Affine Parametric Neural Networks

FOR HIGH-ENERGY PHYSICS

Luca Anzalone[1,3], Tommaso Diotalevi[1,2,3], and Daniele Bonacorsi[1,3]
University of Bologna, CERN, INFN Bologna

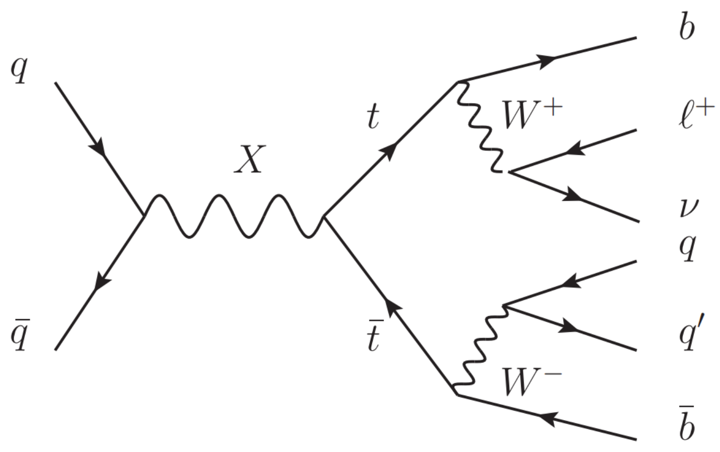
SIGNAL-BKG CLASSIFICATION

HEPMASS-IMB

Introduction

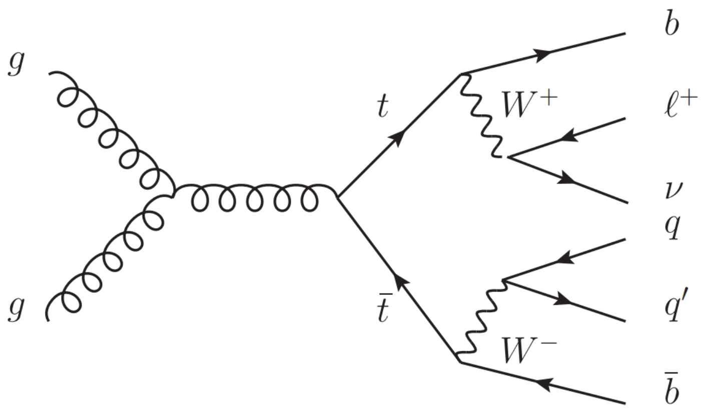
Signal-bkg Classification with HEPMASS

Problem: search for an hypothetical particle X with unknown mass.



Signal: particle X decaying to $t\bar{t}$.

The decay mode considered is $t\bar{t} \rightarrow W^+ b W^- \bar{b} \rightarrow qq'bl\nu\bar{b}$.

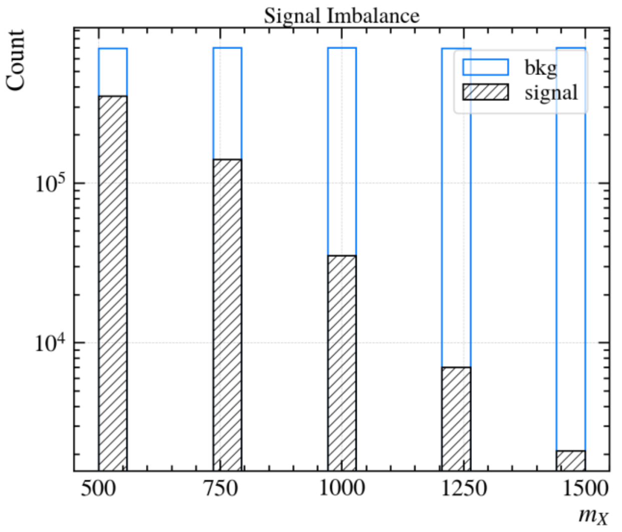


Background: Standard Model $t\bar{t}$ production, identical in decay mode but without the X resonance.

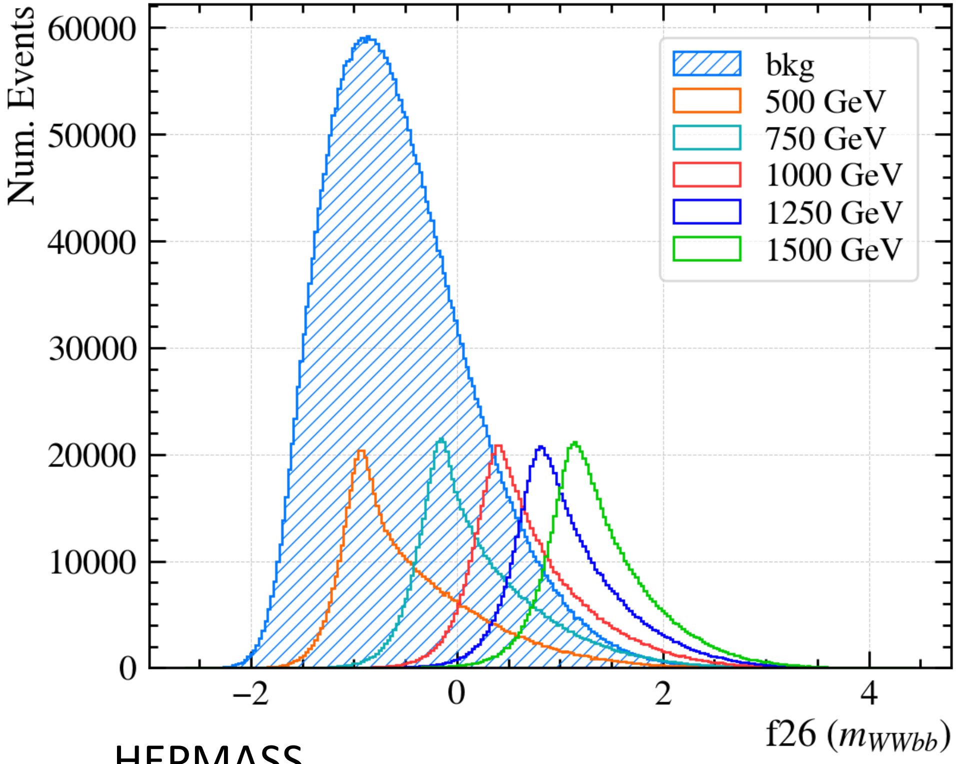
There are ***five* mass hypotheses** for the signal: $m_X = \{500,750,1000,1250,1500\}$ GeV.

HEPMASS-IMB

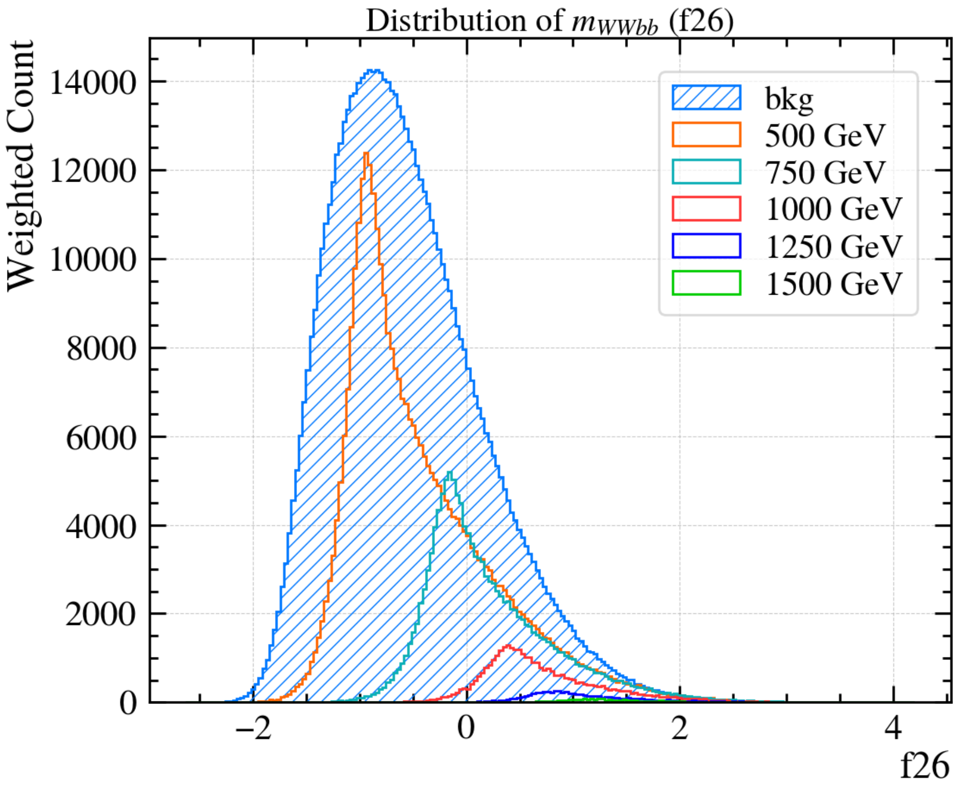
Both ***class*** and ***mass*** are imbalanced!



Double-imbalanced version of HEPMASS:



HEPMASS



HEPMASS-IMB
(bkg weighted by 1/5, for visualization)

BALDI’S PNN

CONDITIONING

AFFINE PNN

Parametric Neural Networks

Parametrized NNs

Neural network classifier with **two inputs**:

- The *features*, x
- The ***physics parameter***: in this case the signal mass hypotheses, m.

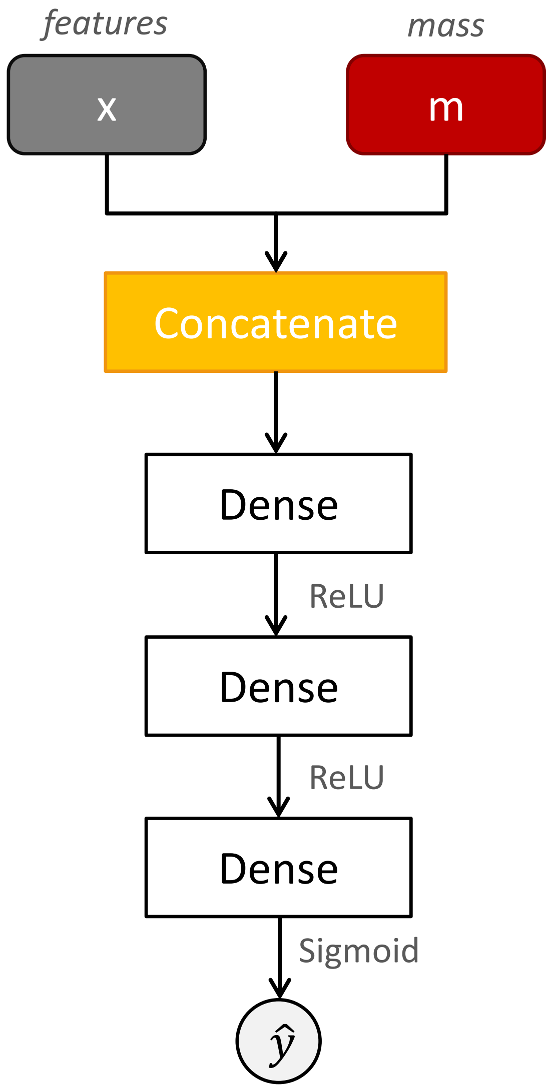
which are combined (e.g. by **concatenation**) to yield:

- $\hat{y} = f_\theta(x, \boldsymbol{m})$.

The **mass feature**, m, is responsible for «parametrizing» the NN:

- Can **replace** $M = |m|$ *individual* classifiers.
- Enables **interpolation** among known mass hypotheses.
- Potentially improves classification performance.

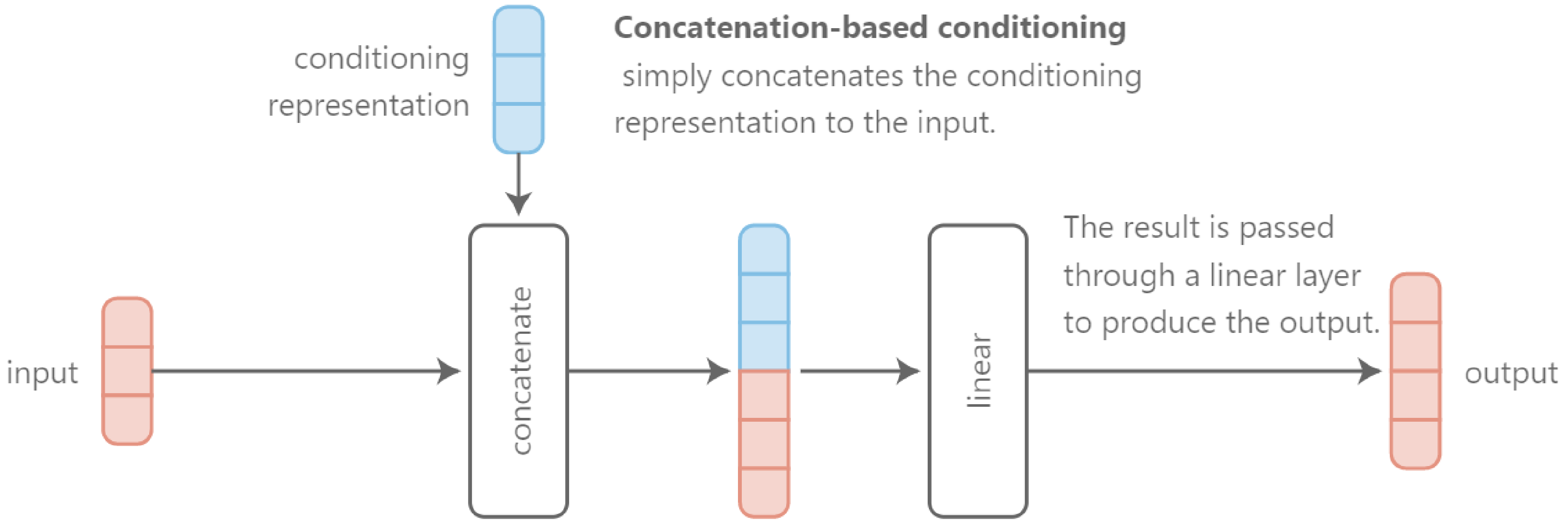
Q1: How to combine x with m?
Q2: How to assign m for the background?
Q3: How to evaluate interpolation?



Concatenation-based Conditioning

A simple **conditioning mechanism**:

$$z = W[x\ m] + \mathrm{b}$$

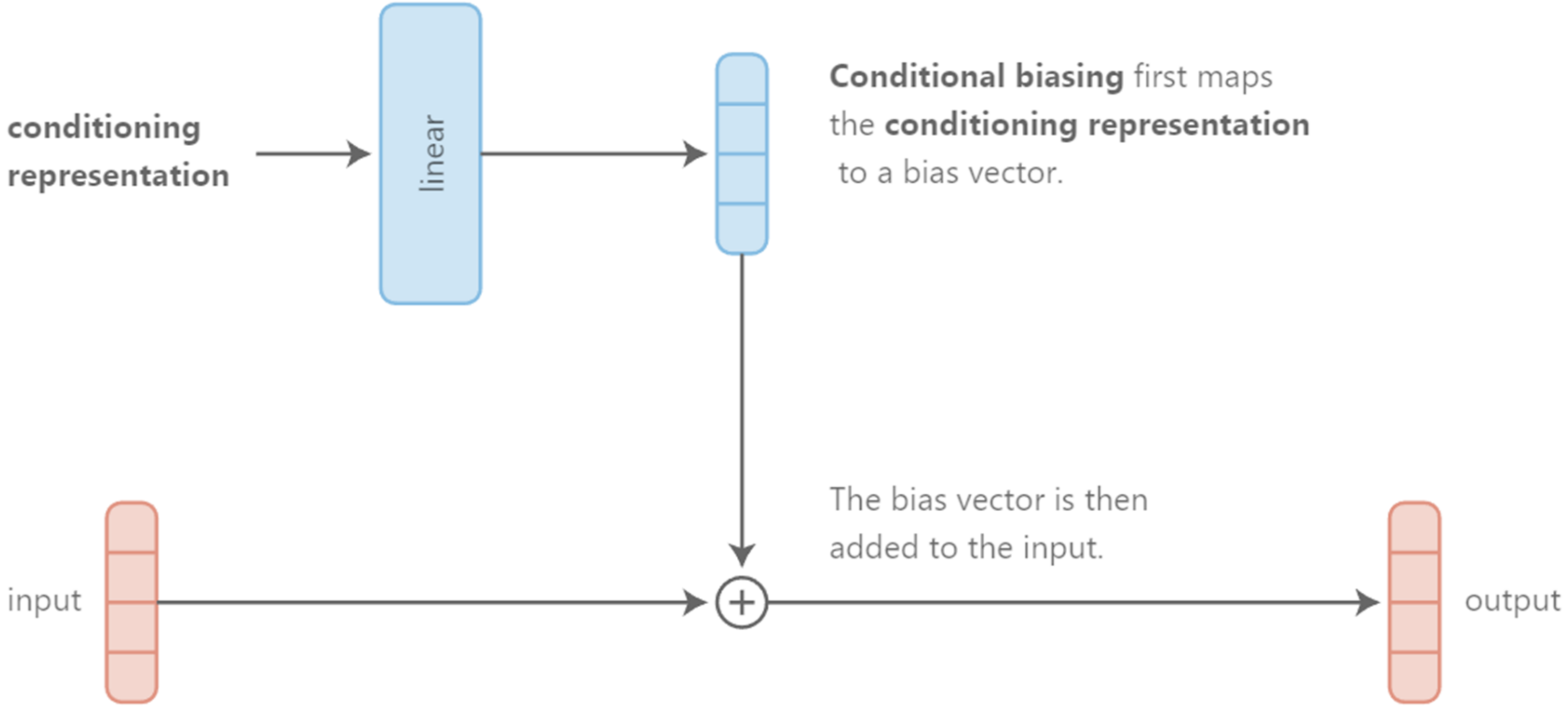


Parametric = *conditioning on a physics parameter*.

Conditional Biasing

Equivalent to concatenation-based conditioning (prev. slide):

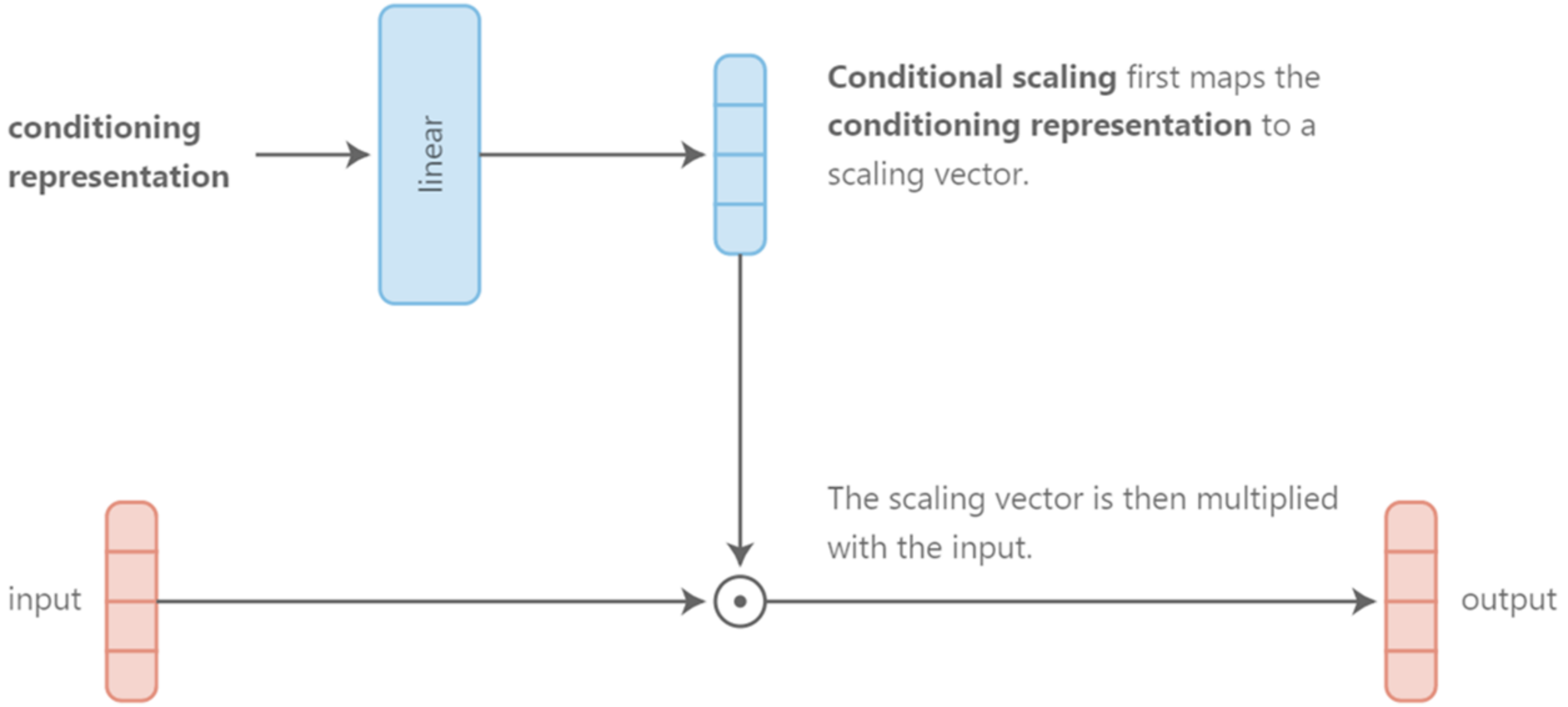
$$z = (Wm + b) + \mathrm{x}$$



Conditional Scaling

Alternative to concatenation and biasing:

$$z = x \odot (Wm + \mathrm{b})$$



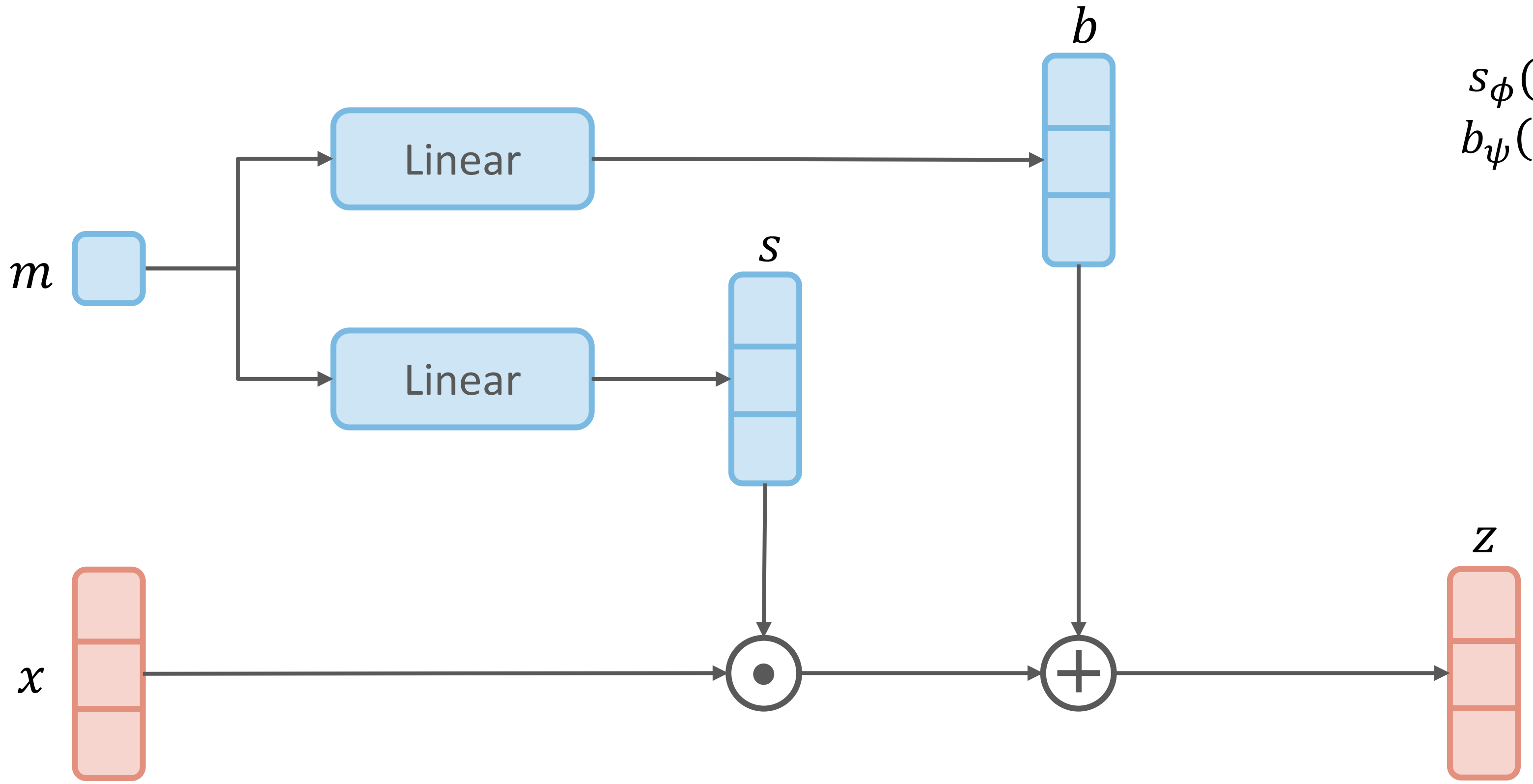
Affine Conditioning

Q1: ✓

A combination of *conditional scaling* and *conditional biasing*:

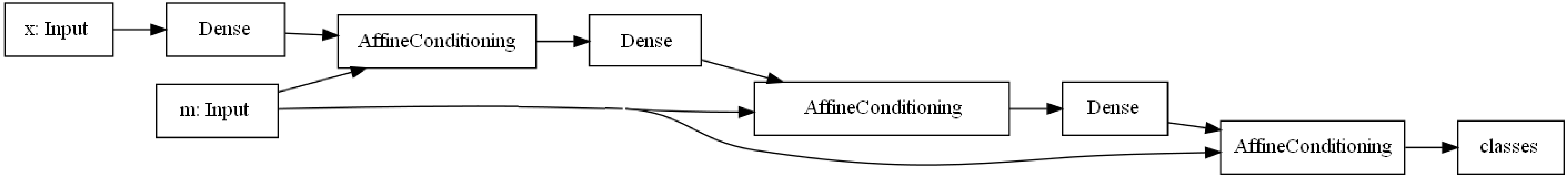
$$z = x \odot s_\phi(m) + \mathrm{b}_\psi(m)$$

$$s_\phi(m) = W_\phi m + b'$$
$$b_\psi(m) = W_\psi m + b''$$



Affine Parametric Neural Networks

Interleave multiple **affine-conditioning layers** in between *dense* layers, to better condition the neural network on the *mass feature*, m:



Full architecture:

- Four dense layers with 300, 150, 100 and 50 units: for a total of ~70k parameters.
- ReLU activation.
- Dropout ($p = 25\%$) after each affine-conditioning layer.

BACKGROUND MASS DISTRIBUTION

BALANCED TRAINING

Improving pNNs

Background’s Mass Distribution

Q2: ✓

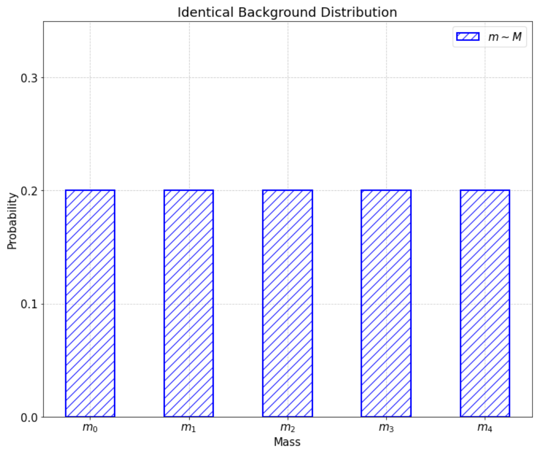
Given $M = \{m_0, m_1, \dots, m_K\}$ signal mass hypotheses, how to assign m for the background?

1. **Identical distribution:** M represents a ***discrete delta distribution***, so assign $m^{(i)}$ from it.

   For example: $m^{(i)} = m_1$, $m^{(j)} = m_3$, and $m^{(k)} = m_3$.

   Values outside the set M are not possible.

   $m^{(i)}$ is a *discrete* value.

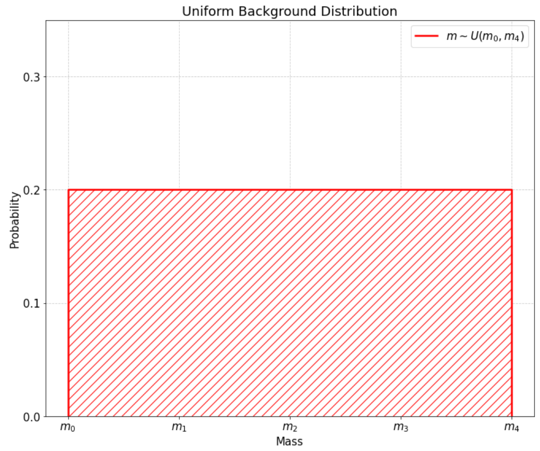


2. **Different distribution:** define a ***probability distribution*** from M.

   E.g. can be uniform $U(m_0, m_K)$, and so $m^{(i)} \sim U$.

   For example: $m^{(i)} = 505.5$, $m^{(j)} = 766.3$

   $m^{(i)}$ is now a *continuous* value.



Two implementations:

- **Fixed**: sampling of $m^{(i)}$ occurs *once* (e.g. beginning of training).
- **Sampled**: assignment of $m^{(i)}$ is done at each *mini-batch*.

Balanced Training

Exploit the ***structure*** of the dataset for training

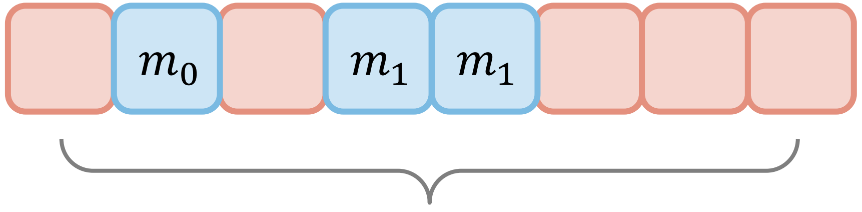
Suppose $\boldsymbol{D} = \{(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{m}, \boldsymbol{p})_i\}_{i=1}^{N}$, where:

- The ***mass label m*** is defined *only* for the signal: $m^{(i)} \in M, \forall i \in S = \{i \mid y^{(i)} = 1\}$.
- The ***process label p*** is defined *only* for the background: $p^{(i)} \in P, \forall i \in B = \{i \mid y^{(i)} = 0\}$.
- Let's $M = \{m_0, m_1, m_2, m_3\}$ and $P = \{\mathrm{p}_1, \mathrm{p}_2\}$.

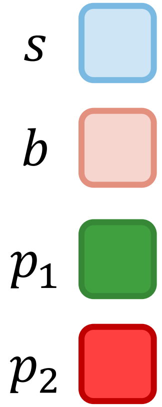
$\Rightarrow$ Both m and p divide $\boldsymbol{S}$ and $\boldsymbol{B}$, respectively, into **sub-classes**!

Balancing each *mini-batch* can remove ***imbalance*** among sub-classes.

No balance (default):



Notation



Each square is a sample; *some sub-classes may be underrepresented*, e.g. M.

Balanced Mini-batches

Class balance: same num. of samples per class, y (regardless m and p).

Background balance: same num. of samples per bkg process, p.

Signal balance: same num. of samples per mass, m.

Full balance: same num. of samples per tuple (y, m, p).

Class balance:

$$|s| = |b|$$

Background balance:

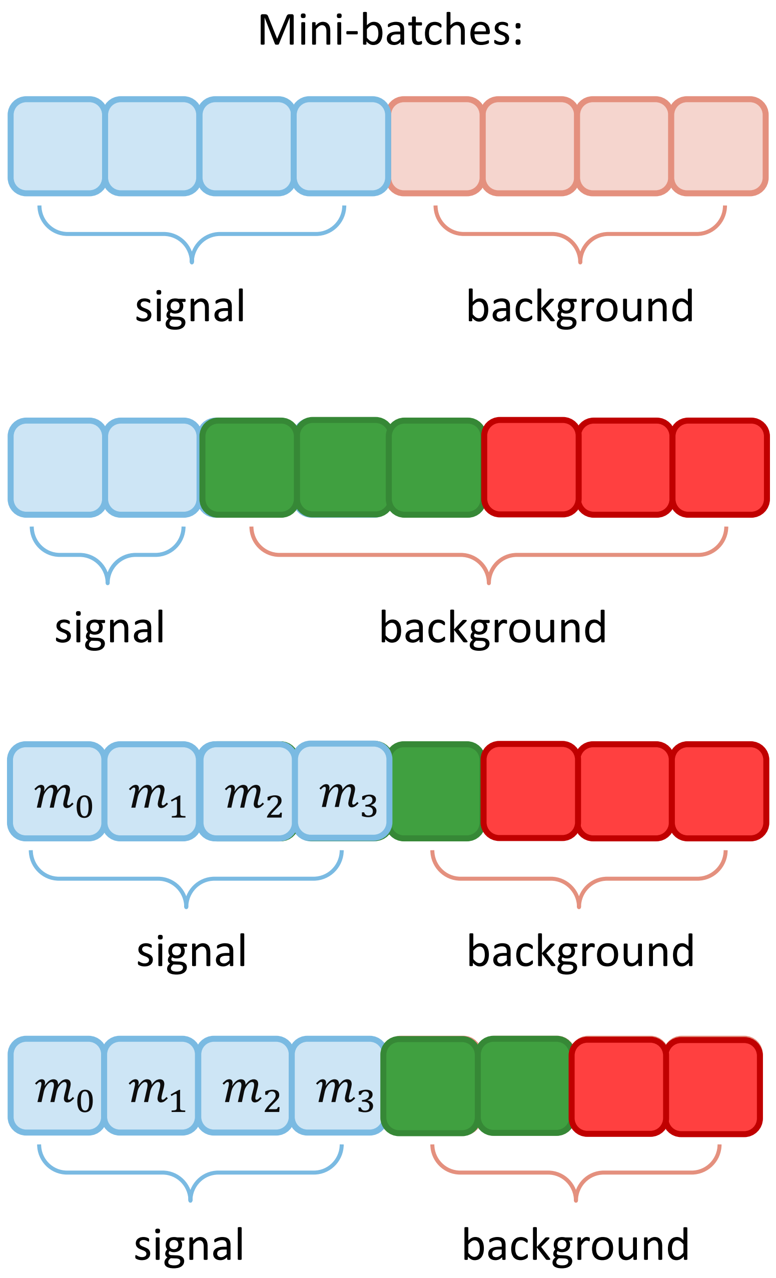
$$|p_1| = |p_2|$$

Signal balance:

$$|m_0| = |m_1| = \cdots = |m_3|$$

Full balance:

$$|s| = |b| \wedge |p| = |m|$$



METRICS

BASELINES

INTERPOLATION

Results

The Significance Ratio Metric

Along with ROC and PR curves, we introduce a new metric (evaluated $\forall t \in [0,1]$):

$$\sigma_{\text{ratio}} = \frac{\max_t \text{AMS}(t)}{s_{\max}/\sqrt{s_{\max}}} = \max_t \left\{ \frac{s_t \cdot \sqrt{s_{\max}}}{s_{\max} \cdot \sqrt{s_t + b_t}} \right\} = \frac{s_\star \cdot \sqrt{s_{\max}}}{s_{\max} \cdot \sqrt{s_\star + b_\star}},$$

where:

- $\text{AMS}(t) = \frac{s_t}{\sqrt{s_t+b_t}}$ is the **significance** computed at classification *threshold* t.
- $\frac{s_{max}}{\sqrt{s_{max}}}$ is the **ideal significance**, when $s_t = s$ (take all signal) and $b_t = 0$ (reject all bkg).

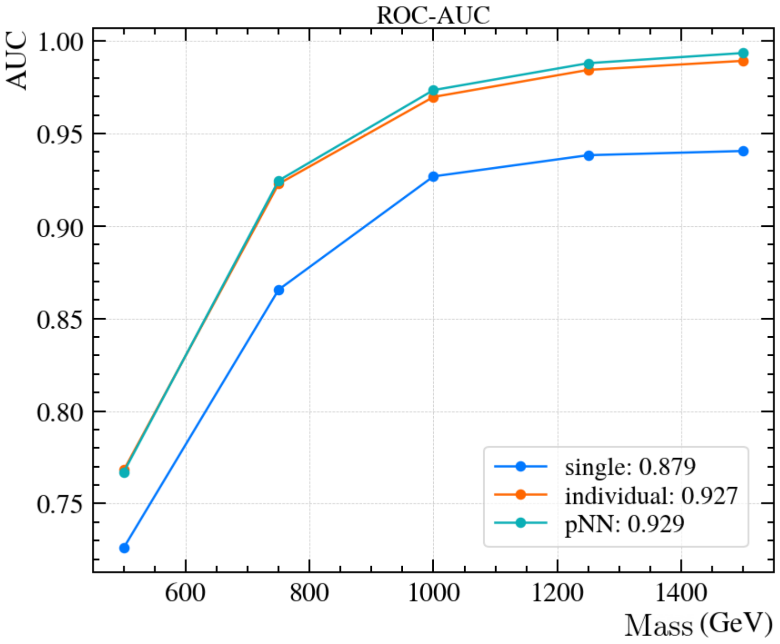
The metric is **normalized in** $[\mathbf{0}, \mathbf{1}]$, regardless the #signal and #background ⇒ Is comparable between different mass hypotheses.

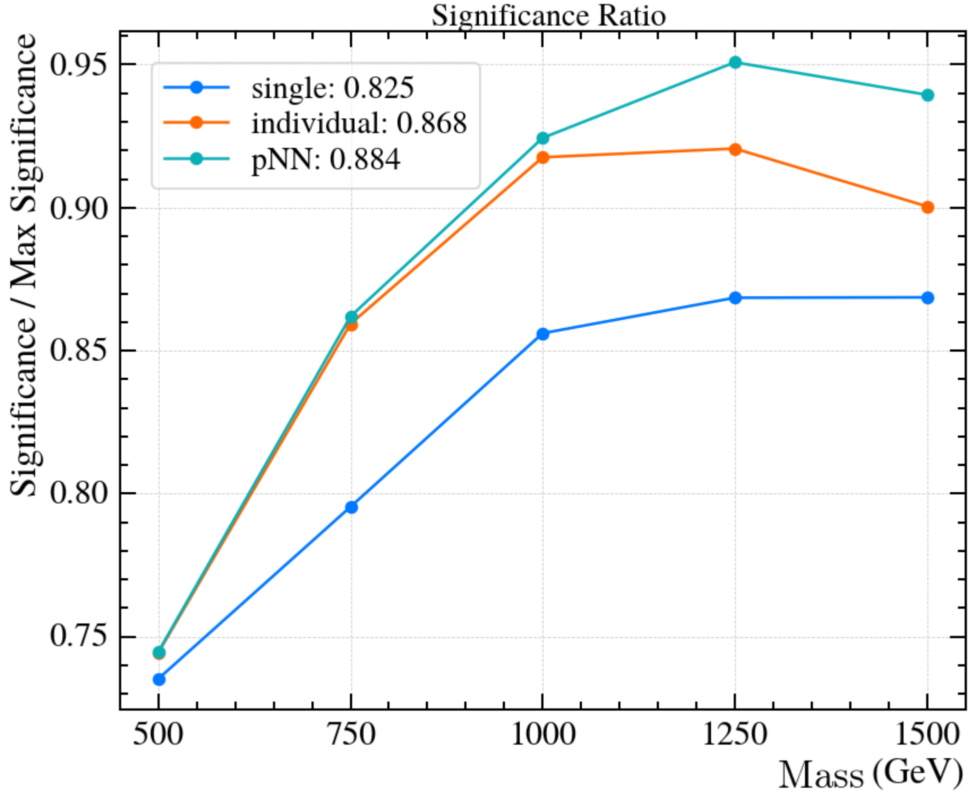
Baseline Models

The pNN outperforms even the set of individual neural networks.

There are three baselines:

- **Single-NN:** *one* neural network trained on all M mass points, but without the mass feature, m, as input – so ***not parametrized***.
- **Individual-NNs:** a *set* of $|M|$ neural networks, each trained on the corresponding mass point, $m_i \in M$.
- **pNN:** Baldi's like parametric neural network, without our improvements.





Interpolation

Interpolation capability implies ***twofold generalization***:

1. On new samples belonging to *training* mass points M, and
2. On novel samples related to the *missing* masses, $\bar{M}$.

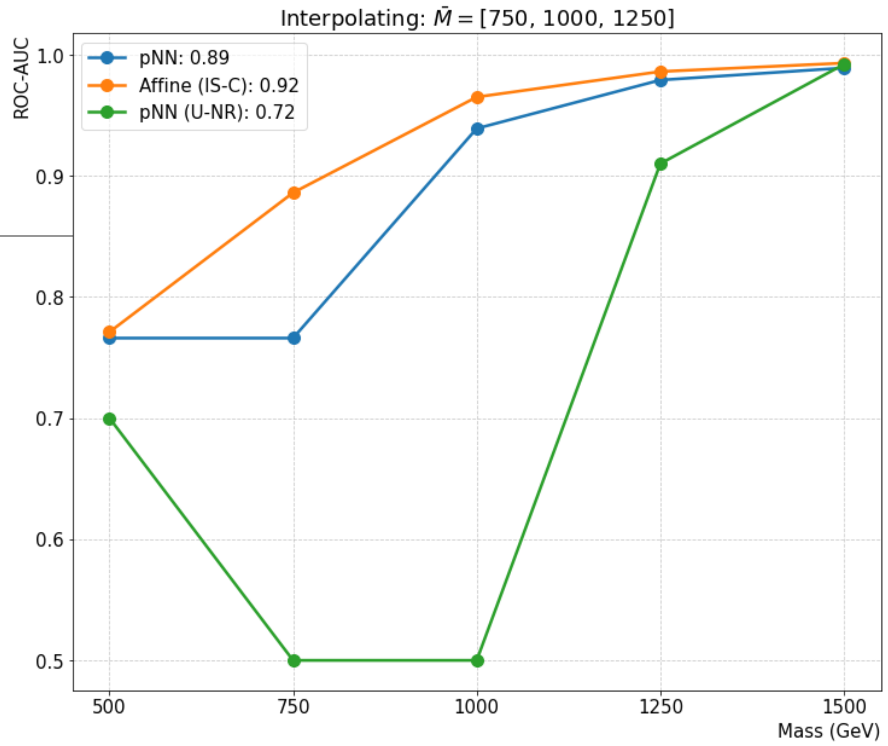
Factors affecting interpolation:

- Distribution of mass-correlated features.
- Background's mass distribution, and regularization.

How to evaluate it?

- Train only on one mass point to asses *similarity* among masses: force pNN to **extrapolate**.
- **Drop about half of the mass points for training.**
- Train on one mass *less*: usually **not enough** the establish interpolation ability.

Q3: ✓



IS-C: Identical-sampled; class-balance.

U-NR: Uniform; no regularization.

SUMMARY

REFERENCES

Conclusions

Summary

Parametric NNs can effectively **replace a set of $|M|$ classifiers**, when:

- The **physics parameter** (e.g. mass) is correctly assigned to the background: for the mass the identical (sampled) assignment strategy works the best.
- The **conditioning** on the parameter is meaningful: simple concatenation may be not enough.
- **Enough regularization** is employed to enable the model to interpolate.

Remember to exploit the structure and information in your own dataset to improve the model at the level of *architecture*, *conditioning* mechanism, and even *training*.

If you need **interpolation** at inference time, be sure to check for it by training a pNN on about 50% less mass points (as a rule of thumb).

References

Parameterized Neural Networks for High-Energy Physics – P. Baldi et al. 2016, EPJ

HEPMASS – UCI ML Repository

Feature-wise Transformations – Distill.pub, 2018

Improving Parametric Neural Networks for High-Energy Physics (and Beyond) – L. Anzalone et al, 2022, MLST, code (github).

HEPMASS-IMB – Zenodo

Thanks for the Attention!

Questions?

Contacts:

luca.anzalone2@unibo.it