep Neural Net

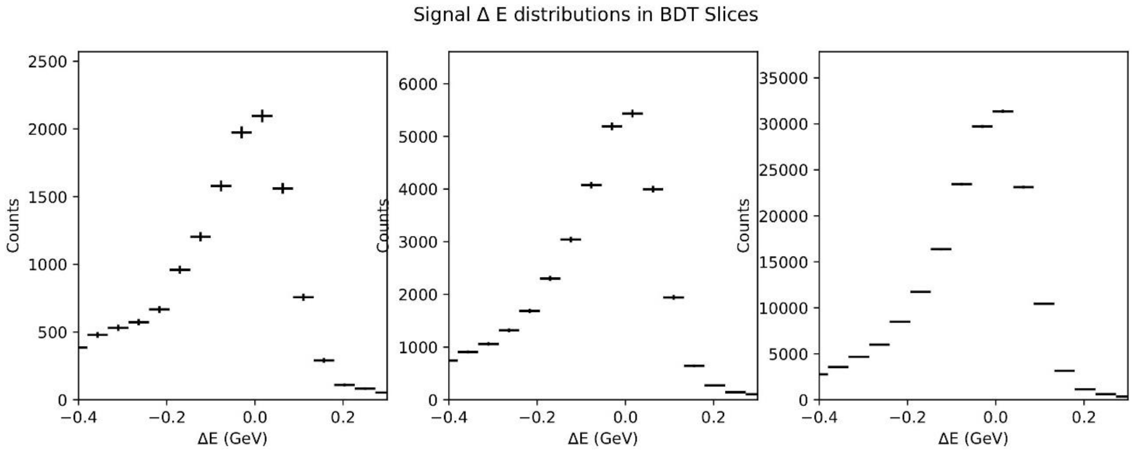


NeuroBayes

Sculpting in ΔE distribution



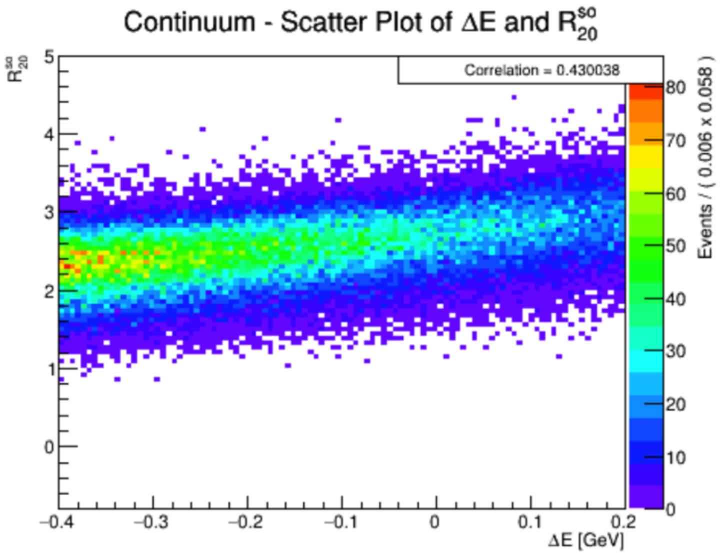




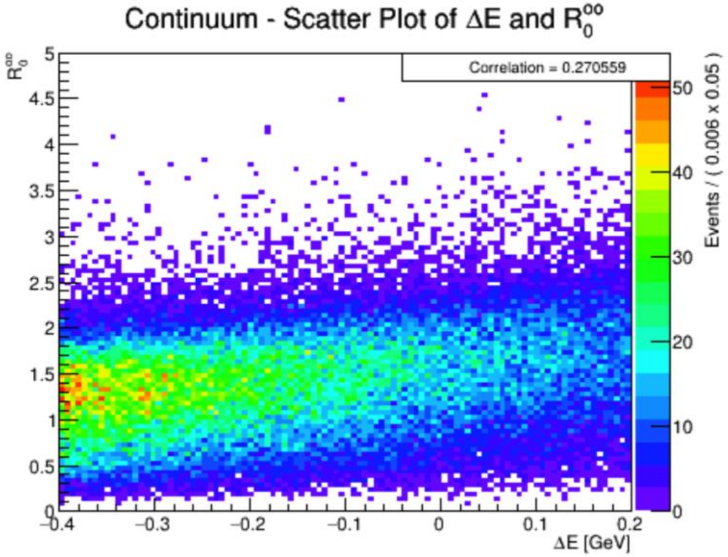
Off-resonance real-data
Deep Neural Net

Boosted Decision Trees

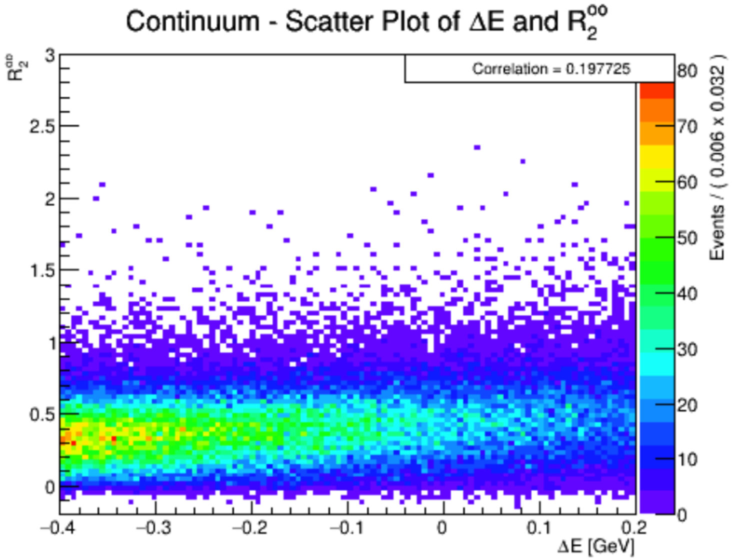
Correlations between KSFW and ΔE



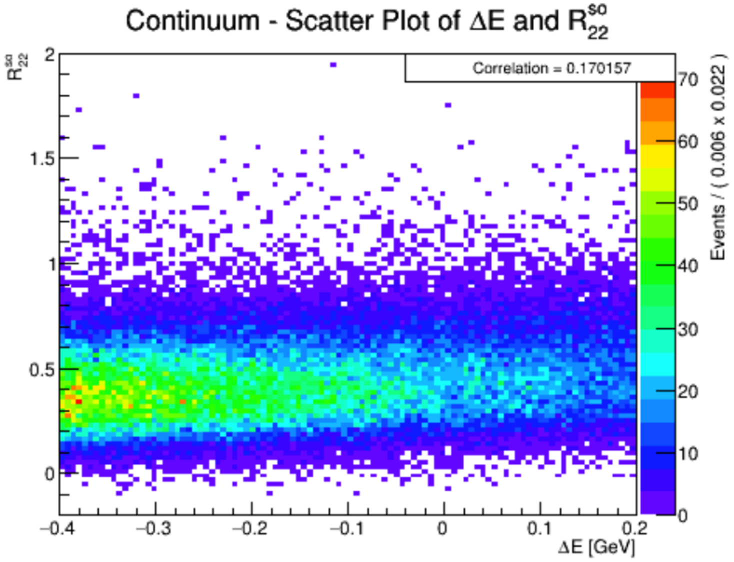
Continuum - Scatter Plot of ΔE and R_{20}^{so}



Continuum - Scatter Plot of ΔE and R_0^{oo}



Continuum - Scatter Plot of ΔE and R_2^{oo}



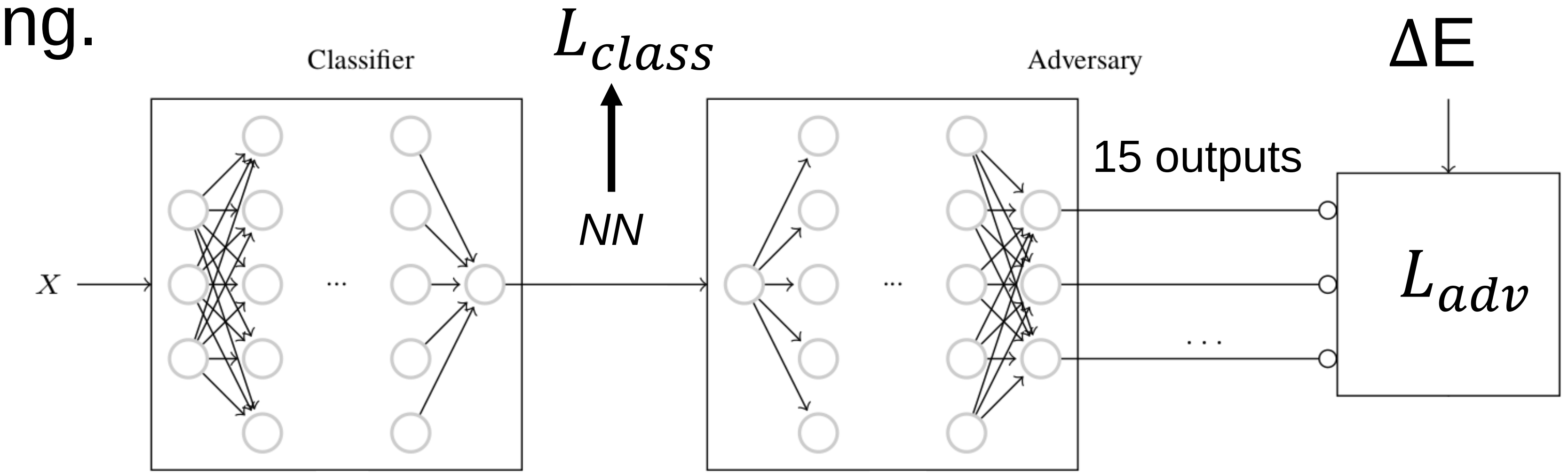
Continuum - Scatter Plot of ΔE and R_{22}^{so}

Classifier	AUC	Correl.
Deep Net All	0.950	0.114
Deep Net 1 removed	0.938	0.073
Deep Net 2 removed	0.928	0.083
Deep Net 3 removed	0.923	0.056
Deep Net 4 removed	0.918	0.062
NeuroBayes All	0.912	0.058
NeuroBayes (reduced)	0.902	-0.001
BDT All	0.922	0.262
BDT (reduced)	0.913	0.054

“reduced” means all four correlated variables removed

Adversarial Neural Network

Build an Adversarial Neural Net to keep the correlated variables but remove the sculpting.



$$L_{adv}(NN,\Delta E) = -\log\left(\sum_{i=1}^{5}\frac{f_i'(NN)}{\sqrt{2\pi\sigma_i^2(NN)}}e^{\frac{-(\mu_i-\Delta E)^2}{2\pi\sigma_i^2(NN)}}\right)$$

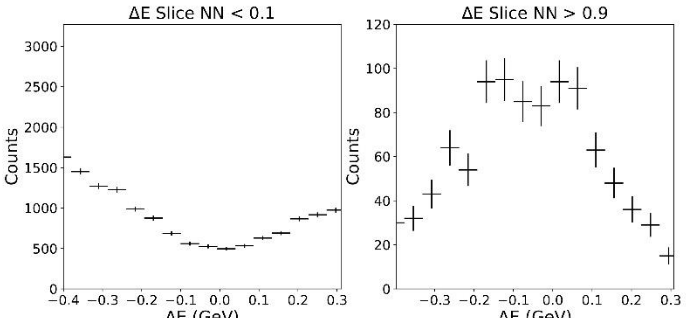
$$L_{tot} = L_{class} - \lambda_{adv}L_{adv}$$

Learning to Pivot with Adversarial Networks. Gilles Louppe, Michael Kagan, and Kyle Cranmer. (arXiv:1611.01046)

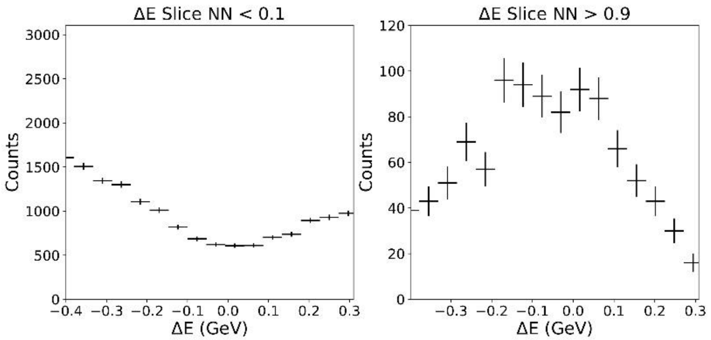
Performance of Adversarial Neural Network

Adjust λ_{adv}

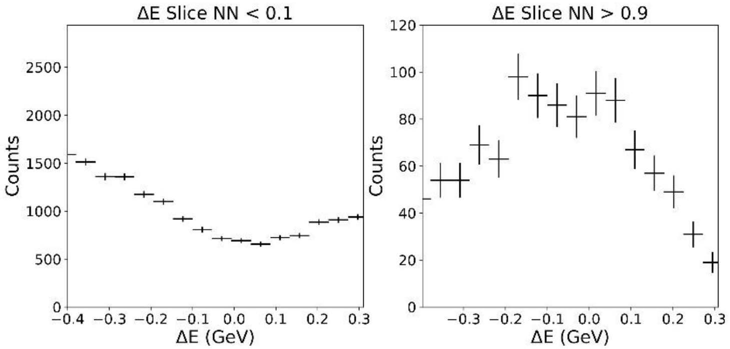
ΔE Distributions for λ_{adv} = 0.0



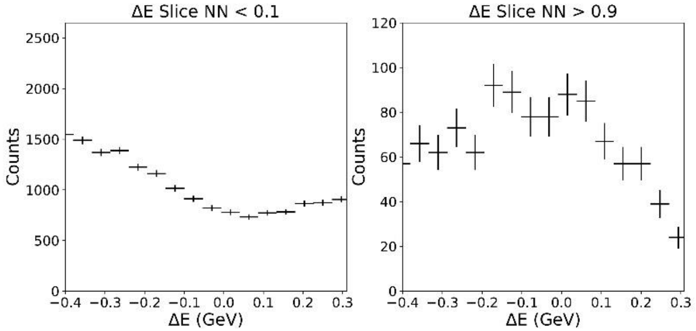
ΔE Distributions for λ_{adv} = 0.25



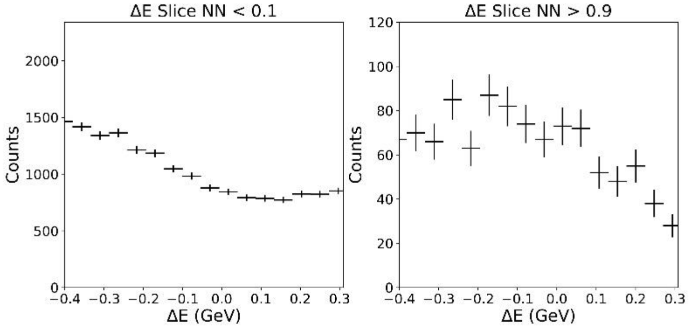
ΔE Distributions for λ_{adv} = 0.5



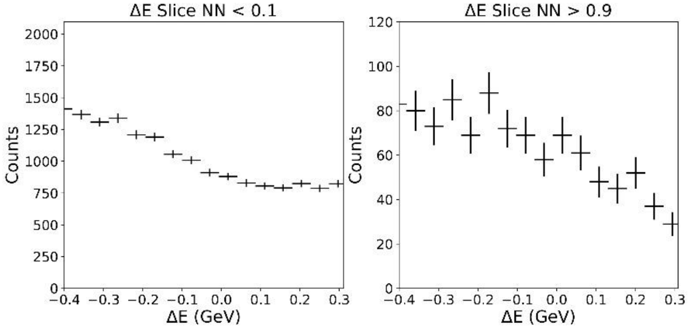
ΔE Distributions for λ_{adv} = 1.0



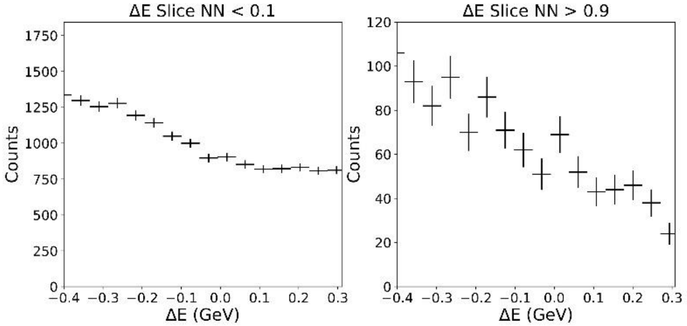
ΔE Distributions for λ_{adv} = 1.5



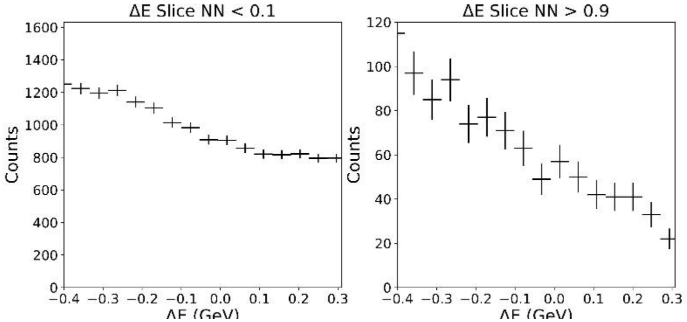
ΔE Distributions for λ_{adv} = 2.0



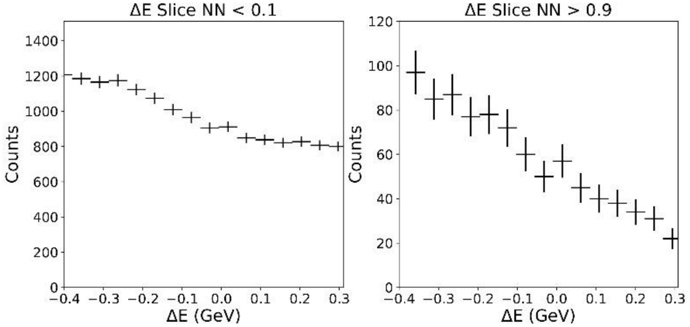
ΔE Distributions for λ_{adv} = 3.0



ΔE Distributions for λ_{adv} = 4.0



ΔE Distributions for λ_{adv} = 5.0



Performance of Adversarial Neural Network

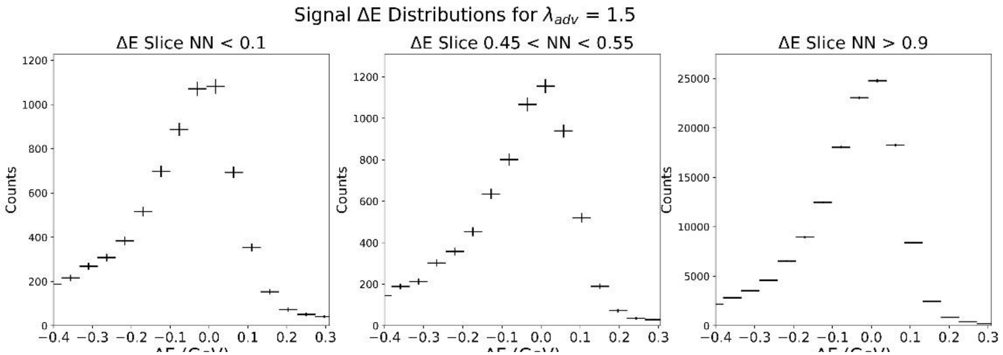
Background rejection for 92.5% signal acceptance in full and signal ($-0.1\ GeV < \Delta E < 0.1\ GeV$) ΔE regions

λ_{adv}	Bck. Rej. Full-ΔE	Bck. Rej Signal-ΔE	Corr with ΔE
0.0	81.3%	63.2%	0.114
0.25	80.6%	65.1%	0.080
0.25	79.7%	66.6%	0.057
0.75	80.0%	67.6%	0.025
1.0	78.4%	67.6%	0.012
1.5	74.6%	67.4%	-0.025
2.0	70.2%	66.6%	-0.057
3.0	64.3%	63.0%	-0.106
4.0	57.1%	57.6%	-0.145
5.0	51.9%	53.4%	-0.175
NeuroBayes	66.7%	64.1%	0.058
NeuroBayes (Reduced)	63.0%	64.1%	-0.001
BDT	73.7%	67.2%	0.262
BDT (Reduced)	66.9%	66.5%	0.054

Working point

“reduced” means all four correlated variables removed

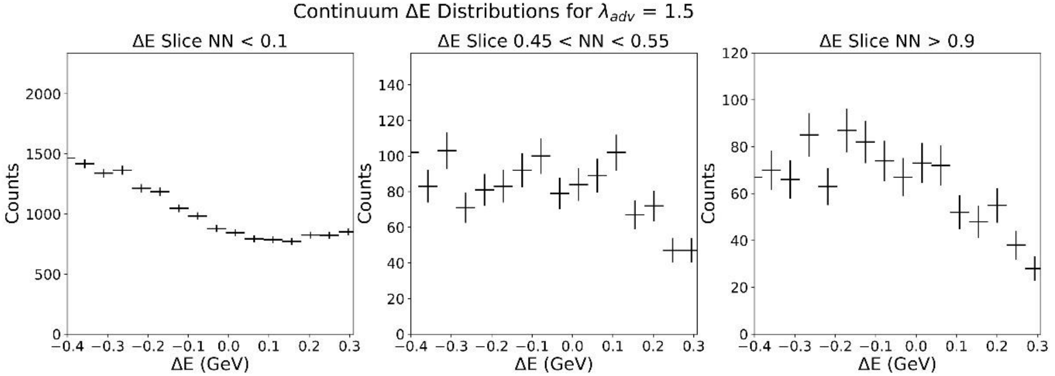
Adversarial Neural Network $\lambda_{adv} = 1.5$



Signal ΔE Distributions for λ_{adv} = 1.5

ΔE Slice NN < 0.1 — ΔE Slice 0.45 < NN < 0.55 — ΔE Slice NN > 0.9

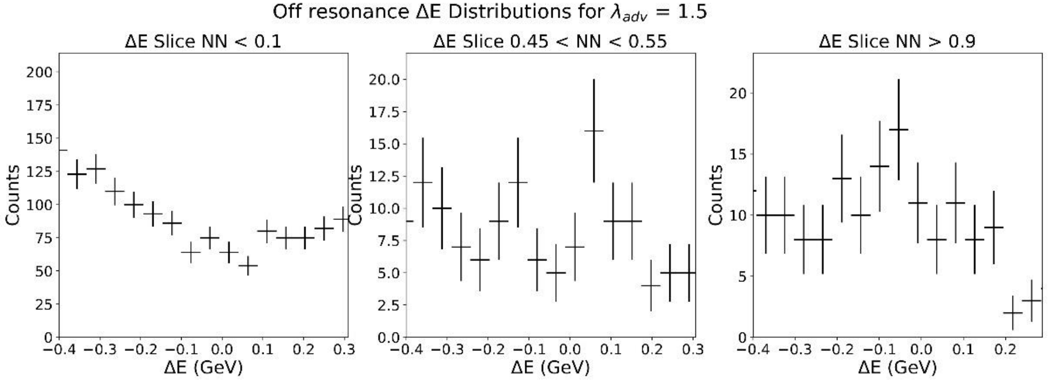
Signal, $\lambda_{adv} = 1.5$



Continuum ΔE Distributions for λ_{adv} = 1.5

ΔE Slice NN < 0.1 — ΔE Slice 0.45 < NN < 0.55 — ΔE Slice NN > 0.9

Continuum MC, $\lambda_{adv} = 1.5$



Off resonance ΔE Distributions for λ_{adv} = 1.5

ΔE Slice NN < 0.1 — ΔE Slice 0.45 < NN < 0.55 — ΔE Slice NN > 0.9

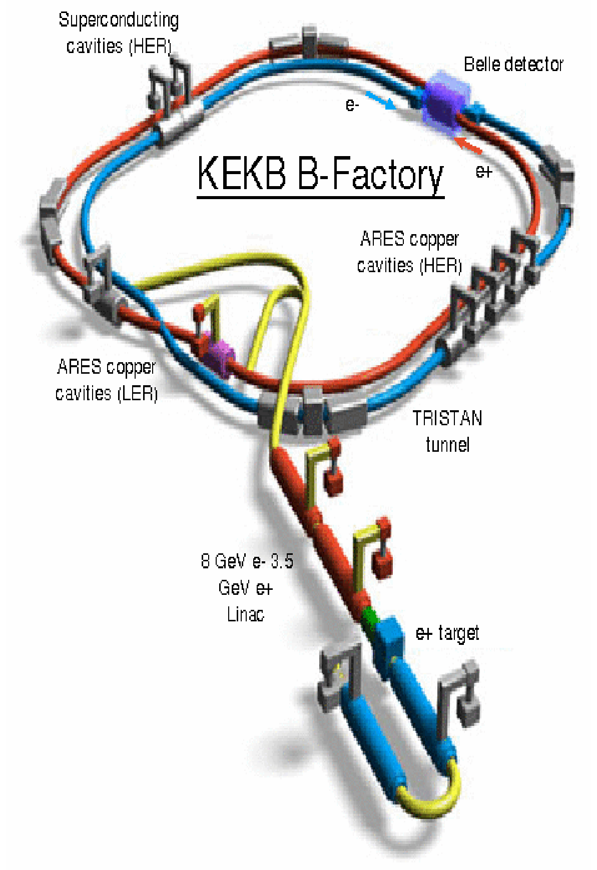
Off-resonance, $\lambda_{adv} = 1.5$

Conclusions

- Deep Neural Nets discovered and exploited a subtle correlation among the KSFW moments
- This sculpted the background ΔE distribution to resemble signal
- Removing the correlated discriminating variables reduces the effectiveness of the classification
- Built an adversarial neural net to counter-act this (negative feedback)
- Hyperparameter λ_{adv} adjusts the strength of the negative feedback
- Increasing λ_{adv} decreases the sculpting and correlation with ΔE
- Too large λ_{adv} causes negative correlation with ΔE
- In optimal region the Deep Net with λ_{adv} = 1.5 has the best background rejection with smallest correlation
- Still some sculpting λ_{adv} = 1.5 but ΔE continuum distribution still significantly different from Signal
- Off-resonance data validates the MC distributions predicted with λ_{adv} = 1.5

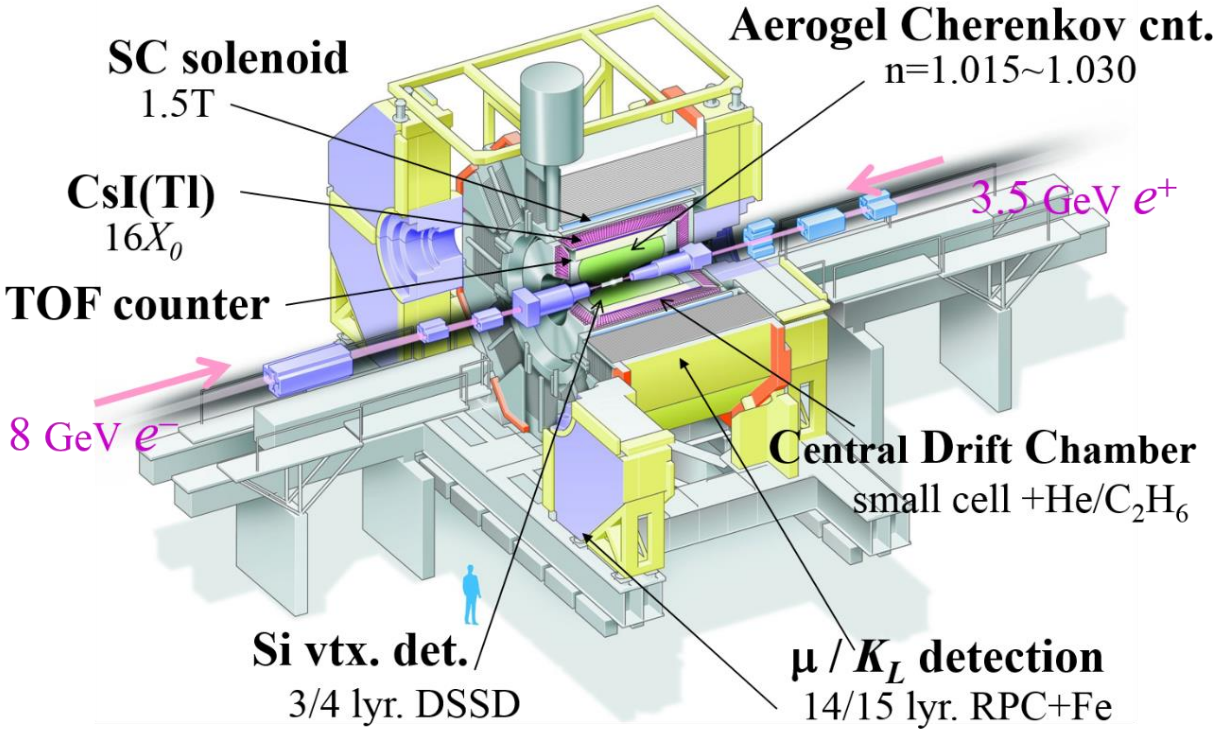
Backup

KEKB and Belle

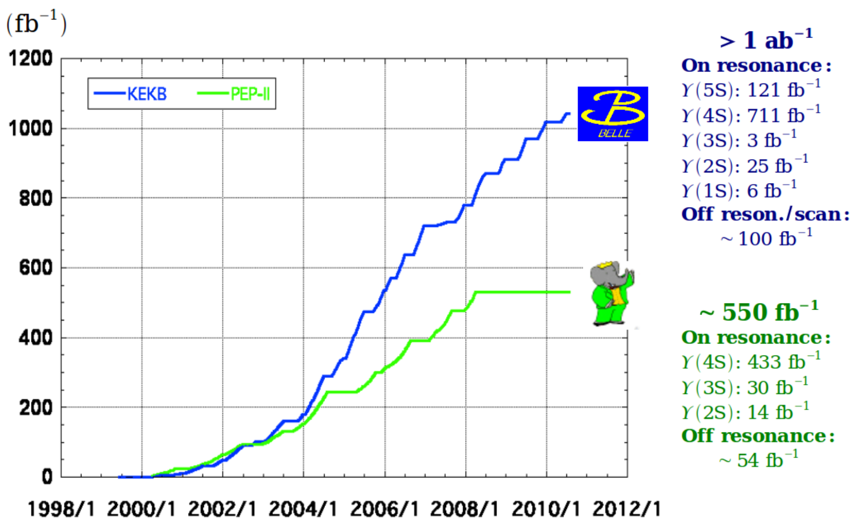

Superconducting cavities (HER)
Belle detector
e-
e+
KEKB B-Factory
ARES copper cavities (HER)
ARES copper cavities (LER)
TRISTAN tunnel
8 GeV e- 3.5 GeV e+ Linac
e+ target

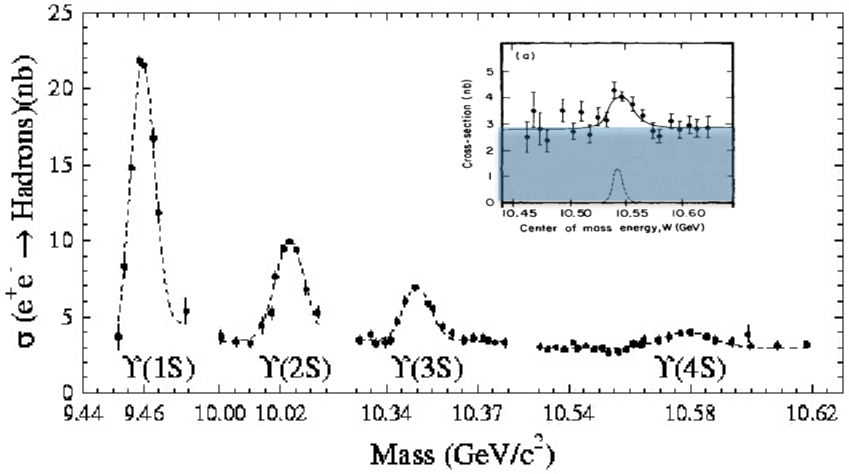
KEKB maximum Luminosity 2.1x10^{34}cm^{2}s^{-1} => 21 B-pairs/sec
SuperKEKB → 8x10^{35}cm^{2}s^{-1} => 800 B-pairs/sec (Currently 2.1x10^{33} cm^{2}s^{-1})

The Belle experiment



Integrated luminosity of B factories





Hyperparameters of Adversarial Neural Network

- 100 training steps.
- 125 events per batch.
- A Learning rate of 0.01.
- Two hidden layers.
- 20 nodes per hidden layer.
- Exponential linear unit activation function for the nodes in the hidden layer.
- 15 output nodes (three output nodes corresponding to each Gaussian):

  – 5 output nodes corresponding to μ_i - no activation function (identity operator).
  – 5 output nodes corresponding to un-normalised fractions f_i - no activation function (identity operator).
  – 5 output nodes corresponding to σ_i, where the ‘activation’ is the exponential function, to ensure that the widths of the Gaussians are positive.

Training the Adversarial Neural Network

1. Train the NN to optimally separate signal and continuum. (TF1)
2. Create the ANN, and the classifying (the original) NN with the same architecture
3. For every 20,000 steps and a given choice of λ_{adv} :

   (a) Train the ANN for the given number of adversary training steps (100 steps), where

      (i) For every event in the batch, get the NN output from the classifier.

      (ii) Using NN and ΔE get the adversarial loss given by L_{class} .

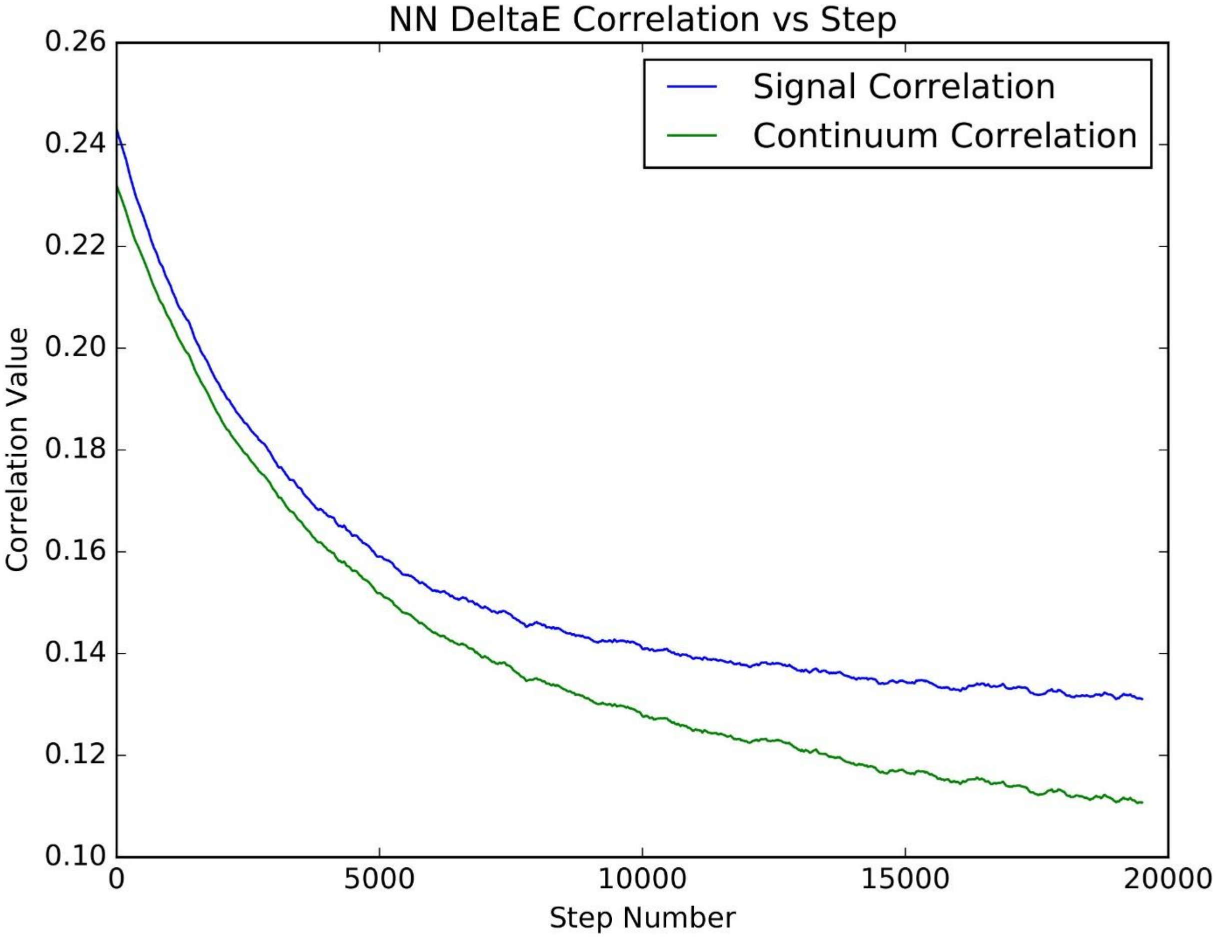
      (iii) Train the ANN given the adversarial loss and adversarial learning rate

   (b) Train the classifier for one training step, with the loss function given by L_{tot} (dependence on ΔE, as well as NN)

4. This is the ANN-corrected Neural Net

Adversarial Neural Network

Correlation as training proceeds for λ_{adv}=0.5



- 4 epochs, of 5000 classifier-training steps
- adversarial network is trained for 125 steps per classifier training step.
- correlations are in the validation data sets, and calculated over the entire range 0 < NN < 1.