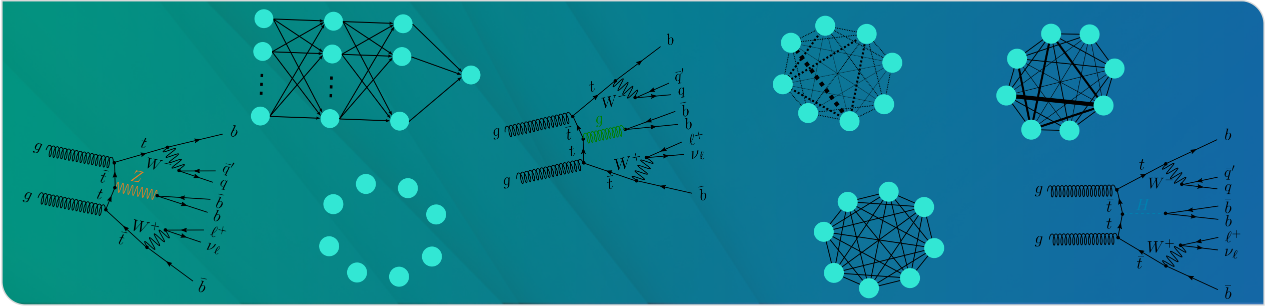
Karlsruhe Institute of Technology

Institute of Experimental Particle Physics

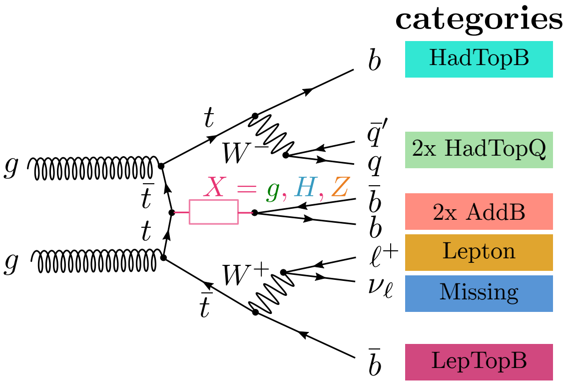
Feasibility and Reliability Studies of Graph Neural Networks for Multivariate $t\bar{t}$+X Event Classification at the CMS Experiment at CERN

Yee-Ying Christina Cung | January 09, 2023

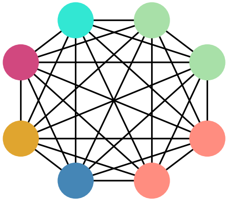


Outline

$t\bar{t}$+X Processes and Category Assignment

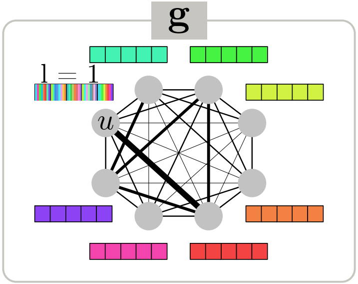


$\hat{=}$



$\Rightarrow$ $t\bar{t}$+$b\bar{b}$, $t\bar{t}$H($b\bar{b}$), $t\bar{t}$Z($b\bar{b}$)

Graph Neural Networks (GNNs)



- Calculation performed in each vertex u

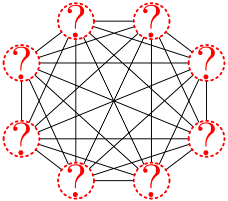
$$\mathbf{x}_u^{(l)} = \textsc{Upd}^{(l-1)}\left(\mathbf{x}_u^{(l-1)}, \textsc{Agg}^{(l-1)}\left(\left\{\mathbf{x}_u^{(l-1)}, \forall v \in \mathcal{N}(u)\right\}\right)\right)$$

- Gated Graph Sequence Neural Network (GGSNN) [1]:
  - AGG: mean, UPD: Gated Recurrent Unit (GRU) cell
- GraphConv [2]:
  - AGG: sum, UPD: NN

Application of GNNs

Karlsruhe Institute of Technology

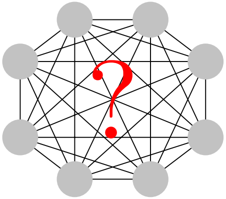
node-level-prediction (NLP)



? = AddB or not AddB

$\Rightarrow$ **Jet assignment**

graph-level-prediction (GLP)



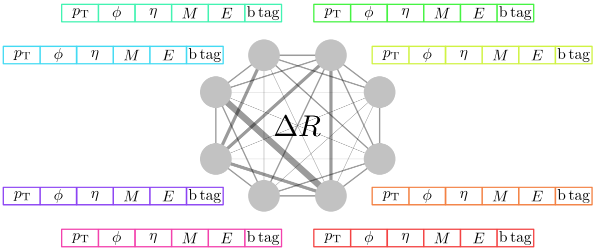
● = jet / lepton / MET

? = $t\bar{t}+b\bar{b}$ vs. $t\bar{t}H(b\bar{b})/t\bar{t}Z(b\bar{b})$ (binary)
$t\bar{t}+b\bar{b}$ vs. $t\bar{t}H(b\bar{b})$ vs. $t\bar{t}Z(b\bar{b})$ (multiclass)

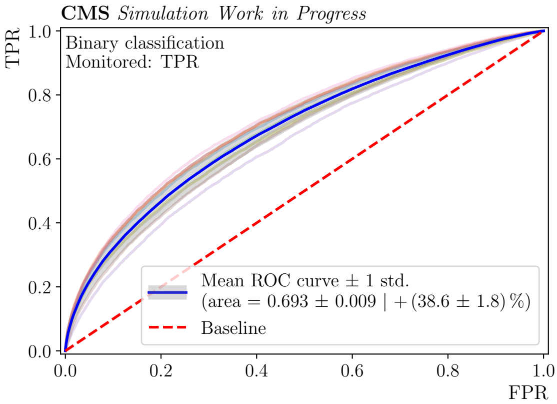
$\Rightarrow$ **Event classification**

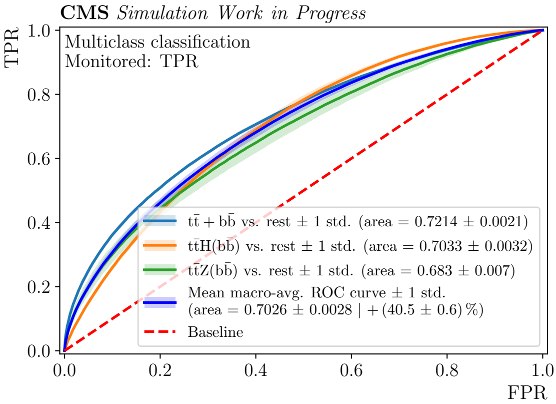
Training Data

- **Monte Carlo simulations** (2017)
- **Region**: $\geq$ 6 jets, $\geq$ 4 b-tagged jets, single lepton channel
- **Total number of events** $\approx$ 190k (60% training | 20% validation | 20% test)
  - $t\bar{t}+b\bar{b} \approx$ 53k
  - $t\bar{t}H(b\bar{b}) \approx$ 100k
  - $t\bar{t}Z(b\bar{b}) \approx$ 33k



Binary and Multiclass Classification - First Attempts





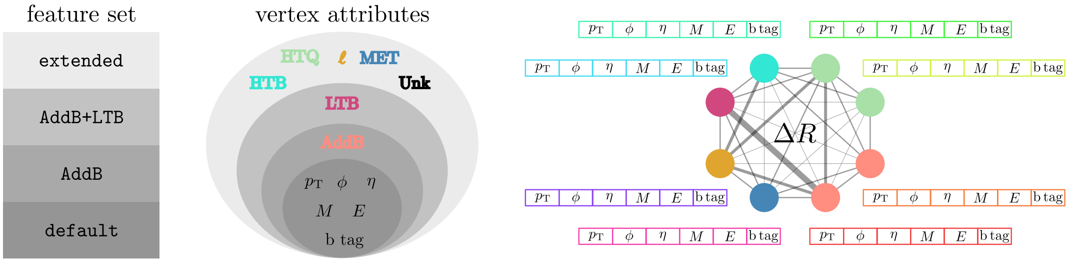
⇒ Less than 41 % better than a random estimator

⇒ Still a lot of room for improvements

Optimization Approaches

1.) **Monitored metric**: true positive rate (TPR) $\longrightarrow$ loss

2.) Extend vertex attributes by **category flags** $= \{0, 1\}$

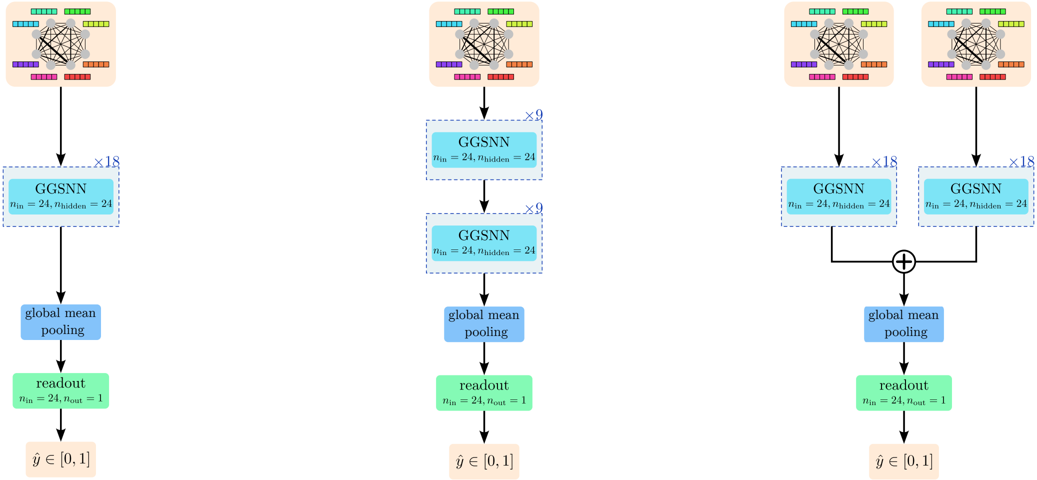


$\Rightarrow$ **Problem**: not inherent in detected data (reconstruction-level)

$\Rightarrow$ **Solution**: e.g., a GNN-based preclassifier (NLP) $\longrightarrow$ cf. Slide 29f

Optimization Approaches

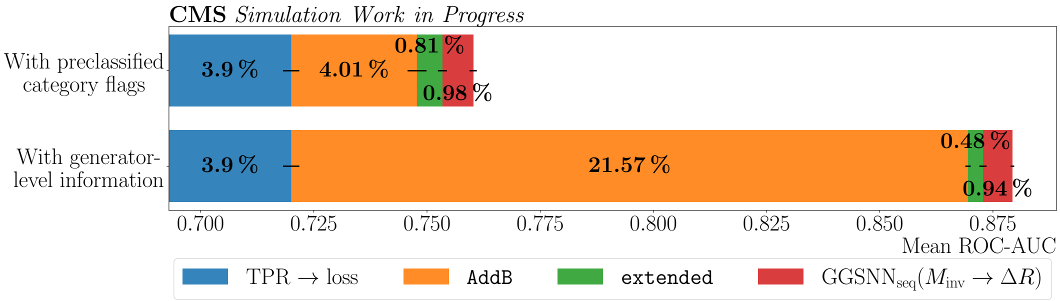
3.) Modify the **model architecture**



(a) GGSNN ($N_{\text{trainable param.}} \approx 14\text{k}$) (b) $\text{GGSNN}_{\text{seq}}$ ($N_{\text{trainable param.}} \approx 18\text{k}$) (c) $\text{GGSNN}_{\text{para}}$ ($N_{\text{trainable param.}} \approx 28\text{k}$)

Performance Improvements

Karlsruhe Institute of Technology



⇒ Performance improves by a total of **(26.9 ± 1.3)** % theoretically

⇒ About 76 % better than a random estimator

⇒ **GNNs are generally suitable for $t\bar{t}$+X event classification** ✔

Explainable AI: GNNExplainer vs. Taylor Coefficient Analysis[1]

Karlsruhe Institute of Technology

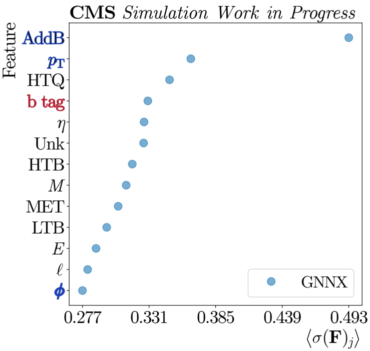
- **GNNExplainer** (GNNX) [3]
  - Designed for explaining GNNs
  - Training required
    → hyperparameters need to be selected
  - Explains the importance of:
    - Vertex attributes
    - Vertices
    - Vertex attributes per vertex
    - Edges/relational information
      (but w/o considering edge attributes!)

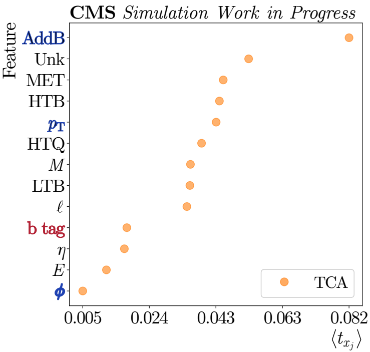
- **Taylor coefficient analysis** (TCA) [4]
  - More versatile → applicable to GNNs or DNNs
  - Deterministic
  - Explains the importance of:
    - Vertex attributes and their relations

[1] Further details on GNNX and TCA can be found, e.g., in the presentation in the ML meeting: https://indico.cern.ch/event/1175373/

Importance of Vertex Attributes

Karlsruhe Institute of Technology



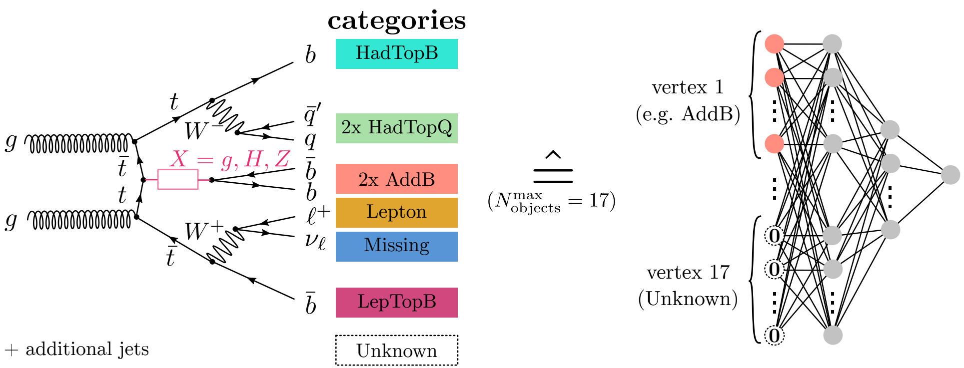


- Both rankings of the b tag's importance are reasonable from a physics perspective
  - In a further study: TCA's explanations are clearly more reasonable (cf. Slide 39)

$\Rightarrow$ **Explainable AI reveals: GNNs behave as expected $\rightarrow$ GNNs are indeed reliable ✔**

GNN and DNN Properties

	GNN	DNN	
input size (N_{objects} in an event)	flexible	fixed	→ affects data handling
permutation invariance	✔	✗	→ affects data handling
parameter sharing	✔	✗	→ affects degrees of freedom (DOF)



Conditions for Comparability

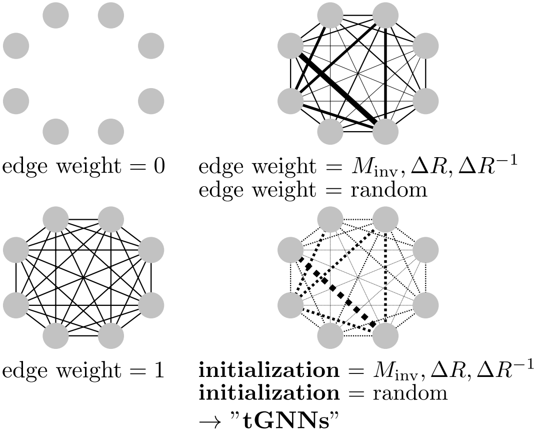
conditions	realizable	comments
a) same loss function	✔	BINARY CROSS-ENTROPY
b) same optimizer	✔	ADAM
c) same activation function	✔	RELU, SIGMOID
d) same feature space	✗	since no differentiation between vertex, edge and graph attributes for DNNs
e) same n_{input}	✗	due to d) and fixed input size of the DNNs
f) same n_{hidden}	✔	
g) same n_{output}	✔	
h) same $N_{\text{trainable param.}}$ (= DOF)	✔	because of e)

⇒ **Comparison A:** compare models with same n_{hidden} or

⇒ **Comparison B:** compare models with same $N_{\text{trainable param.}}$, cf. Slide 49ff

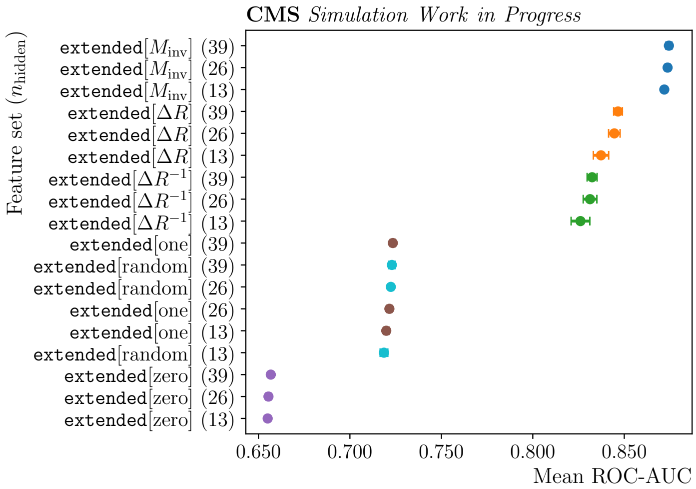
Analysis Strategy - Comparison A

- **Idea**: compare models with same $n_{\text{hidden}} \in N_{\text{hidden}}$: $N_{\text{HL}} = \{1, 2\}$, $N_{\text{hidden}} = \{13, 26, 39\}^{n_{\text{HL}} \in N_{\text{HL}}}$
- Deploy models that are as basic as possible:
  - **DNN**: fully-connected feed-forward neural network
  - **GNN**: GraphConv
- Enhance comparability by training GNNs with different graph connectivity schemes
- **Number of compared models**: 120 (GNNs) + 96 (DNNs) = 216



Performance of GNNs (1 HL)

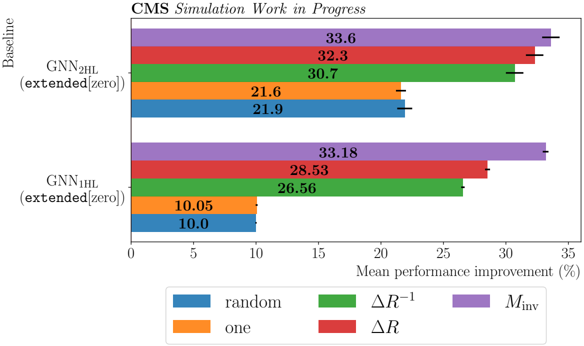
Karlsruhe Institute of Technology

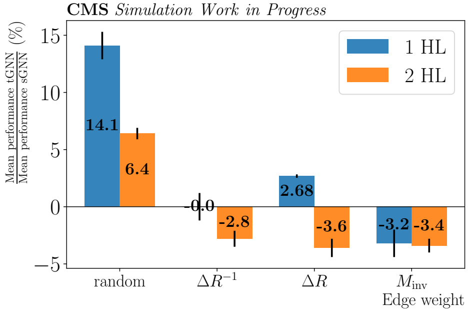


- Edge weight = 0 / prohibit message passing
  $\longrightarrow$ Rather random estimators
- Using physically **non-meaningful** or **physically motivated** edge weights
  $\longrightarrow$ Improves model performance

GNN: Performance Improvement by Different Edge Weights

Karlsruhe Institute of Technology



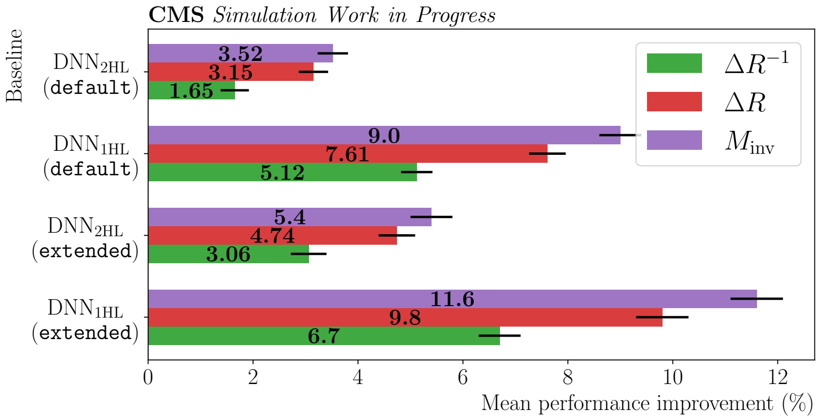


⇒ Edge weight = M_{inv} leads to the best GNN performance

⇒ Beneficial to train edges with physically non-meaningful weights

⇒ Edge weight = M_{inv} seems to be the **best choice** for $t\bar{t}$+X event classification

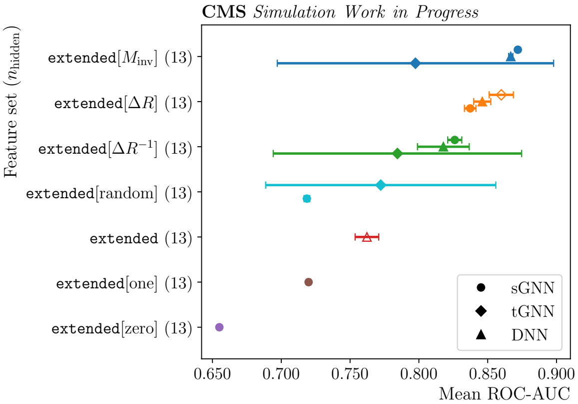
DNN: Performance Improvement by Different Edge Weights



⇒ Using relational information is also beneficial for the performance of DNNs

⇒ But: n_{input} increases from 221 to 493 → significant increase in $N_{\text{trainable param.}}$!

Combined Results of Representative Models (1 HL)



- **Large error bars** for tGNNs and DNNs
  → not stable training due to high number of DOF?
- **DNNs** trained w/o relational info **outperform** GNNs trained with graphs w/o physically motivated, untrained edge weights
- **GNNs** still work the **best** for $t\bar{t}$+X event classification
- Similar results for models with 2 HL

Summary and Outlook

Karlsruhe Institute of Technology

Feasibility Study

- Event classifier theoretically improves by about **27 %** overall

  ⇒ About 76 % better than a random estimator
  ⇒ **GNNs are generally suitable for $t\bar{t}$+X event classification** ✔

Benchmarking Equivalent GNNs and DNNs

	GNN	DNN
model performance	👍	👍
training stability	👍 / 👉	👉
DOF	👍	👉
data preprocessing effort	👍	👎

⇒ Beneficial to **prefer GNNs** to DNNs ✔
⇒ Outlook: non-CMS paper currently prepared

Reliability Study

- The features identified as important by GNNX and TCA are reasonable from a physics point of view

  ⇒ **GNNs are reliable/trustworthy** ✔

Outlook: develop a multi-task network?

- Simultaneously trained on additional b jet assignment and $t\bar{t}$+X event classification
- Advantage: end-to-end model
  → easier to be **retrained**, **optimized** and **distributed**

References

[1] Yujia Li, Daniel Tarlow, Marc Brockschmidt, et al. “Gated Graph Sequence Neural Networks”. In: *International Conference for Learning Representations (ICLR)* (2017). arXiv: 1511.05493v4 [cs.LG].

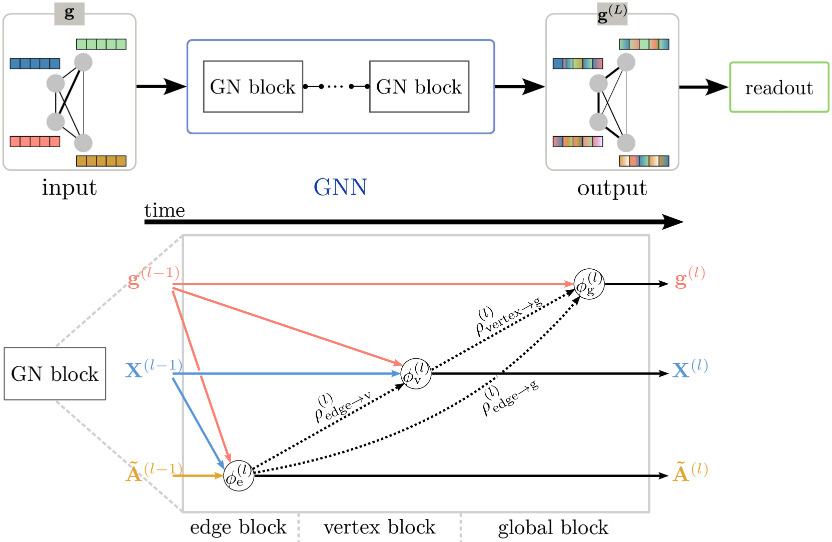
[2] Christopher Morris, Martin Ritzert, Matthias Fey, et al. “Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33(01). 2019, pp. 4602–4609. DOI: 10.1609/aaai.v33i01.33014602.

[3] Rex Ying, Dylan Bourgeois, Jiaxuan You, et al. “GNNExplainer: Generating Explanations for Graph Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, et al. Vol. 32. 2019. URL: https://proceedings.neurips.cc/paper/2019/file/d80b7040b773199015de6d3b4293c8ff-Paper.pdf.

[4] Stefan Wunsch, Raphael Friese, Roger Wolf, et al. “Identifying the Relevant Dependencies of the Neural Network Response on Characteristics of the Input Space”. In: *Computing and Software for Big Science* 2(5) (2018). DOI: 10.1007/s41781-018-0012-1.

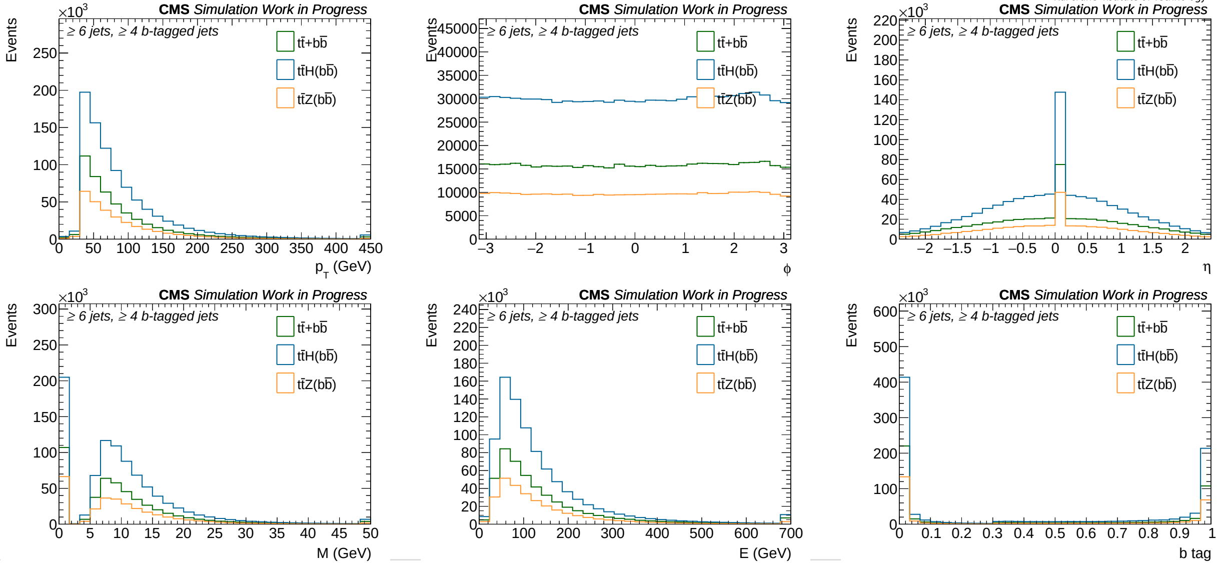
Backup

Graph Network Formalism



Distribution of Input Variables

Karlsruhe Institute of Technology



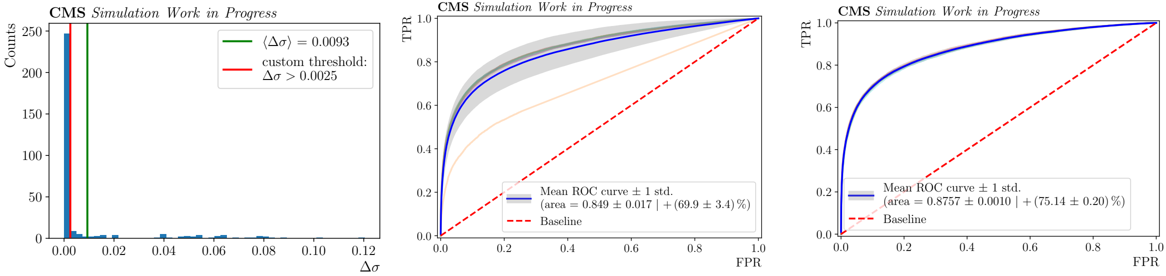
Feasibility Study

Outlier Criteria

Karlsruhe Institute of Technology

a) If the trained model is a random estimator (ROC-AUC = 0.5) **or**

b) ROC-AUC $\notin$ mean ROC-AUC $\pm 1.5 \cdot \sigma^{\text{pre}}_{\text{ROC-AUC}}$ **and** $\Delta\sigma = \sigma^{\text{pre}}_{\text{ROC-AUC}} - \sigma^{\text{post}}_{\text{ROC-AUC}} > 0.0025$

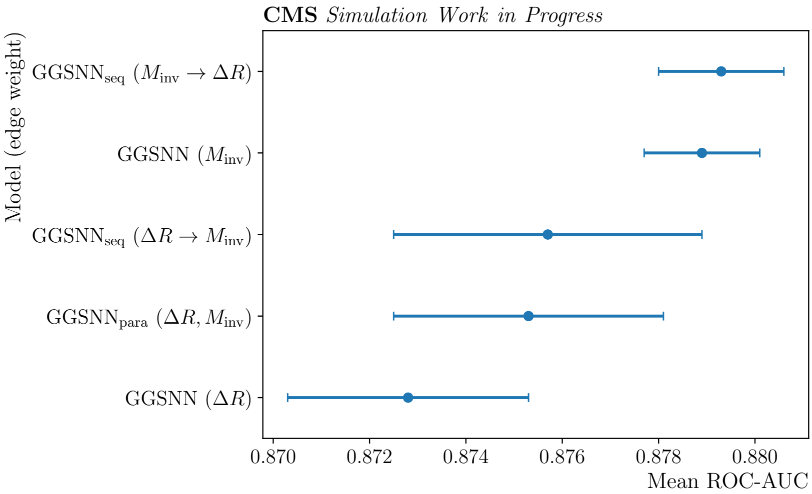


Left: Histogramm of the standard deviation difference pre- and post-removal of models with ROC-AUC values beyond the range of mean ROC-AUC $\pm 1.5 \cdot \sigma^{\text{pre}}_{\text{ROC-AUC}}$. Middle: Exemplary ROC curve of a trained model fulfilling criterion b). Right: Exemplary ROC curve of a trained model fulfilling criterion b), which is not desired, if $\Delta\sigma > 0.0025$ would be omitted.

Training Information

hyperparameter	setting
$n_{\text{input}}/n_{\text{hidden}}$	24
n_{HL}	18
n_{output} (of readout)	1 (binary), 3 (multiclass)
bias	true
aggregation functions	mean
global pooling method	mean
maximum number of epochs	200
EARLY-STOPPING	$\Delta\text{epoch} = 15$, $\Delta\text{TPR} = 0.01$ or $\Delta\text{epoch} = 15$, $\Delta\text{loss} = 0.001$
mini-batch size	200
optimizer	ADAM ($\gamma = 0.01$)
activation function (in output layer)	SIGMOID (binary), SOFTMAX (multiclass)
loss function	BINARY/CATEGORICAL CROSS-ENTROPY
number of repetitions	10

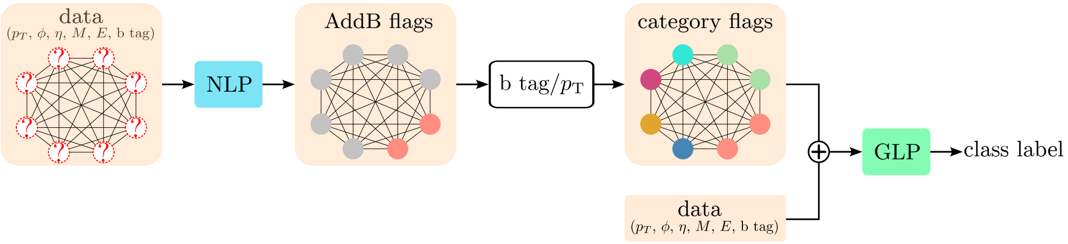
Different Edge Weights and Model Architectures



Preclassification of Category Flags

Karlsruhe Institute of Technology

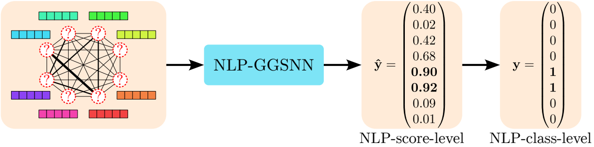
- With a GNN-based preclassifier (NLP), an overall improvement of about **10 %** is still achievable

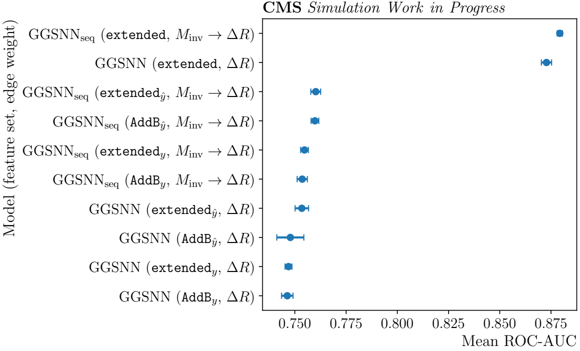


- Modeling of the dependency of the event classifier on the preclassification shows (cf. Slide 31ff):
  - Optimizing the preclassifier's TPR just by ≈ **0.17 %** → **2 %** better event classifier
  - **But:** a further optimization of the preclassifier's TPR by ≈ **6 %** would be required for improving the performance of the event classifier by another **2 %**

Preclassification of Category Flags

Karlsruhe Institute of Technology





- True positive rate achieved with the GNN-based preclassifier + the joint b tag/p_T approach

category	TPR (%)
AddB	70.88
HadTopB	65.61
HadTopQ	79.04
LepTopB	52.26
Unknown	62.24
Lepton/Missing	100.00

Dependency of the Event Classifier on the Preclassification

- **Idea**:
  - Manipulate the category flags of an increasingly larger fraction of the events in the data set
  - Modeling can be simplified to only modeling the additional b jet assignment correctly

- **AddB-LTB modeling**:
  - AddB flag $\leftrightarrow$ LTB flag

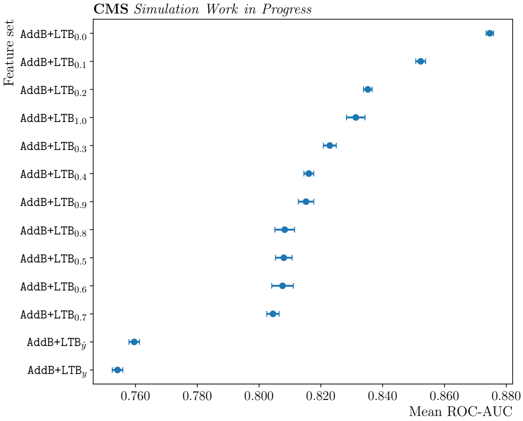
$$\mathbf{x}_i = \begin{pmatrix} \cdots & \text{AddB} = 1 & \cdots & \text{LTB} = 0 & \cdots \end{pmatrix}^T$$
$$\leftrightarrow \begin{pmatrix} \cdots & \text{AddB} = 0 & \cdots & \text{LTB} = 1 & \cdots \end{pmatrix}^T$$

  - The preclassifier confuses these categories the most
  - Only 1 LepTopB jet but 2 AddB jets in each event $\longrightarrow$ AddB jet to manipulate is randomly chosen

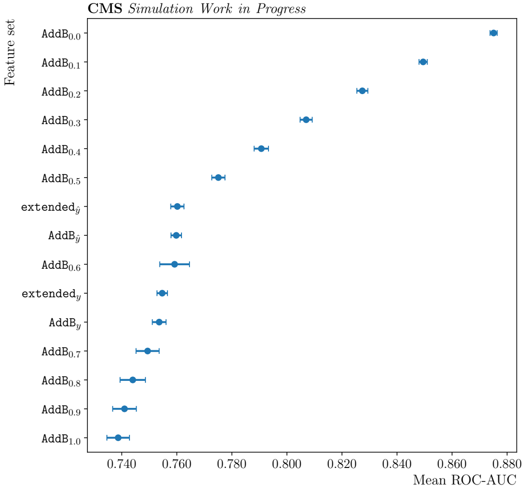
- **AddB-X modeling**:
  - AddB flag $\leftrightarrow$ any other category flag
  - Category flag with which it is manipulated in an event is chosen on the basis of the normalized preclassifier's *class specific* confusion rate (CR), 1/2 and 0/2 rates

class	$\langle CR_{HadTopB} \rangle$	$\langle CR_{LepTopB} \rangle$	$\langle CR_{HadTopQ} \rangle$	$\langle CR_{Unknown} \rangle$	$\langle CR_{Lepton} \rangle$ $\langle CR_{Missing} \rangle$	1/2 rate	0/2 rate
$t\bar{t}H(b\bar{b})$	36.89 ± 0.19	51.23 ± 0.19	8.49 ± 0.08	3.397 ± 0.033	0.0 ± 0.0	89 ± 10	11 ± 10
$t\bar{t}Z(b\bar{b})$	32.58 ± 0.14	52.97 ± 0.16	10.84 ± 0.12	3.608 ± 0.033	0.0 ± 0.0	88 ± 11	12 ± 11
$t\bar{t}+b\bar{b}$	34.12 ± 0.12	48.88 ± 0.13	10.52 ± 0.08	6.48 ± 0.05	0.0 ± 0.0	84 ± 11	16 ± 11

Dependency of the Event Classifier on the Preclassification



(a) AddB-LTB modeling



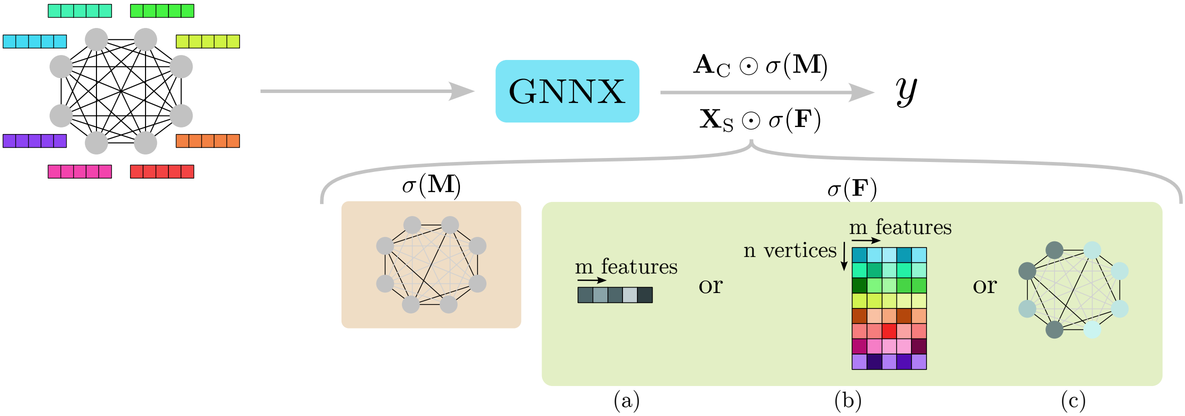
(b) AddB-X modeling

Properties of the Manipulated Data Sets

modeling strategy	fraction of manipulated events	fraction of manipulated objects in the categories AddB	HadTopB	LepTopB	HadTopQ	Unknown	Lepton/Missing
AddB-LTB	10	5.00	0.0	10.00	0.0	0.0	0.0
	20	10.00	0.0	20.00	0.0	0.0	0.0
	30	15.00	0.0	30.00	0.0	0.0	0.0
	40	20.00	0.0	40.00	0.0	0.0	0.0
	50	25.00	0.0	50.00	0.0	0.0	0.0
	60	30.00	0.0	60.00	0.0	0.0	0.0
	70	35.00	0.0	70.00	0.0	0.0	0.0
	80	40.00	0.0	80.00	0.0	0.0	0.0
	90	45.00	0.0	90.00	0.0	0.0	0.0
	100	50.00	0.0	100.0	0.0	0.0	0.0
AddB-X	10	5.60	4.10	5.62	0.61	0.34	0.0
	20	11.21	8.17	11.31	1.21	0.66	0.0
	30	16.80	12.24	16.97	1.79	1.00	0.0
	40	22.45	16.44	22.63	2.38	1.34	0.0
	50	28.07	20.62	28.19	2.98	1.70	0.0
	60	33.69	24.74	33.85	3.58	2.03	0.0
	70	39.32	28.97	39.41	4.18	2.39	0.0
	80	44.91	33.14	45.00	4.76	2.71	0.0
	90	50.53	37.31	50.60	5.35	3.07	0.0
	100	56.13	41.49	56.18	5.94	3.40	0.0

Reliability Study

GNNExplainer



Taylor Coefficient Analysis

Karlsruhe Institute of Technology

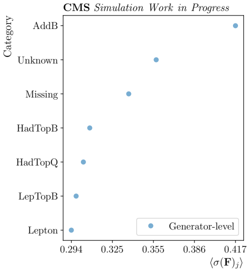
- **Idea**: perform a Taylor expansion on the model function Φ at the expansion points $\mathbf{z} \in \mathbb{R}^m$

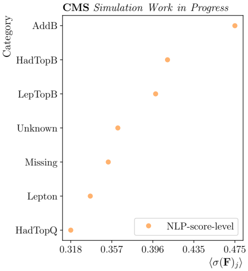
$$T_\Phi(x_1,\ldots,x_m) = \sum_{n_1=0}^{\infty} \cdots \sum_{n_m=0}^{\infty} \left(\frac{\partial^{n_1+\cdots+n_m}\Phi(z_1,\ldots,z_m)}{\partial x_1^{n_1}\cdots \partial x_m^{n_m}} \right) \frac{(x_1-z_1)^{n_1}\cdots(x_m-z_m)^{n_m}}{n_1!\cdots n_m!}$$

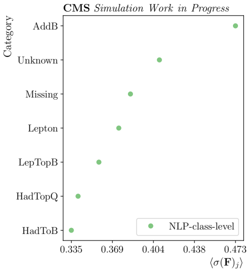
$$= \underbrace{\Phi(z_1,\ldots,z_m)}_{\equiv t_0} + \sum_{j=1}^{m} \underbrace{\frac{\partial\Phi(z_1,\ldots,z_m)}{\partial x_j}}_{\equiv t_{x_j}}(x_j-z_j) + \frac{1}{2!}\sum_{j=1}^{m}\sum_{k=1}^{m} \underbrace{\frac{\partial^2\Phi(z_1,\ldots,z_m)}{\partial x_j \partial x_k}}_{\equiv t_{x_j x_k}}(x_j-z_j)(x_k-z_k) + \ldots$$

$\Rightarrow$ The Taylor coefficients $t_\alpha, \alpha \in \{x_j, x_j x_k, \ldots\}$ are a measure of the importance of the corresponding features

GNNExplainer



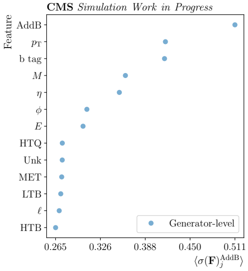


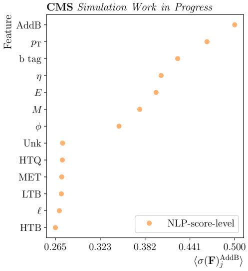


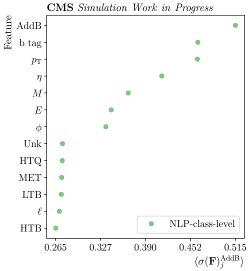
GNNX:

Generator-level vs. NLP-score-level:
$\langle \Delta r \rangle = 1.71$ ($\hat{=}$ 50.0 %)
$\Delta r_{\max} = 3.00$ (LepTopB)

Generator-level vs. NLP-class-level:
$\langle \Delta r \rangle = 1.14$ ($\hat{=}$ 66.67 %)
$\Delta r_{\max} = 3.00$ (HadToB, Lepton)







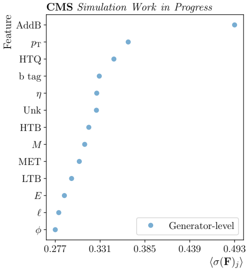
GNNX:

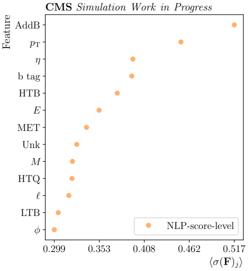
Generator-level vs. NLP-score-level:
$\langle \Delta r \rangle = 0.62$ ($\hat{=}$ 90.48 %)
$\Delta r_{\max} = 2.00$ (M, E)

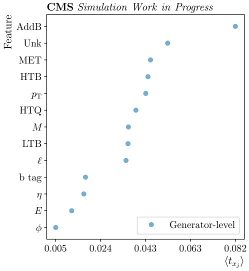
Generator-level vs. NLP-class-level:
$\langle \Delta r \rangle = 0.62$ ($\hat{=}$ 90.48 %)
$\Delta r_{\max} = 1.00$
(p_T, ϕ, η, M, E, b tag, HTQ, Unk)

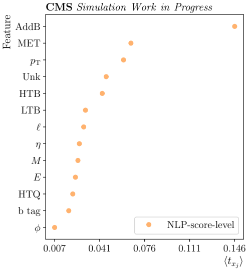
GNNX vs. TCA - Feature Importance

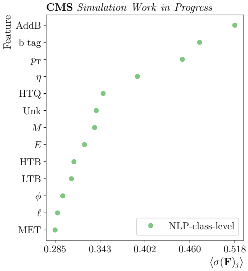
Karlsruhe Institute of Technology







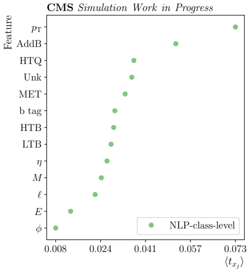




GNNX:

Generator-level vs. NLP-score-level:
$\langle \Delta r \rangle = 1.85$ ($\hat{=}$ 71.43 %)
$\Delta r_{\max} = 7.00$ (HTQ)

Generator-level vs. NLP-class-level:
$\langle \Delta r \rangle = 1.38$ ($\hat{=}$ 78.57 %)
$\Delta r_{\max} = 4.00$ (MET)

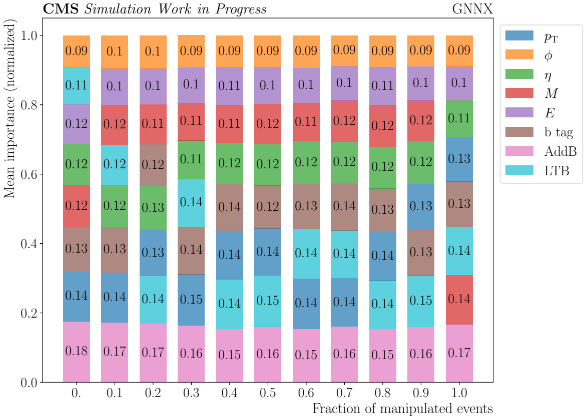


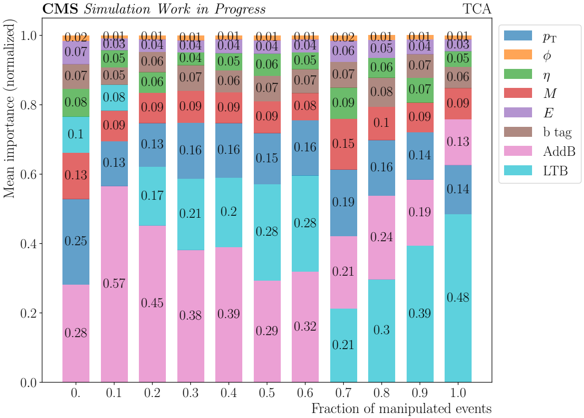
TCA:

Generator-level vs. NLP-score-level:
$\langle \Delta r \rangle = 1.85$ ($\hat{=}$ 71.43 %)
$\Delta r_{\max} = 5.00$ (HTQ)

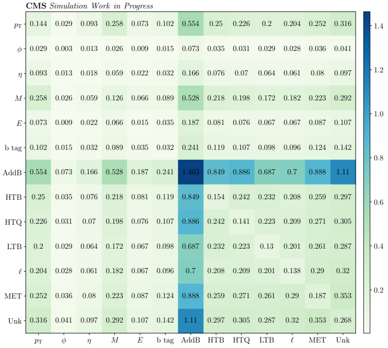
Generator-level vs. NLP-class-level:
$\langle \Delta r \rangle = 2.0$ ($\hat{=}$ 69.05 %)
$\Delta r_{\max} = 4.00$ (p_T, b tag)

Evolution of the Feature Importance in AddB-LTB Modeling

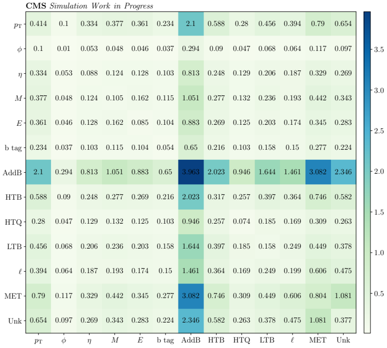




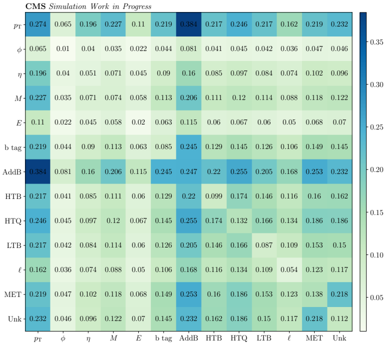
Second-Order TCA



(a) Generator-level GNN



(b) NLP-score-level GNN



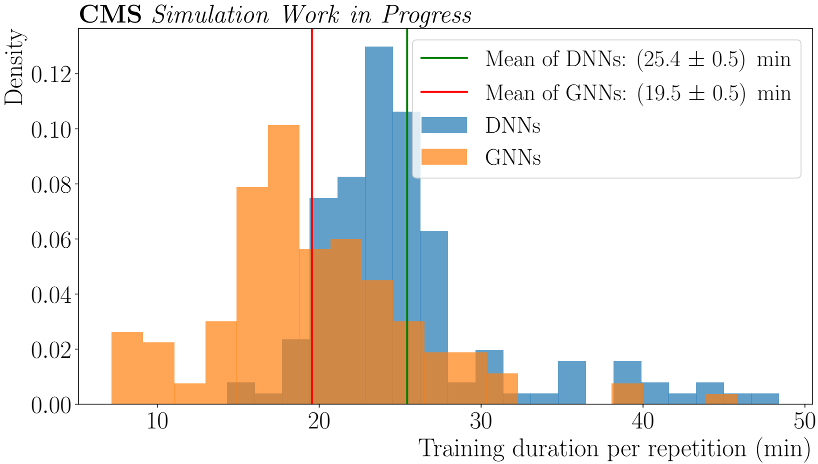
(c) NLP-class-level GNN

GNNs vs. DNNs

Training Information

hyperparameter	GNN	DNN
n_{input} (feature set)	13 (`extended`*)	102 (`default`) 221 (`extended`) 374 (`default`*) 493 (`extended`*)
N_{HL}	$\{1, 2\}$	
N_{hidden}	$\{13, 26, 39\}^{n_{\text{HL}} \in N_{\text{HL}}}$	
n_{output} (of readout)	1	
bias	true	
aggregation functions	sum	
global pooling method	mean	
maximum number of epochs	200	
EARLY-STOPPING	$\Delta\text{epochs} = 15$, $\Delta\text{loss} = 0.001$	
mini-batch size	200	
optimizer	ADAM ($\gamma = 0.01$)	
activation function (in hidden layers)	RELU	
activation function (in output layer)	SIGMOID	
loss function	BINARY CROSS-ENTROPY	
number of repetitions	10	

Training Duration



Note that these values are only of diminished expressive power and should rather be seen as a rough trend since the utilized hardware was not solely used for processing the trainings.

Convergence Speed and Degrees of Freedom

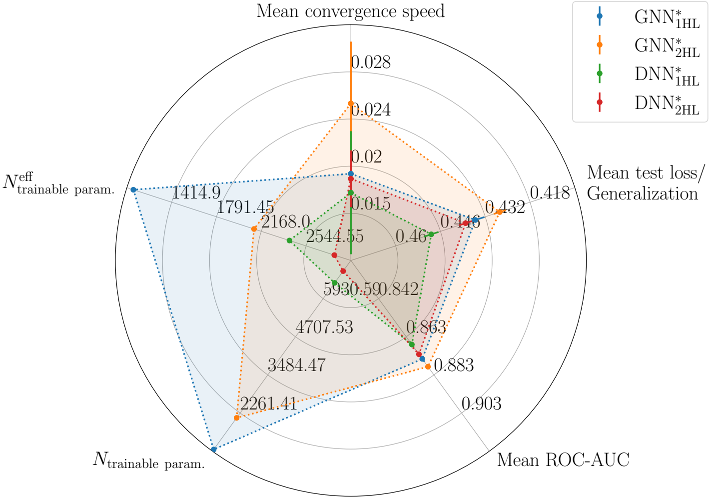
model A	model B (baseline)	$\langle \Delta \text{speed} \rangle$ (%)	$\langle \Delta N_{\text{trainable param.}} \rangle$ (%)
$\text{sGNN}_{1\text{HL}}$	$\text{DNN}_{1\text{HL}}$	-20.1 ± 3.3	-94.33
$\text{tGNN}_{1\text{HL}}$	$\text{DNN}_{1\text{HL}}$	26 ± 13	-88.47
$\text{sGNN}_{2\text{HL}}$	$\text{DNN}_{2\text{HL}}$	4.1 ± 2.5	-84.49
$\text{tGNN}_{2\text{HL}}$	$\text{DNN}_{2\text{HL}}$	31 ± 4	-68.36

Best Models

	GNN		DNN	
edge weight	M_{inv}	M_{inv}	M_{inv}	M_{inv}
n_{hidden}	(39)	$(26, 26)$	(13)	$(13, 26)$
$N_{\text{trainable param.}}$	1093	2107	6436	6813
$N^{\text{eff}}_{\text{trainable param.}}$	—	—	2405	2782
mean ROC-AUC	0.87441 ± 0.00051	0.87860 ± 0.00035	0.86676 ± 0.00050	0.87198 ± 0.00044
identifier	$\text{GNN}^*_{\text{1HL}}$	$\text{GNN}^*_{\text{2HL}}$	$\text{DNN}^*_{\text{1HL}}$	$\text{DNN}^*_{\text{2HL}}$

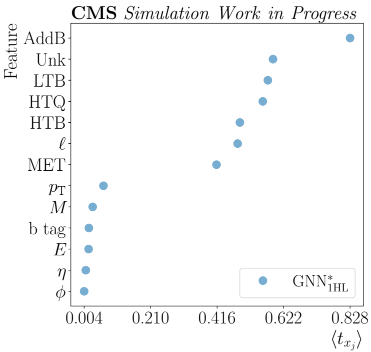
Best Models

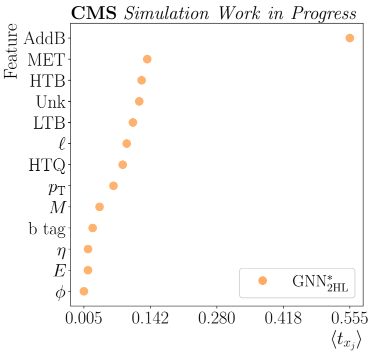
CMS *Simulation Work in Progress*



- Performance of the best GNNs and DNNs are comparable
- Biggest difference in convergence speed and $N_{\text{trainable param.}}$
- Convergence speed appears to be rather independent of $N_{\text{trainable param.}}$

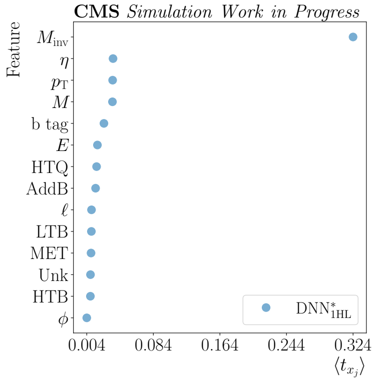
TCA - Best GNNs

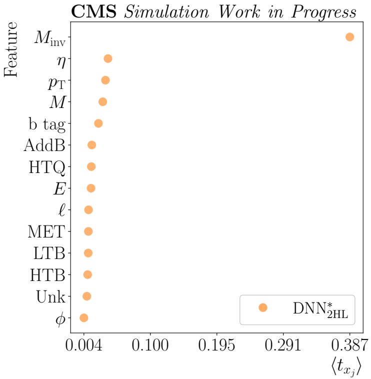




- **Reasonable**:
  - Most important category flag: AddB
  - Most important kinematic feature: p_T
  - Least important feature: ϕ
- **Surprising**: any category flag is more important than any kinematic features

TCA - Best DNNs[2]





- **Surprising**: all category flags are ranked in the lower half
  $\rightarrow$ Possibly because of redundancy
  $\rightarrow$ Already encoded in input vector due to lack of permutation invariance
- **Least important feature**: ϕ
- **Most important feature**: M_{inv}
  $\rightarrow$ Encoded in graph structure
  $\rightarrow$ DNN also learned to look at that

[2] 493 input features $\rightarrow$ 493 Taylor coefficients $\rightarrow$ considered “global” features instead and only considered non-padded features

Analysis Strategy - Comparison B

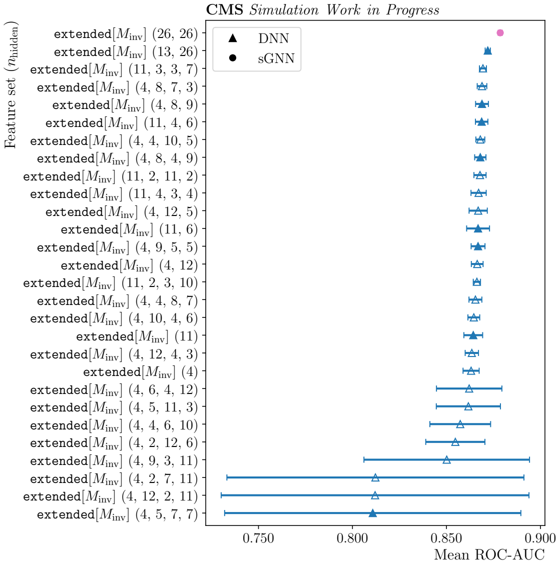
- **Idea**: compare models with similar number of DOF
  - **1.) How well do DNNs perform if their number of DOF is restricted to the number of DOF of $\mathrm{GNN}^{*}_{\mathrm{2HL}}$?**
    - $N_{\mathrm{HL}} = \{1, \ldots, 4\}$
    - $N_{\mathrm{hidden}} = \{5, 6, \ldots, 50\}^{n_{\mathrm{HL}} \in N_{\mathrm{HL}}}$
    - For each HL: consider only the model(s) that are closest to $N^{\mathrm{GNN}^{*}_{\mathrm{2HL}}}_{\text{trainable param.}} = 2107$
  - **2.) How well do GNNs perform if their number of DOF is expanded to the number of DOF of $\mathrm{DNN}^{*}_{\mathrm{2HL}}$?**
    - $N_{\mathrm{HL}} = \{1, \ldots, 4\}$
    - $N_{\mathrm{hidden}} = \{2, 4, \ldots, 12\}^{n_{\mathrm{HL}} \in N_{\mathrm{HL}}}$
    - For each HL: consider only the model(s) that are closest to $N^{\mathrm{DNN}^{*}_{\mathrm{2HL}}}_{\text{trainable param.}} = 6813$, $N^{\mathit{eff},\mathrm{DNN}^{*}_{\mathrm{2HL}}}_{\text{trainable param.}} = 2782$

- **Bonus: Can DNNs outperform $\mathrm{GNN}^{*}_{\mathrm{2HL}}$ if only the number of DOF is tuned?**
  - $N_{\mathrm{HL}} = 3$
  - $N_{\mathrm{hidden}} = \{6, 13, 26\}^{n_{\mathrm{HL}} \in N_{\mathrm{HL}}}$

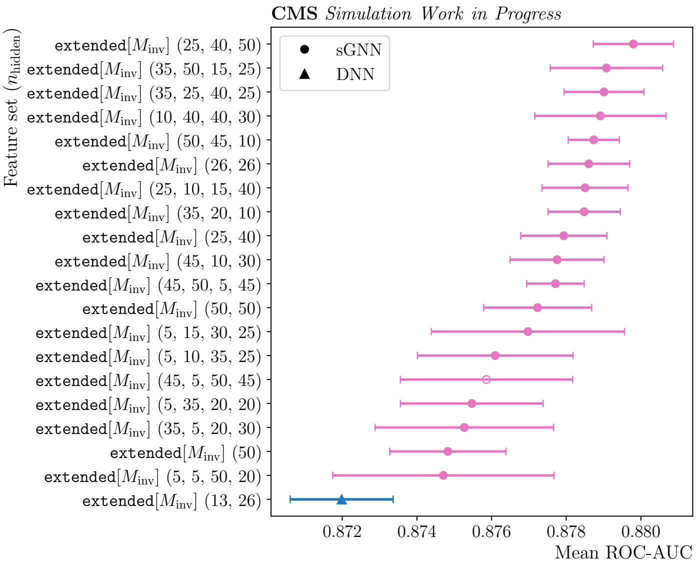
  $\Rightarrow$ Empirically motivated: rather increase number of hidden layers instead of number of hidden nodes

- **Number of compared models**: 27+26+18 = 71

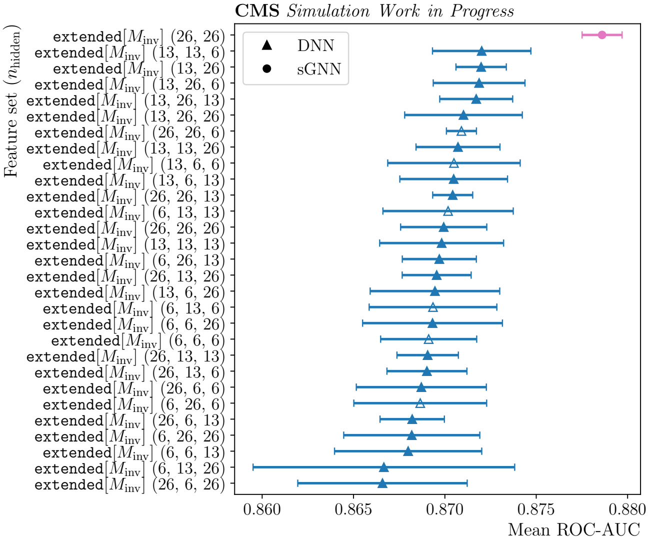
1.) DNNs with a Restricted Number of DOF



2.) GNNs with an Expanded Number of DOF



Bonus: Can DNNs Outperform $\text{GNN}^{*}_{\text{2HL}}$?



Results - Comparison B

- **Question 1.)**
  - Many trainings contain outliers $\longrightarrow$ rather not stable training?
  - The majority of the models are only
    - slightly worse than $\text{DNN}^*_{\text{2HL}}$ and
    - around 1 % worse than $\text{GNN}^*_{\text{2HL}}$ in the best case
- **Question 2.)**
  - Only some expanded GNNs perform better than $\text{GNN}^*_{\text{2HL}}$
  - The best expanded GNN improves the previous best performance by $(0.14 \pm 0.06)\,\%$
- **Bonus: Can DNNs outperform the $\text{GNN}^*_{\text{2HL}}$ if only the number of DOF is tuned?** $\longrightarrow$ No!
  - Having more HLs does not seem to be beneficial
    $\leftrightarrow$ (probably) regularization methods required for models with more HLs
  - DNNs still perform at least $(-0.75 \pm 0.10)\,\%$ worse than $\text{GNN}^*_{\text{2HL}}$