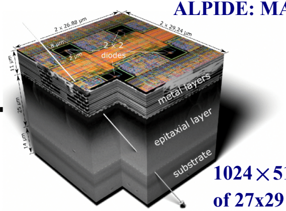
Application of Bayesian Neural Networks to clustering for LUXE experiment

Roman Urmanov

05.06.2023

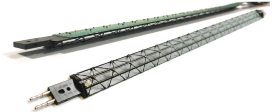
Introduction: Tracker

ALPIDE: MAP Sensor



1024 × 512 matrix of 27x29 μm^2 pixels

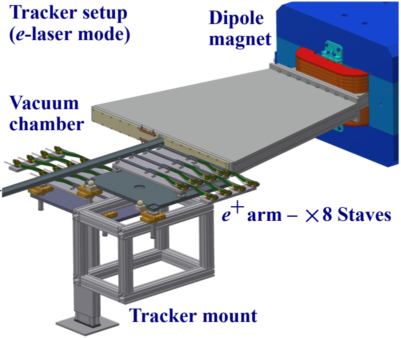
Stave – × 9 ALPIDEs



Tracker setup (*e*-laser mode)

Dipole magnet

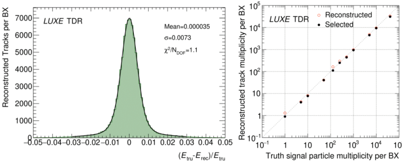
Vacuum chamber



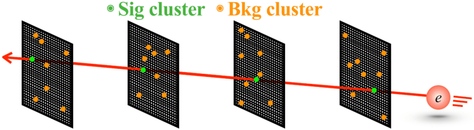
e^+ arm – × 8 Staves

Tracker mount

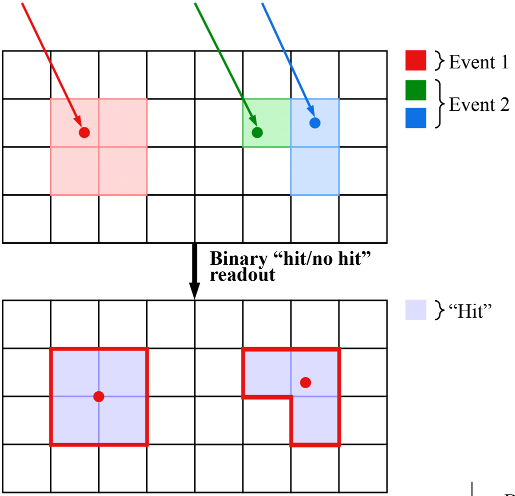
Physics analysis



Track reconstruction



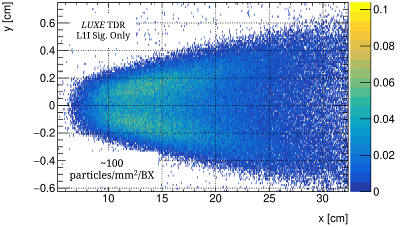
Clustering: ML motivation



"Pac-Man": adjacent hits combined into clusters. Cluster's "center of mass" is taken for track inference

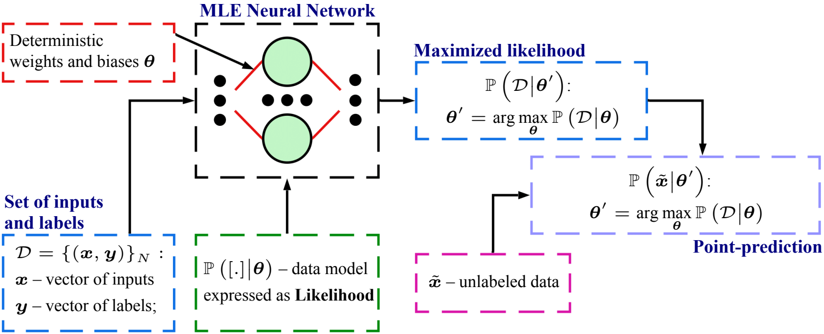
- Binary output – no information on number of particles
- LUXE will work in a wide range of pixel occupancies
- "Pac-Man" – up to $\lesssim 10^4$ e per BX
- Up to 3×10^5 e per BX in later stages

Hit density at ξ=7



Detector reference	Hit density [mm^{-2}]		
	MCD	ATLAS ITk	ALICE ITS3
Pixel Layer 0	3.68	0.643	0.85
Pixel Layer 1	0.51	0.022	0.51

Bayesian Neural Network: General structure



MLE Neural Network

Deterministic
weights and biases $\boldsymbol{\theta}$

Maximized likelihood

$\mathbb{P}\left(\mathcal{D}|\boldsymbol{\theta}'\right)$:

$\boldsymbol{\theta}' = \arg\max_{\boldsymbol{\theta}} \mathbb{P}\left(\mathcal{D}|\boldsymbol{\theta}\right)$

$\mathbb{P}\left(\tilde{\boldsymbol{x}}|\boldsymbol{\theta}'\right)$:

$\boldsymbol{\theta}' = \arg\max_{\boldsymbol{\theta}} \mathbb{P}\left(\mathcal{D}|\boldsymbol{\theta}\right)$

Point-prediction

Set of inputs and labels

$\mathcal{D} = \{(\boldsymbol{x}, \boldsymbol{y})\}_N$:

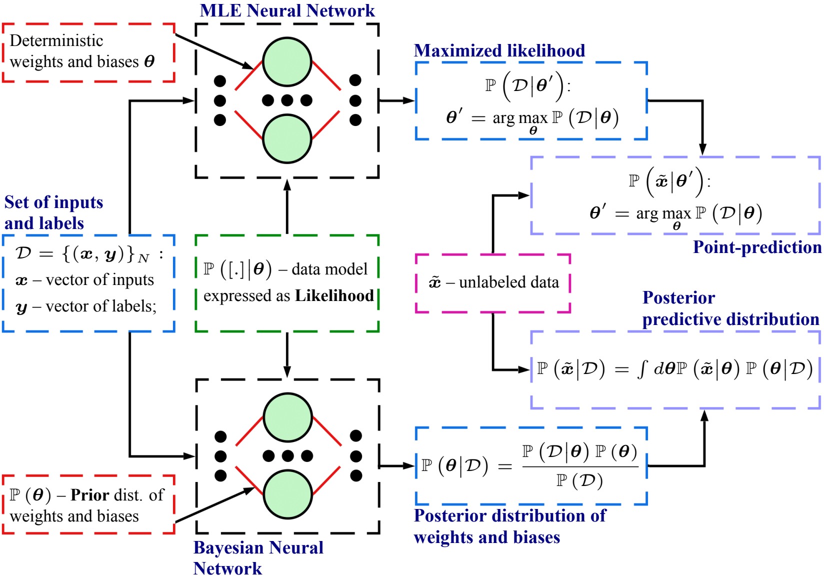
$\boldsymbol{x}$ – vector of inputs

$\boldsymbol{y}$ – vector of labels;

$\mathbb{P}\left([.]|\boldsymbol{\theta}\right)$ – data model expressed as **Likelihood**

$\tilde{\boldsymbol{x}}$ – unlabeled data

Bayesian Neural Network: General structure



MLE Neural Network

Deterministic weights and biases $\boldsymbol{\theta}$

Maximized likelihood

$\mathbb{P}\left(\mathcal{D}|\boldsymbol{\theta}'\right)$:

$\boldsymbol{\theta}' = \arg\max_{\boldsymbol{\theta}} \mathbb{P}\left(\mathcal{D}|\boldsymbol{\theta}\right)$

$\mathbb{P}\left(\tilde{\boldsymbol{x}}|\boldsymbol{\theta}'\right)$:

$\boldsymbol{\theta}' = \arg\max_{\boldsymbol{\theta}} \mathbb{P}\left(\mathcal{D}|\boldsymbol{\theta}\right)$

Point-prediction

Set of inputs and labels

$\mathcal{D} = \{(\boldsymbol{x}, \boldsymbol{y})\}_N$:

$\boldsymbol{x}$ – vector of inputs

$\boldsymbol{y}$ – vector of labels;

$\mathbb{P}\left([.]|\boldsymbol{\theta}\right)$ – data model expressed as **Likelihood**

$\tilde{\boldsymbol{x}}$ – unlabeled data

Posterior predictive distribution

$\mathbb{P}\left(\tilde{\boldsymbol{x}}|\mathcal{D}\right) = \int d\boldsymbol{\theta} \mathbb{P}\left(\tilde{\boldsymbol{x}}|\boldsymbol{\theta}\right) \mathbb{P}\left(\boldsymbol{\theta}|\mathcal{D}\right)$

$\mathbb{P}\left(\boldsymbol{\theta}\right)$ – **Prior** dist. of weights and biases

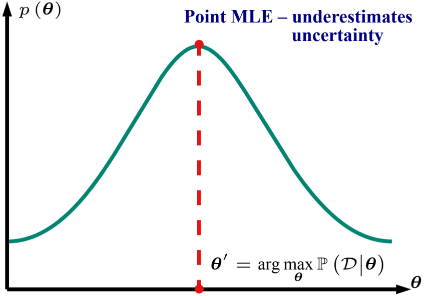
$\mathbb{P}\left(\boldsymbol{\theta}|\mathcal{D}\right) = \dfrac{\mathbb{P}\left(\mathcal{D}|\boldsymbol{\theta}\right) \mathbb{P}\left(\boldsymbol{\theta}\right)}{\mathbb{P}\left(\mathcal{D}\right)}$

Posterior distribution of weights and biases

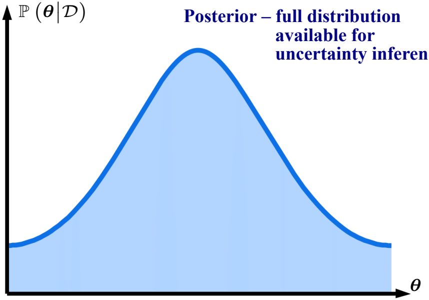
Bayesian Neural Network

- Bayesian Neural Network – Point-Prediction network with stochastic weights and biases

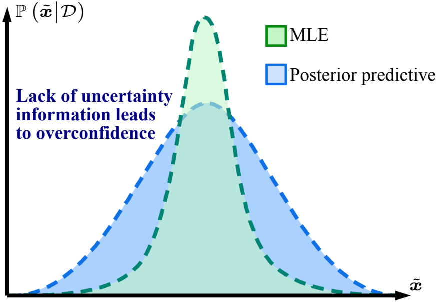
Bayesian Neural Networks: Motivation



Point MLE – underestimates uncertainty



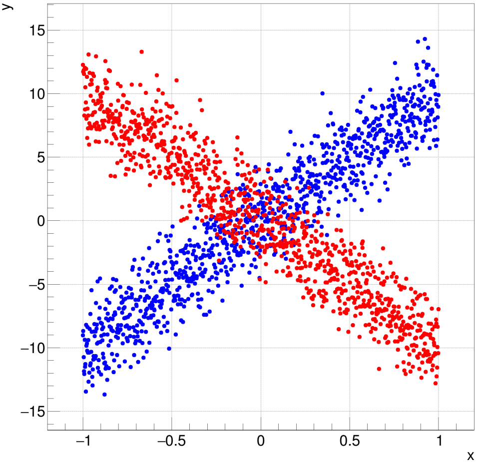
Posterior – full distribution available for uncertainty inference



Lack of uncertainty information leads to overconfidence

Predictive distributions for the unlabeled data

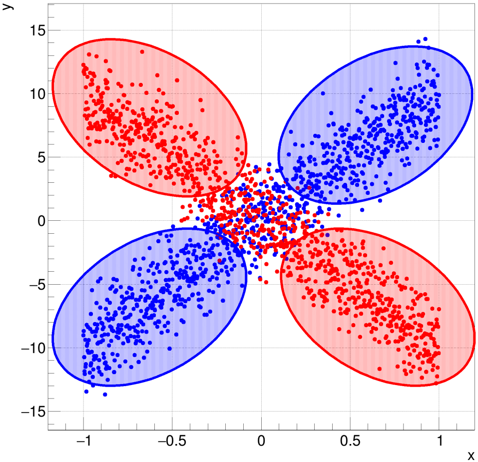
Bayesian Neural Networks: Motivation



Example of a classification problem with a significant class overlap

- Classification task: based on the (x, y) position assign one of the two classes to an event

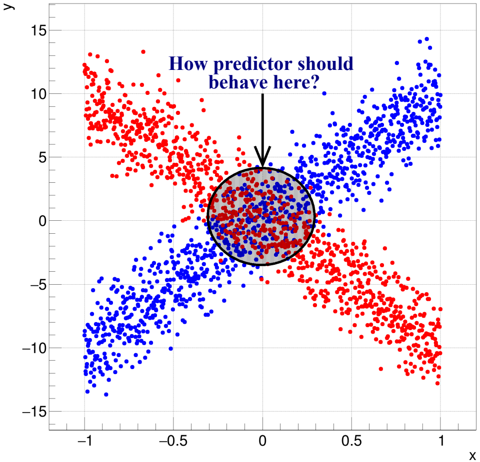
Bayesian Neural Networks: Motivation



Example of a classification problem with a significant class overlap

- Classification task: based on the (x, y) position assign one of the two classes to an event
- It is clear what we want from a predictor in regions with no overlap

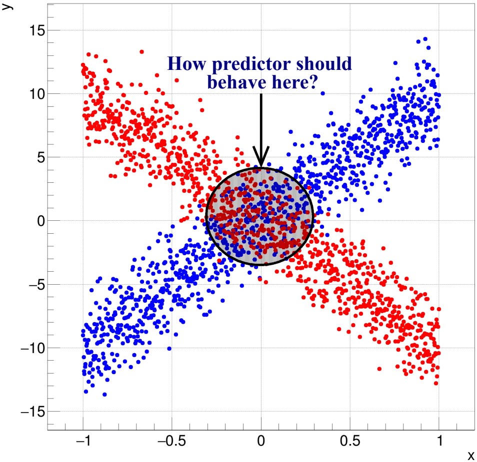
Bayesian Neural Networks: Motivation



Example of a classification problem with a significant class overlap

- Classification task: based on the (x, y) position assign one of the two classes to an event
- It is clear what we want from a predictor in regions with no overlap
- What should the predictor do in the overlap region?
- If no more information is available about the data, one can as well believe whatever predictor infers

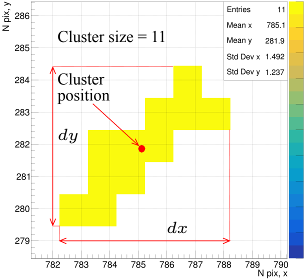
Bayesian Neural Networks: Motivation



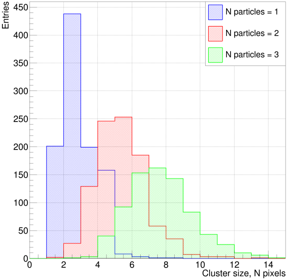
Example of a classification problem with a significant class overlap

- Classification task: based on the (x, y) position assign one of the two classes to an event
- It is clear what we want from a predictor in regions with no overlap
- What should the predictor do in the overlap region?
- If no more information is available about the data, one can as well believe whatever predictor infers
- If know about the data more than we're able to supply to the predictor uncertainty estimation becomes meaningful

Bayesian Neural Networks: Motivation

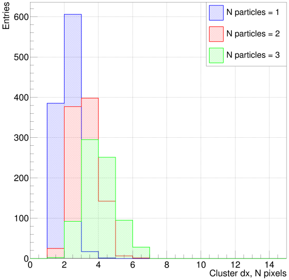


Generated cluster

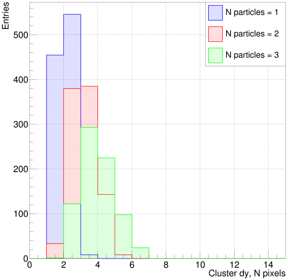


Cluster size distribution

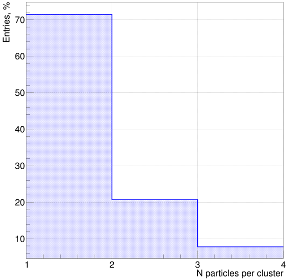
- Classification task: infer the number of particles in a cluster (out of 3)
- Significant paramteric overlap between classes
- In LUXE we expect ∼3 times more 2-particle clusters than 3-particle clusters and ∼4 times more 1-particle clusters than 2-particle clusters



Cluster dx distribution



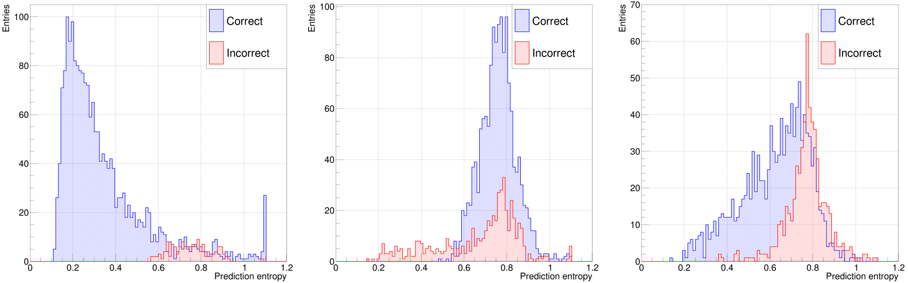
Cluster dy distribution



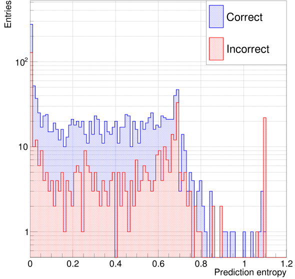
N particles per cluster distribution

Performance analysis: accuracy and confidence

Entropy distribution for 1,2 and 3 particle BNN predictions (left-to-right)



Entropy distribution for 2-particle MLE NN predictions



- Prediction: $C' = \arg\max_{C} \mathbb{P}\left(\tilde{\boldsymbol{x}} \middle| \mathcal{D}\right), C \in \{1, 2, 3\}$

- Distribution's entropy: $H(\mathbb{P}) = -\sum_{n} P_n \log\left(P_n\right)$

- $H(\mathbb{P}) = 0 \iff F = \{1, 0, 0\}$

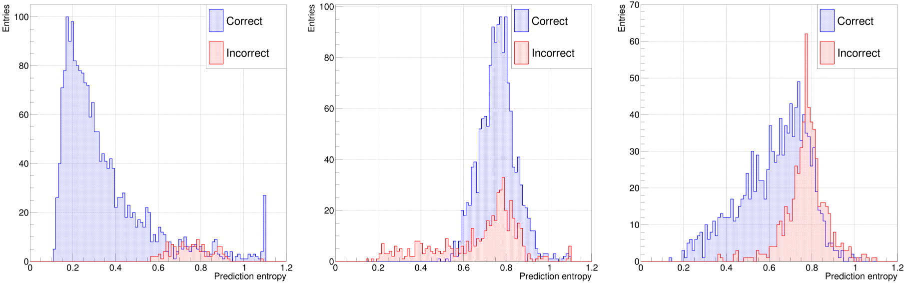
$$H(\mathbb{P}) \rightarrow \max \iff \mathbb{P} = \left\{\tfrac{1}{3}, \tfrac{1}{3}, \tfrac{1}{3}\right\}$$

Performance:

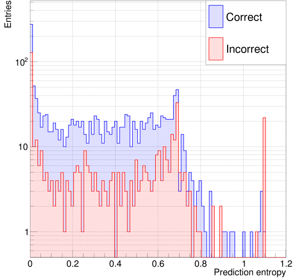
Efficiency	BNN,%	MLE NN,%
ε_1	93.41	92.69
ε_2	73.89	75.13
ε_3	67.38	66.63

Performance analysis: accuracy and confidence

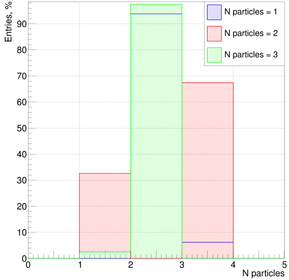
Entropy distribution for 1,2 and 3 particle BNN predictions (left-to-right)



Entropy distribution for 2-particle MLE NN predictions

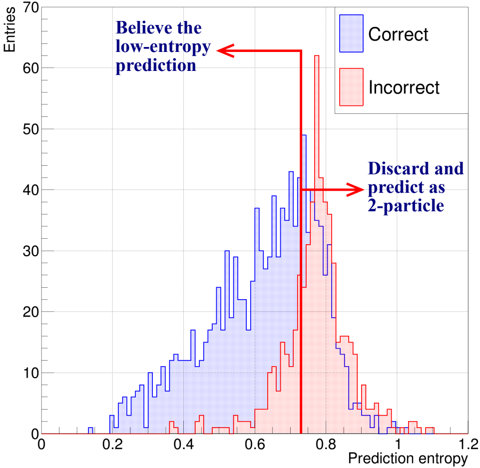


Contamination of misclassified events



- There's no clear separation in MLE NN entropy distribution
- BNN's entropy distribution allows additional analysis to be employed
- Expected distribution between classes and contamination levels can be leveraged

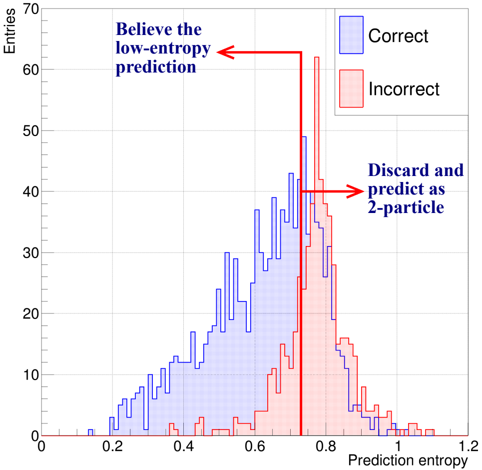
Efficiency improvement



Example of the entropy-thresholded decision making process

- Decision making procedure:
  1. Set the decision threshold on the prediction's entropy H_0
  2. If for a given 2 or 3 particle prediction $H(\mathbb{P}) < H_0$ proceed with the network's inference
  3. If for a given 2 or 3 particle prediction $H(\mathbb{P}) \geq H_0$ enforce classification as a 2-particle cluster
  4. Find the optimal H_0 by observing the efficiency ε on the validation set

Efficiency improvement



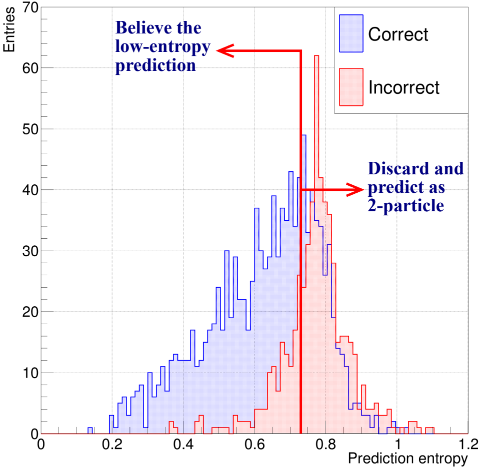
Example of the entropy-thresholded decision making process

- A separate dataset with the distribution of events between cluster classes has been prepared
- Overall efficiency ε_{tot} of 87.24% and combined efficiency for 2 and 3 particle clusters $\varepsilon_{2\cup3}$ of 74.55% have been achieved on the MLE network
- Thresholded BNN improves required computational time for tracking by ∼5-10%

BNN's efficiency for different entropy thresholds

H_0	0.65	0.68	0.71	0.74
ε_{tot}, %	90.22	**90.49**	90.33	90.07
$\varepsilon_{2\cup3}$, %	83.27	**83.76**	83.12	82.52
ε_2, %	**97.14**	95.52	93.44	90.64
ε_3, %	40.63	45.21	49.58	**54.58**

Efficiency improvement



Example of the entropy-thresholded decision making process

- Procedure is very crude – a lot of room for improvement
- Bernoulli-like decision tree can be implemented
- Distribution of above-threshold-entropy predictions can be accounted for
- More sensitive to the ε_3 criterion of optimal H_0 can be derived
- Architectures with better entropy separation can be searched for

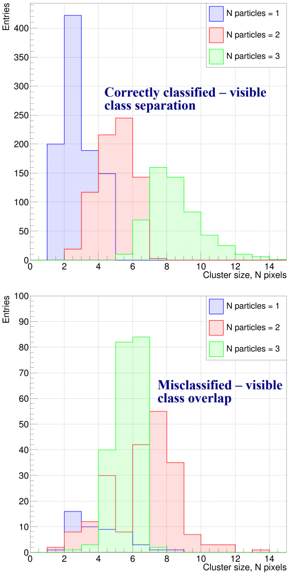
BNN's efficiency for different entropy thresholds

H_0	0.65	0.68	0.71	0.74
ε_{tot}, %	90.22	**90.49**	90.33	90.07
$\varepsilon_{2\cup 3}$, %	83.27	**83.76**	83.12	82.52
ε_2, %	**97.14**	95.52	93.44	90.64
ε_3, %	40.63	45.21	49.58	**54.58**

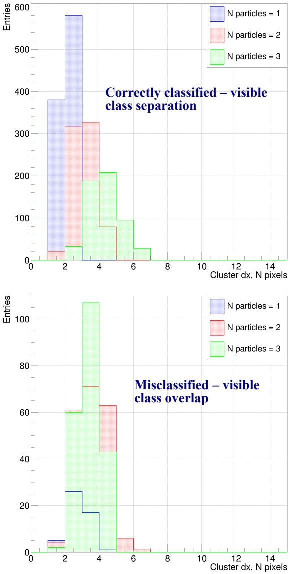
Summary

- Cluster merging causes track reconstruction issues. Clustering algorithm is required to perform over high range of occupancies
- MLE and Bayesian Neural Networks are applied to clustering
- Accuracies of $\sim$93%, $\sim$70% and $\sim$75% are achieved for 1, 2 and 3-particle events respectively for both networks
- BNN's predictions allow for meaningful uncertainty estimation
- Ad-hoc procedure of alternative decision making has shown promising performance

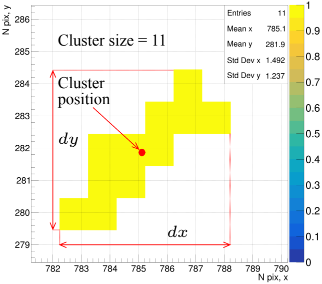
Backup: detailed observables investigation (1)



Cluster size for correct and misclassified predictions

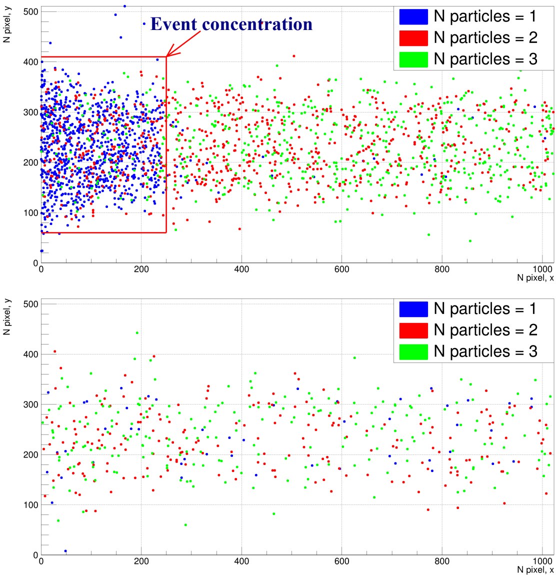


Cluster dx for correct and misclassified predictions

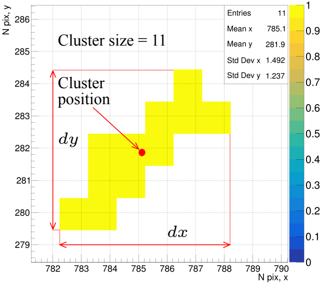


- BNN detects geometrical features
- Errors happen in regions of parametric overlap
- Better entropy separation requires architecture modifications

Backup: detailed observables investigation (2)

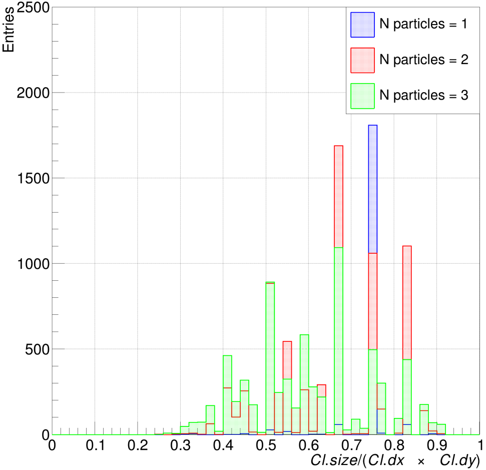


Cluster position distribution for correct (top) and misclassified (bottom) predictions



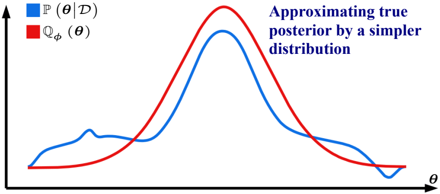
- BNN detects spatial features
- Higher multiplicity events are badly separated – additional parametrs, more complex network

Backup: detailed observables investigation (3)

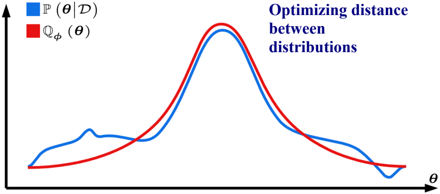


Percentage of area filled with active pixels over cluster's dimension rectangle

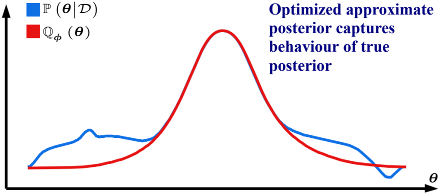
Backup: SVI



Approximating true posterior by a simpler distribution



Optimizing distance between distributions



Optimized approximate posterior captures behaviour of true posterior

- Predictions are Posterior-based. Have to infer $\mathbb{P}\left(\boldsymbol{\theta}|\mathcal{D}\right)$
- MC methods are usually inefficient due to high dimensionality
- SVI – approximate + high variance gradients

Simple approximate posterior $\mathbb{Q}_{\phi}\left(\boldsymbol{\theta}\right)$

$\Downarrow$

KL divergence:

$$D_{\mathrm{KL}}\left(\mathbb{Q}_{\phi}\left(\boldsymbol{\theta}\right)||\mathbb{P}\left(\boldsymbol{\theta}|\mathcal{D}\right)\right) = \mathbb{E}_{\boldsymbol{\theta}\sim\mathbb{Q}_{\phi}(\boldsymbol{\theta})}\left[\log\left(\frac{\mathbb{Q}_{\phi}\left(\boldsymbol{\theta}\right)}{\underset{\substack{\uparrow\\ \textbf{unknown}}}{\mathbb{P}\left(\boldsymbol{\theta}|\mathcal{D}\right)}}\right)\right]$$

$$D_{\mathrm{KL}}\left(\mathbb{Q}_{\phi}\left(\boldsymbol{\theta}\right)||\mathbb{P}\left(\boldsymbol{\theta}|\mathcal{D}\right)\right) \to \min_{\phi}$$

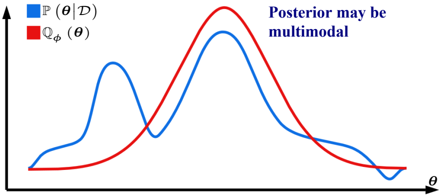
$\Downarrow$

Evidence Lower Bound (ELBO):

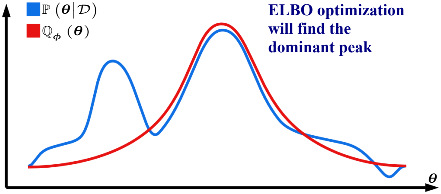
$$\mathcal{L}\left(\mathbb{Q}\right) = \mathbb{E}_{\boldsymbol{\theta}\sim\mathbb{Q}_{\phi}(\boldsymbol{\theta})}\left[\log\left(\frac{\overset{\substack{\textbf{BNN output}\\ \downarrow}}{\mathbb{P}\left(\mathcal{D}|\boldsymbol{\theta}\right)}\,\mathbb{P}\left(\boldsymbol{\theta}\right)}{\mathbb{Q}_{\phi}\left(\boldsymbol{\theta}\right)}\right)\right]$$

$$\mathcal{L}\left(\mathbb{Q}\right) = \log\mathbb{P}\left(\mathcal{D}\right) - D_{\mathrm{KL}}\left(\mathbb{Q}_{\phi}\left(\boldsymbol{\theta}\right)||\mathbb{P}\left(\boldsymbol{\theta}|\mathcal{D}\right)\right) \to \max_{\phi}$$

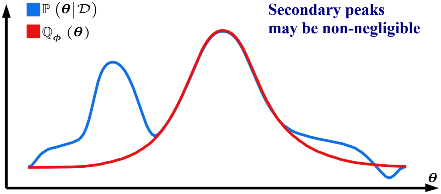
Backup: Boosted BBVI (1)



Posterior may be multimodal



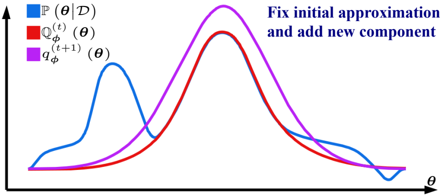
ELBO optimization will find the dominant peak



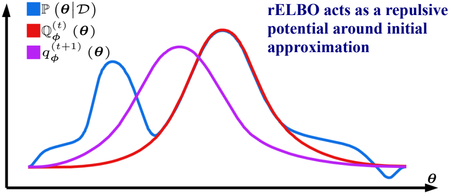
Secondary peaks may be non-negligible

- SVI depends on our assumption about the true posterior shape
- Due to high-variance gradients, it is hard to approximate multimodality by fitting a mixture
- Boosted BBVI algorithm:
  1. Find initial approximation by maximixing ELBO
  2. Fix initial approximation parameters

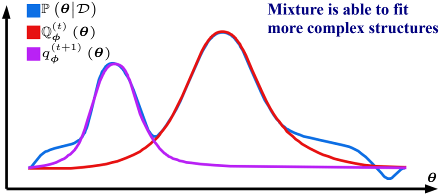
Backup: Boosted BBVI (2)



Fix initial approximation and add new component



rELBO acts as a repulsive potential around initial approximation



Mixture is able to fit more complex structures

- SVI depends on our assumption about the true posterior shape
- Due to high-variance gradients, it is hard to approximate multimodality by fitting a mixture
- Boosted BBVI algorithm:
  1. Find initial approximation by maximixing ELBO
  2. Fix initial approximation parameters
  3. Take next approximation component and construct a mixture
  4. Optimize added component parameters by maximixing rELBO

rELBO:

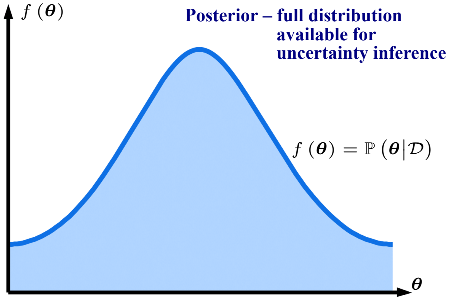
$$\mathcal{L}'(\mathbb{Q}) = \overbrace{\mathbb{E}_{\boldsymbol{\theta}\sim q_{\boldsymbol{\phi}}^{(t+1)}}\left[\log\left(\frac{\mathbb{P}(\mathcal{D},\boldsymbol{\theta})}{q_{\boldsymbol{\phi}}^{(t+1)}(\boldsymbol{\theta})}\right)\right]}^{\text{ELBO: approximation step has to be close to the posterior}} -$$

$$\underbrace{-\mathbb{E}_{\boldsymbol{\theta}\sim q_{\boldsymbol{\phi}}^{(t+1)}}\left[\log \mathbb{Q}_{\boldsymbol{\phi}}^{(t)}(\boldsymbol{\theta})\right]}_{\text{residual: but not too close to the previous approximation step}}$$

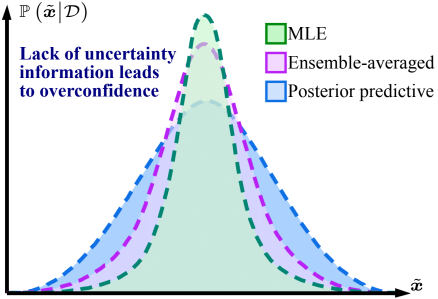
Backup: additional BNN motivation

- BNNs are hard to overtrain: $\mathbb{P}\left(\underset{\uparrow\,\text{Emphasis on parameters}}{\boldsymbol{\theta}}\middle|\mathcal{D}\right)$, while $\mathbb{P}\left(\overset{\text{Emphasis on data}\,\downarrow}{\mathcal{D}}\middle|\boldsymbol{\theta}\right)$

- BNNs perform better on OOD data
- BNNs are found to be generally more calibrated
- Bayesian formalizm scales intuitively to multi-stage problems



Posterior – full distribution available for uncertainty inference

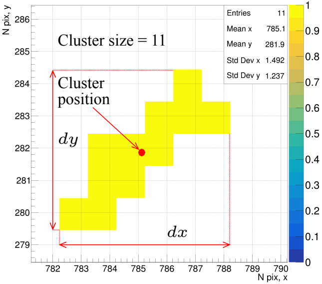


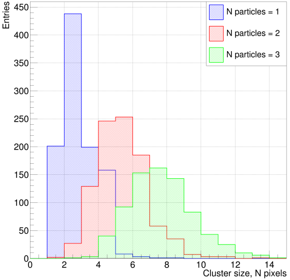
Lack of uncertainty information leads to overconfidence

Backup: BNN realization

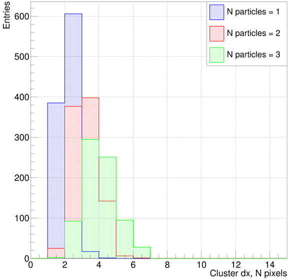
- Data: Geant4 full physical simulation. Allpix2 digitazation
- Network: three fully connected ReLU layers
- Inputs: 7×7 matrix with active pixels, cluster position within tracker
- Weights: Laplace prior; Biases: Normal prior;
- Approx. posterior: Laplace distribution

Generated cluster

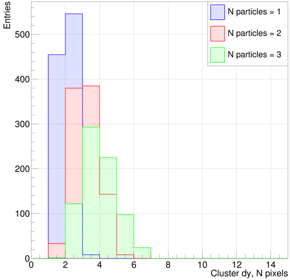




Cluster size distribution



Cluster dx distribution



Cluster dy distribution