PROCEEDINGS OF SCIENCE

Application of transfer learning to event classification in collider physics

Tomoe Kishimoto,[a,b,c,*] **Masahiro Morinaga,**[b,c] **Masahiko Saito**[b,c] **and Junichi Tanaka**[b,c]

[a] *Computing Research Center, High Energy Accelerator Research Organization, 1-1 Oho, Tsukuba, Japan*

[b] *International Center for Elementary Particle Physics, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan*

[c] *Institute for AI and Beyond, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan*

E-mail: tomoe.kishimoto@kek.jp

An event classification problem in collider physics is a fundamental problem to accelerate searches for new phenomena in nature. The event classification problem is aimed at discriminating the signal events of interest from the background events as much as possible. In this study, the transfer learning technique in deep learning was employed to efficiently address this event classification problem. In collider physics experiments, there are many types of data analyses that target different signal events. To ensure the transferability of these data analyses, a deep learning model based on a graph neural network architecture is proposed in this study. By applying the transfer learning with this model, we observed that a high accuracy of event classification can be achieved even with a small amount of data. In particular, a significant improvement was observed when the physics processes of the events were similar between the source and target datasets of the transfer learning. This achievement by the transfer learning provides a potential approach for saving computing resources for future collider experiments.

International Symposium on Grids & Clouds 2022 (ISGC 2022)
21 - 25 March, 2022
*Online, Academia Sinica Computing Centre (ASGC), Taipei, Taiwan****

*Speaker

PoS(ISGC2022)016

1. Introduction

The aim of experimental particle physics is to understand the fundamental laws of nature. In collider physics experiments, a large number of events[1] are produced from particle collisions using high energy accelerators, such as the Large Hadron Collider (LHC) [1]. Therefore, the classification of events becomes important for data analysis, where interesting signal events are separated from the background events as much as possible.

Although machine learning (ML), such as boosted decision trees, has a long history in collider physics [2, 3], deep learning (DL) is also widely used to enhance the performance of event classification. DL can provide significant discrimination power by utilizing its huge parameter space; however, a large amount of data is required to maximize its performance. In the field of collider physics, training data are typically generated using Monte Carlo (MC) simulations based on theories of signal and background processes. However, MC simulations are computationally expensive. For instance, LHC experiments are expected to require large computing resources for MC simulations in the near future [4]. Therefore, maximizing DL performance with a small amount of data is a key concept to address the scalability and sustainability of future collider physics experiments.

We consider the transfer learning (TL) technique in DL to be a feasible approach to address these issues. The DL model consisted of a stack of layers with nonlinear functions. The initial part of the layers is to learn the local features of data, and the subsequent layers learn global features. This indicates that the knowledge of local features gained while solving one problem can be transferred to different problems that involve common local features. For example, we can assume that the knowledge gained in recognizing cars can be used in recognizing trucks. This is known as the TL technique and it has been successfully applied in the computer vision field, especially for image classification [5, 6]. In collider physics experiments, there are many data analyses targeting various signal events, such as Higgs boson measurements and new phenomena searches. In the present analysis workflow, dedicated DL models for each data analysis are trained from scratch, that is, random initial values, indicating that a large amount of training data is required for each data analysis. If TL works effectively for different data analyses, DL models can be trained using pretrained weight parameters. Consequently, many computing resources for MC simulations and model training are saved.

In this study, we report that event classification can be performed with high accuracy, even with a small amount of data, by applying the TL technique. Event classification is typically performed based on the information of reconstructed particles (objects). The number of objects in the final state differs depending on the data analysis. Thus, the DL model must work with a variable number of objects and be insensitive to the ordering of objects to ensure transferability. To overcome these problems, we propose a DL model based on a graph neural network (GNN) architecture. The details of the DL model and improvements by TL are discussed below.

The remainder of this paper is organized as follows. Section 2 describes related works. Section 3 summarizes the datasets used in this study. Section 4 provides details of the proposed model. Section 5 presents the experimental results. Finally, Section 6 concludes the paper.

[1]The term "event" corresponds to "image" in the image classification.

2. Related work

DL has been successfully adapted for event classification in collider physics. A previous study reported that DL outperformed traditional MLs by discovering powerful features and providing better discrimination power [7]. As discussed in the previous section, these DL models are trained from scratch and optimized for each problem. To use a single DL model for closely related problems, such as event classification at different signal masses, a parameterized neural network for high energy physics was proposed [8] , which includes physics parameters as inputs. For example, by including the mass of a signal particle, the single DL model provides improved discrimination power across different signal masses. In addition, a study related to the transferability of DL models in different signal events [9] reported that DL provides discriminative power to other signals that vary kinematically. In contrast to these previous studies, our study has the following novelties:

- A DL model based on the GNN architecture is proposed. This model allows us to examine the transferability for different event classifications. For example, the proposed model ensures the transferability between event classification problems with different number of observed objects. This transferability is not directly covered by the parameterized neural network.

- Fine-tuning is performed using a small amount of data. In this study, we transfer a part of the weight parameters of the model and update the weight parameters using a given data. In the event classification problem, this fine-tuning is expected to work effectively to absorb differences in physics processes between the source and target datasets. To the best of our knoledge, effectiveness of the fine-tuning is not discussed in any prior works.

3. Datasets

The training data in this study were produced using particle physics simulations: proton-proton collision events were generated by MadGraph5_aMC@NLO [10] at a center of mass energy of 13 TeV, with showering and hadronization performed by Pythia8 [11] and detector response simulated by Delphes [12].

In this study, the dataset used to learn features with a large amount of data is called a *source* dataset and datasets used to evaluate the performance of TL with a small amount of data are called *target* datasets. The physics processes of the signal and background events in each dataset are designed to have the same final state particles, making it difficult to discriminate. Table 1 summarizes the physics processes for each dataset and the descriptions are as follows:

- Source dataset: Two-Higgs-Doublet model (2HDM) [13, 14], which introduces additional Higgs bosons, H^0, A and $H^\pm$, is used as the signal event. Top pair production ($t\bar{t}$) of the Standard Model (SM) is used as the background event. Figure 1 shows Feynman diagrams for the signal and background processes. The final state particles are one lepton (ℓ), one neutrino (ν), two b-quarks (b), and two light-quarks (j).

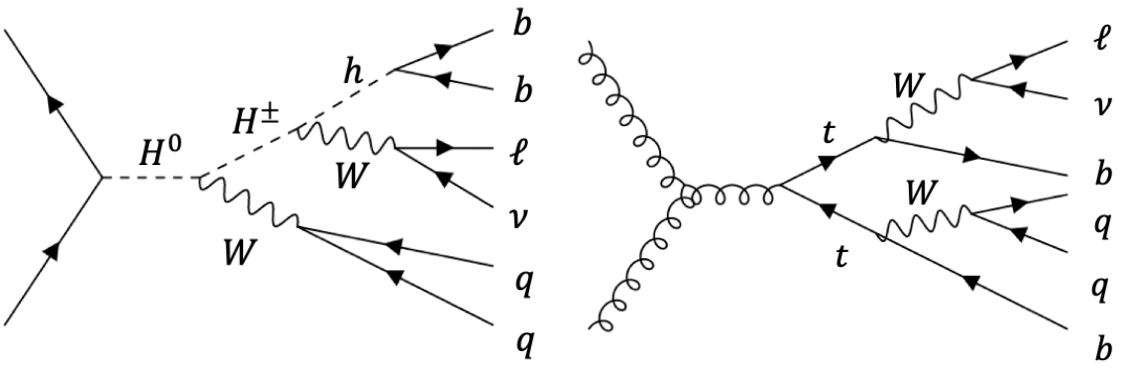
- Target dataset 1: The same 2HDM process is used as the signal event; however, the masses of the additional Higgs bosons are different from the source dataset, which are heavier than

Table 1: Summary of physics processes.

Category	Bkg.	Sig.	Signal mass (GeV)	Final state	# of variables
Source dataset	$t\bar{t}$	$2HDM$	$m_{H^0}, m_{H^\pm}$ = 425, 325	$\ell\nu bbjj$	5×6
Target dataset 1	$t\bar{t}$	$2HDM$	$m_{H^0}, m_{H^\pm}$ = 500, 400	$\ell\nu bbjj$	5×6
Target dataset 2	$t\bar{t}$	Z'	$m_{Z'}$ = 1000	$\ell\nu bbjj$	5×6
Target dataset 3	$ttbb$	ttH	Standard model	$\ell\nu bbbbjj$	5×8
Target dataset 4	$Z\nu\nu$	$\tilde{g}\tilde{g}$	$m_{\tilde{g}}$ = 607	$\nu(\tilde{\chi}_1^0)\nu(\tilde{\chi}_1^0)jjjj$	5×5

the source dataset. The background events are the same $t\bar{t}$ process as the source dataset. The decay chains are the same as the source dataset.

- Target dataset 2: A heavy neutral particle (Z') decaying into top quark pair [15, 16] is used as the signal event. The background events are the same $t\bar{t}$ process as the source dataset. The decay chains of the top pairs are the same between signal and background events as shown in Figure 1 (b).
- Target dataset 3: Higgs boson production associated with top pair (ttH) in the SM, where Higgs boson decays into b-quark pair, is used as the signal event. Top pair production in association with two b-quark ($ttbb$) is used as the background event. In this dataset, additional two b-quarks are observed in the final state compared to the source dataset.
- Target dataset 4: gluino pair production ($\tilde{g}\tilde{g}$) in supersymmetry model [17], where a gluino decays into a neutralino ($\tilde{\chi}_1^0$) and two light-quarks, is used as the signal events. Z boson production in association with four light-quarks, where Z boson decays into neutrino pair ($Z\nu\nu$), is used as the background event. Thus, the final state particles are two neutralinos and four light-quarks for the signal events and two neutrinos and four light-quarks for the background events, respectively.



(a) **(b)**

Figure 1: Feynman diagrams for (a) 2HDM signal and (b) $t\bar{t}$ background processes in the source dataset.

Neutrinos were reconstructed as a missing transverse momentum (MET) object. b-quark and light-quark were reconstructed as b-jet and light-jet objects, respectively. The four-momenta of

Table 2: The number of generated MC events.

Category	Training data		Validation data		Test data	
	Sig.	Bkg.	Sig.	Bkg.	Sig.	Bkg.
Source dataset	1.0×10^7	1.0×10^7	5.0×10^4	5.0×10^4	5.0×10^4	5.0×10^4
Each target dataset	5.0×10^5	5.0×10^5	5.0×10^4	5.0×10^4	5.0×10^4	5.0×10^4

each object (p_{T}, η, ϕ, mass) and object-type were used as input variables in this study[2], where the object type is indicated by a integer. The log transformation was applied to p_{T} and mass (GeV) to fit the values withion a reasonable range. $\eta = 0$ and mass = 0 were assigned to the MET object. The number of input variables for each dataset is also summarized in Table 1. For example, the source dataset was 5 (feature variables) × 6 (objects) = 30. Table 2 summarizes the number of generated MC events for each category. According to the concept of the TL, a large amount of the training data was prepared for the source dataset, which is a total 2.0×10^7 events and smaller training data were prepared for each target dataset, which is a total 1.0×10^6 events. The training, validation and test data were used as independent data to evaluate the performance without bias.

4. Proposed deep learning model

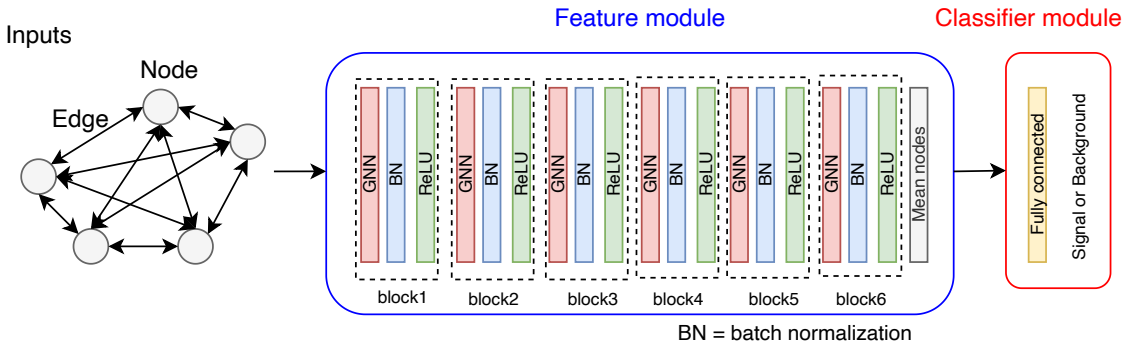


Figure 2: Overview of the proposed DL model.

Figure 2 presents an overview of the proposed DL model. The input data were prepared to have a graph structure composed of *nodes* and *edges*. Each node in the graph corresponds to an object in this study. Thus, object features: p_{T}, η, ϕ, mass, and object-type are assigned to the node attribute. Edges are prepared as a fully-connected bidirectional graph, including a self-loop. Edge attributes are empty in the initial inputs. Thus, the shape of the initial input is $N \times C$, where N is the number of objects and C is the number of features.

The model consisted of two parts: a feature and a classifier module. The feature module is aimed at extracting data features. The output of the feature module was fed into the classifier module to predict the data class, that is, a signal or background event. A GNN architecture was employed in the feature module. The GNN allows us to handle variable number of objects and overcome

[2]Collider physics terminologies are described in Ref. [18].

Table 3: The number of trainable parameters. “w/o attention mechanism" indicates to use a simple message passing described in the text, and “w/ attention mechanism" indicates to use the graph attention network.

Model	Feature module	Classifier module	Total
w/o attention mechanism (GNN model)	333,312	514	333,826
w/ attention mechanism (GAT model)	334,848	514	335,362

the ordering problem by using permutation-invariant reduction functions, such as element-wise sums, means, or maximums. In this study, the message passing paradigm in the GNN consisted of edge-wise and node-wise computations. In edge-wise computation, a message from each edge is generated based on the edge attributes and neighboring node attributes. In node-wise computation, each node attribute is updated by aggregating its incoming messages using the reduction function. As a simple message passing of the GNN, we implemented a node-wise computation that averages the incoming messages, which are the attributes of the source nodes. We also examined the graph attention network (GAT) [19] based on the attention mechanism [20], which learns the weights of node relations as edge attributes. For instance, if the relation between b-quarks from Higgs boson is more important than other relations, the attention weight becomes a large value. Then, the corresponding messages are weighted by these values in message passing. The multi-head technique in the attention mechanism is also employed to enhance the performance. The attention mechanism is expected to extract features effectively and also is useful for visualizing the importance of objects.

The feature module consists of a stack of blocks, as illustrated in Figure 2, where there are 6 blocks. Each block consists of a GNN layer, batch normalization (BN) layer [21], and ReLU activation function. In the final layer of the feature module, a global feature is obtained by averaging the node attributes. The classifier module consisted of only one fully connected layer. Table 3 summarizes the number of trainable parameters with and without the attention mechanism. The attention mechanism introduces additional trainable parameters compared with the simple message passing in this study. The models with and without the attention mechanism are referred to as GNN model and GAT model, respectively.

5. Experiments

Our codes for this experiment were implemented using PyTorch [22] and DGL [23] and are available [24]. NVIDIA Tesla A100 was used for all executions.

5.1 Training of source task

Table 4 summarizes the hyperparameters of training for the source dataset. The training was performed for up to 100 epochs and the best epoch for the validation data was used as the final weight parameters. The hyperparameters of the model architecture, such as the number of hidden features in the GNN layer were determined by a grid search for several points. The numbers in brackets in Table 4 indicate the searched parameter points and the bold numbers are selected as the final parameters.

Figure 3 shows the accuracy of event classification during training, which confirm that the training and validation data exhibit similar performance. Thus, the overtraining is suppressed

Table 4: The hyperparameters of the source dataset training. The learning rate is decreased by cosine annealing algorithm [25]. In the model architecture, the bold numbers are selected by grid search. The number of hidden features is the length of each node attribute in the GNN layer. All blocks have the same number of hidden features and multi-head.

# of epoch	100
Batch size	2,048
Loss function	Cross entropy loss
Optimizer	SGD algorithm
learning rate	0.01 - 0.0001
momentum	0.9
weight_decay	0.00005
Model architecture	
# of blocks	[5, **6**, 7, 8]
# of hidden features	[128, **256**, 512, 1024]
# of multi-head	[2, **4**, 8, 16]

owing to the large amount of data. The GAT model exhibits slightly better performance than the GNN model. This may explain why the attention mechanism works effectively for extracting data features. The observed accuracies for the test data are 0.810 and 0.819 for the GNN and GAT models, respectively. The training throughput of one GPU for the GNN and GAT models was approximately 66 batches and 38 batches per second.

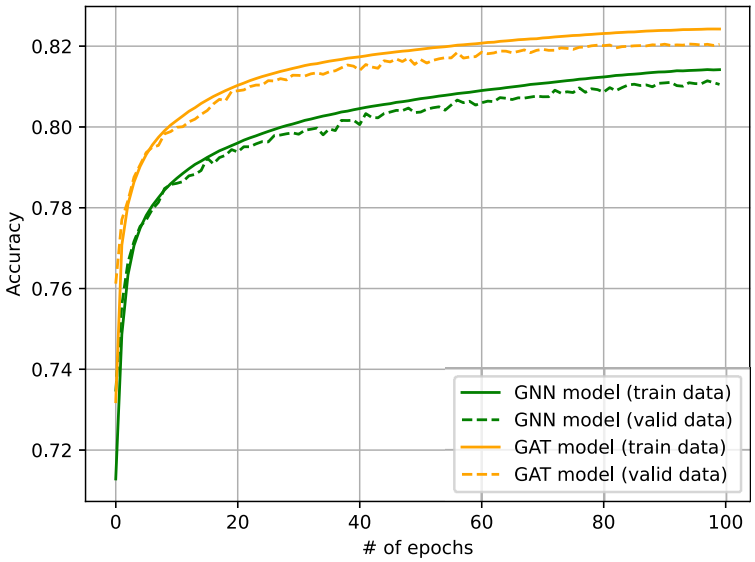


Figure 3: Accuracies over epochs in the source dataset.

Figure 4 shows the attention weights of the GAT model after the training. The values are averaged over all events in the test data and also the multi-heads. It can be seen that the nodes

corresponding to the b-jets show higher values. In the signal process, the b-quarks originate from Higgs boson. Thus, the existence of Higgs boson is an important discriminant to separate the signal events from the background events. The observation that the b-jets show large weights is consistent with our analysis knowledge.

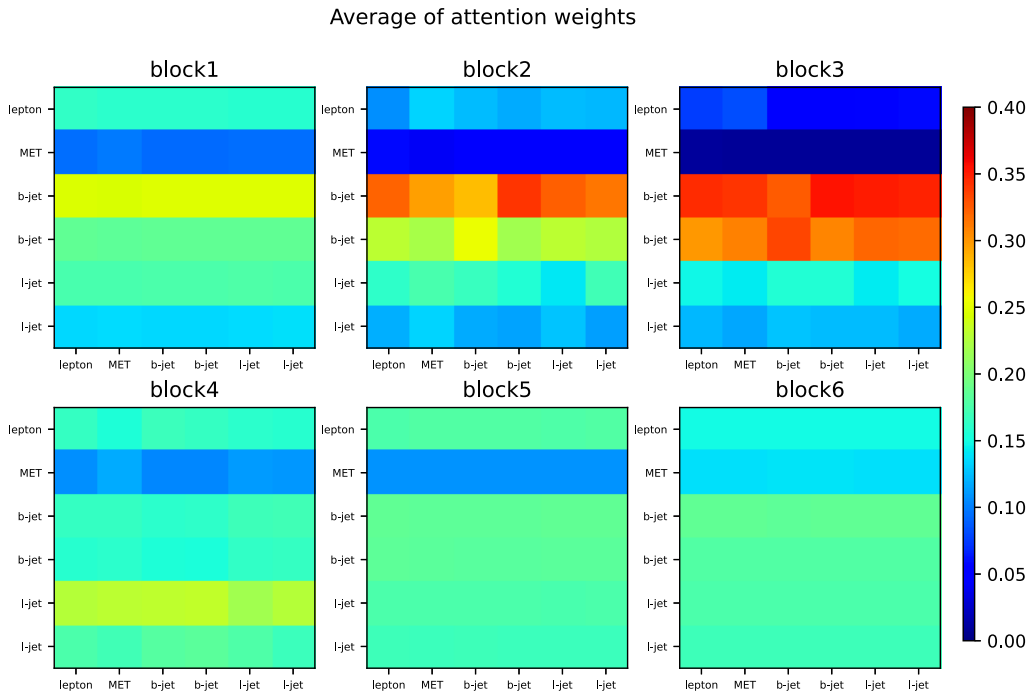


Figure 4: Average of attention weights in the source dataset. The test data is used. Y-axis and X-axis are the source and target nodes in the message-passing. The b-jets and light-jets are ordered by p_{T}.

5.2 Training of target tasks

To evaluate transferability, three types of training were defined for the target datasets:

- No TL: No transfer learning is applied to the target datasets. All weight parameters of the model are trained from scratch using a given target dataset.
- TL with fixed weights: The weight parameters of the feature module are transferred from the source model, but the transferred weight parameters are fixed during the training using a given target dataset. The weight parameters of the classifier module are trained from scratch.
- TL with fine-tuning: The weight parameters of the feature module are transferred from the source model, and the transferred weight parameters are updated during the training using a given target dataset. The weight parameters of the classifier module are trained from scratch.

In all three trainings, the classifier module is always trained from scratch because we assume that common knowledge is extracted by the feature module and the classifier module highly depends on

the target domains. The hyperparameters of the trainings were the same as those of the source task; however the batch size was decreased to 256 to increase the parameter update frequency for small datasets. To evaluate the performance in terms of the number of target events, cases of 10^3, 10^4, 10^5, and 10^6 events were examined, which were quite small compared to the source dataset.

5.3 Results

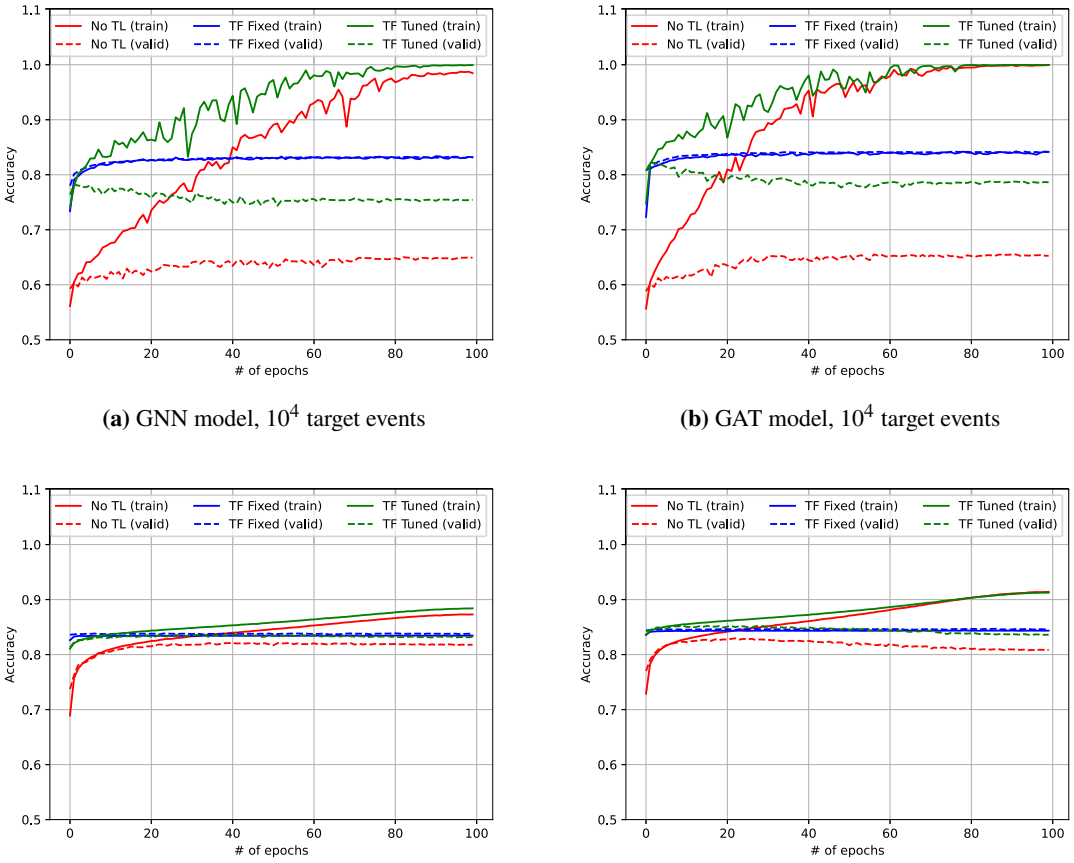
Figure 5 shows the accuracies for the target dataset training with the cases of 10^4 and 10^6 target events. Target dataset 1 is used in this example, which has a very similar topology to the source dataset because only the masses of the signal particles are different. If we focus on Figures 5 (a) and (b) for the case of relatively small data, significant improvements are observed through TL, which are approximately 20% improvements in accuracy. In such small data, the TL with fine-tuning (green lines) still causes over-training, and the TL with fixed weights (blue lines) shows better performance in these examples. However, if sufficient data are available, the TL with fine-tuning shows a similar or better performance compared to the fixed weights, as confirmed in Figures 5 (c) and (d).

Figure 6 summarizes the observed accuracies of the test data in terms of the number of target events for all the target datasets. TL with fine-tuning (solid lines) provides improvements compared with no TL (dotted lines) for all datasets. To evaluate the relative improvements by TL, error reduction was defined as follows:

$$\text{Error reduction} = 1 - \frac{\text{Error}^{\text{TL}}}{\text{Error}^{\text{No TL}}}, \ \text{Error} = 1 - \text{Accuracy}, \tag{1}$$

where the positive value of the error reduction indicates improvements through TL, whereas a negative value indicates degradation through TL. Figure 7 summarizes the error reductions for all the target datasets. As described above, significant error reductions by TL were observed in the target dataset of 1. The error reductions in target dataset 2 are also visible. This is probably because the same background process as of the source dataset, $t\bar{t}$, was used in this target dataset. However, the error reductions decrease if the topologies are different from the source dataset, such as the target datasets 3 and 4. These behaviors are consistent with our initial expectation that TL works effectively if common knowledge exists between the source and target domains. However, we still observe more than a 10% error reduction in target datasets 3 and 4 when the target data size is very small. The TL with fine-tuning (dashed lines) shows similar performance to the No TL when the number of events is sufficient, although the TL with fixed weights (solid lines) shows a degraded performance.

Figure 8 shows the attention weights of the GAT model in target dataset 4 before and after performing fine-tuning as an example. The definition of average values is the same as that shown in Figure 4. The light-jets show higher values before fine-tuning, and the MET shows higher values after fine-tuning. This indicates that fine-tuning increases the importance of MET in this dataset. In dataset 4, the neutralinos of the signal events were observed as the MET object. Thus, this behavior is reasonable based on our analysis knowledge.



(a) GNN model, 10^4 target events

(b) GAT model, 10^4 target events

(c) GNN model, 10^6 target events

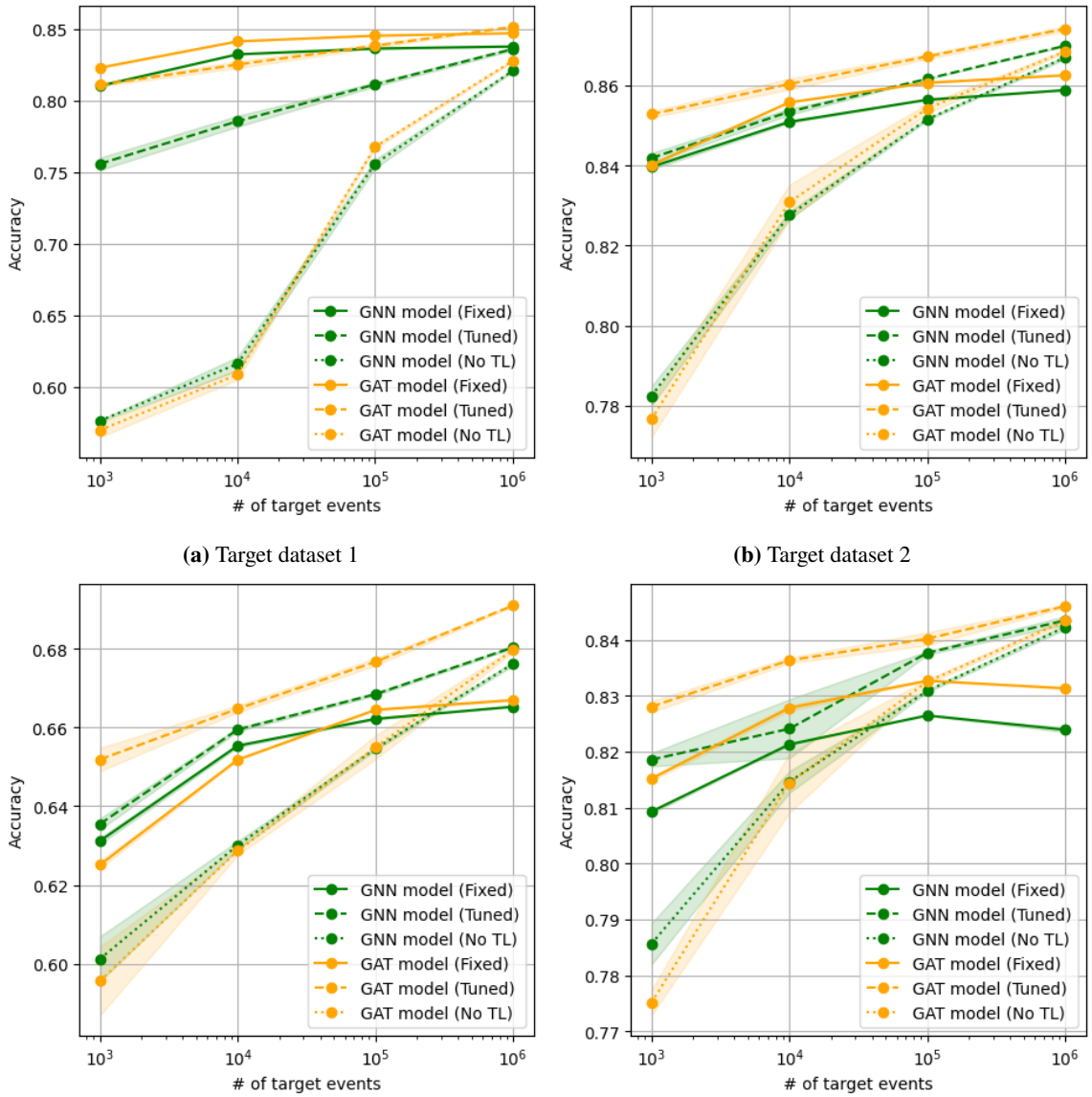
(d) GAT model, 10^6 target events

Figure 5: Accuracies for different models and the number of target events. The target dataset 1 is used. "No TL" indicates that no transfer learning is applied. TL Fixed and TL Tuned indicate the transfer leaning is applied with the fixed weight parameters and fine-tuning, respectively.

6. Conclusion

In this study, the transferability of event classification in collider physics is discussed. We proposed a DL model based on a GNN architecture to handle a variable number of objects and overcome the ordering problem. We confirmed that high accuracy of the event classifications was achieved even with a small amount of data by applying the transfer learning. If the topologies are different between the source and target datasets, fine-tuning is very effective in absorbing the differences, which is confirmed by visualizing the attentions weights. If a large amount of data is available, the performance converges with and without the transfer learning.

In this study, the source dataset consisted of only two specific processes: 2HDM and $t\bar{t}$. To improve the generalization of transferability, including more physics processes in the source dataset is a feasible approach. A deeper understanding of the behavior of attention weights will improve the clarity of transferability, which is a subject for future research.



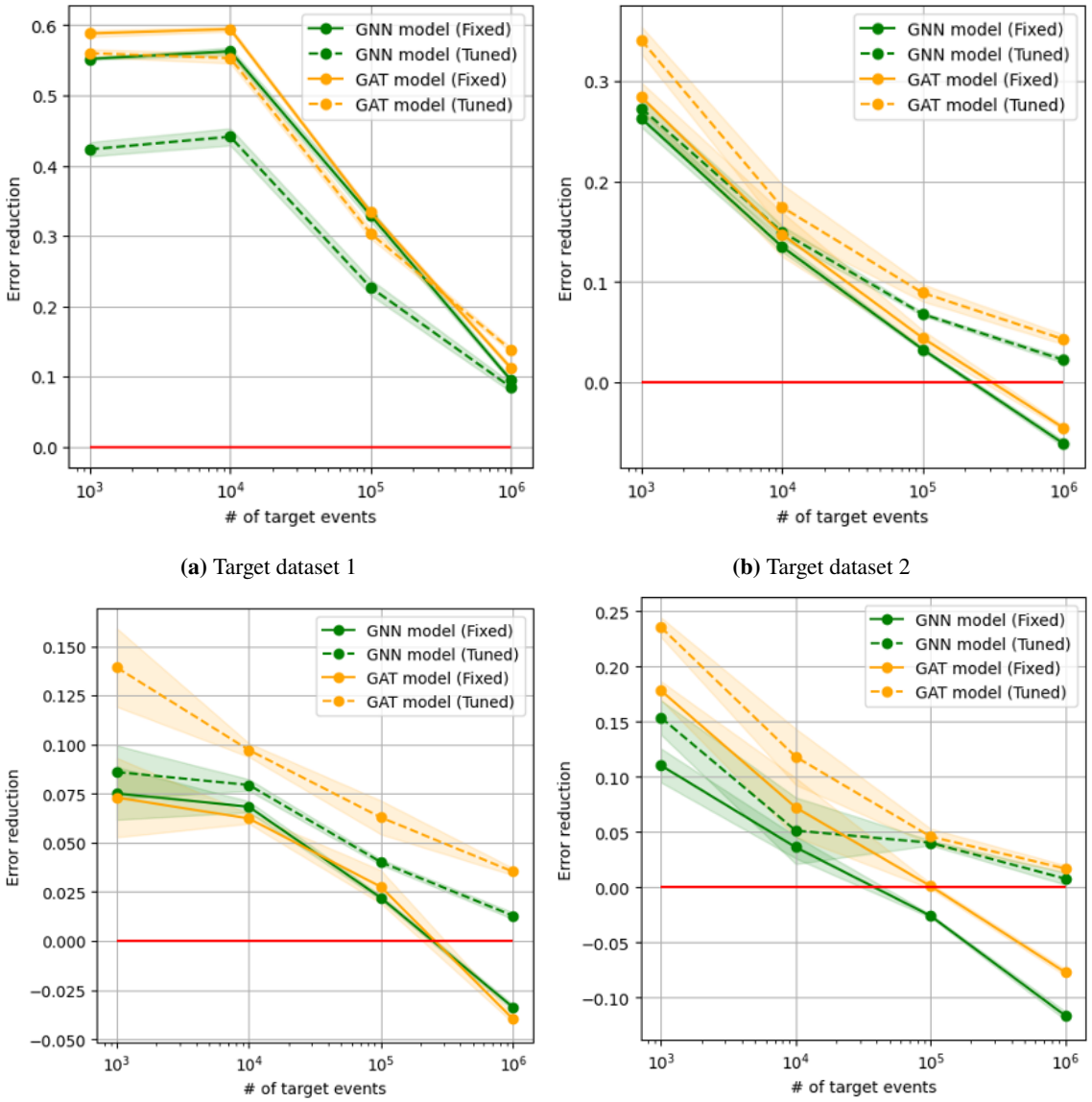
(a) Target dataset 1

(b) Target dataset 2

(c) Target dataset 3

(d) Target dataset 4

Figure 6: The accuracies for different datasets. "No TL" indicates that no transfer learning is applied. "Fixed" and "Tuned" indicate the transfer leaning with the fixed weight parameters and fine-tuning, respectively. Each point obtained using the test data and shows average value of three runs with one standard deviation band.



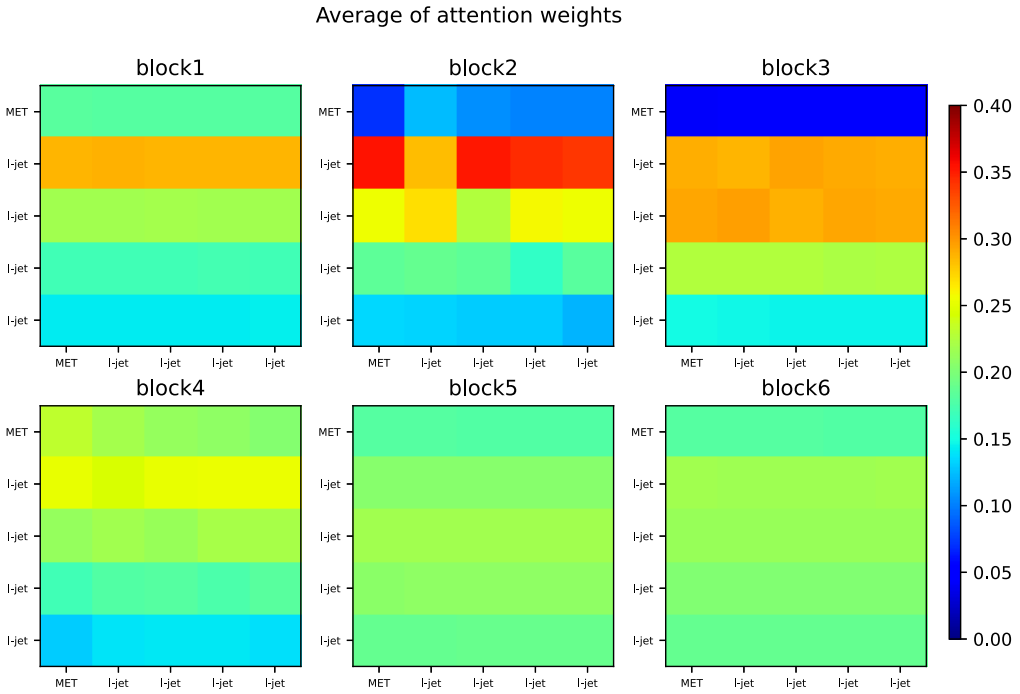
(a) Target dataset 1

(b) Target dataset 2

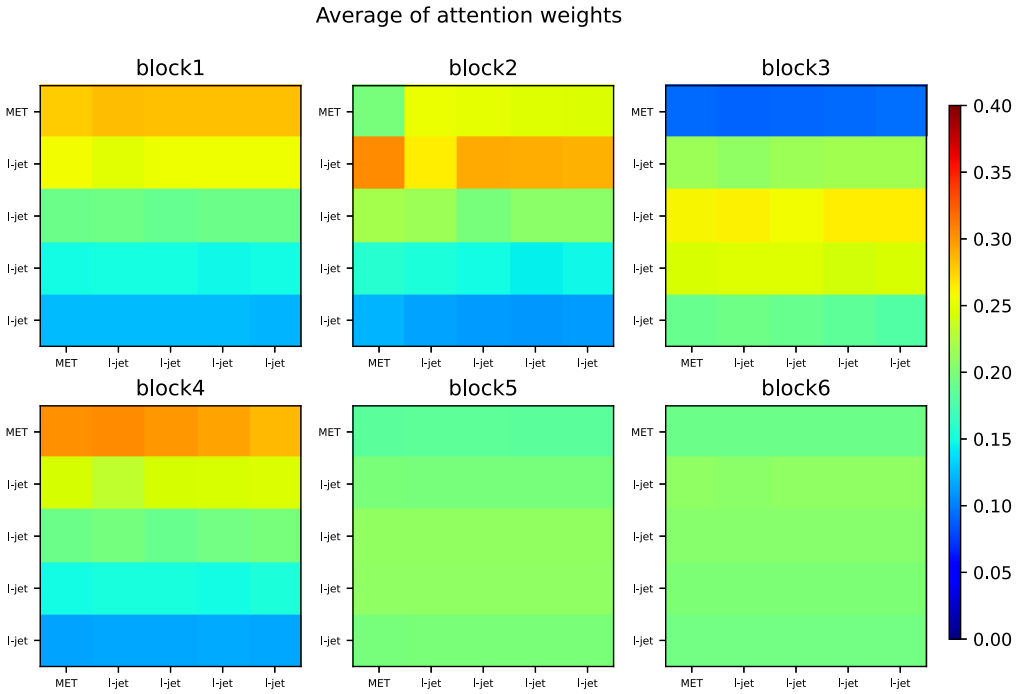
(c) Target dataset 3

(d) Target dataset 4

Figure 7: The error reductions for different datasets. “Fixed" and “Tuned" indicate the transfer learning with the fixed weight parameters and fine-tuning, respectively. Each point obtained is using the test data, and shows average value of three runs with one standard deviation band.



(a) Before the fine-tuning



(b) After the fine-tuning

Figure 8: Average of attention weights in the target dataset 4. The light-jets are ordered by p_{T} in these figures. (a) is the figure before performing the fine-tuning and (b) is the figure after performing the fine-tuning using 10^6 target events.

References

[1] L. Evans and P. Bryant, *LHC machine*, *Journal of Instrumentation* **3** (2008) S08001.

[2] P.A. Kim Albertsson and D.A. *et al*, *Machine learning in high energy physics community white paper*, *Journal of Physics: Conference Series* **1085** (2018) 022008.

[3] A. Radovic, M. Williams, D. Rousseau, M. Kagan, D. Bonacorsi, A. Himmel et al., *Machine learning at the energy and intensity frontiers of particle physics*, *Nature* **560** (2018) 41.

[4] ATLAS Collaboration, *ATLAS Software and Computing HL-LHC Roadmap*, Mar, 2022.

[5] M. Hussain, J. Bird and D. Faria, *A Study on CNN Transfer Learning for Image Classification*, 06, 2018.

[6] W. Zhu, B. Braun, L.H. Chiang and J.A. Romagnoli, *Investigation of transfer learning for image classification and impact on training sample size*, *Chemometrics and Intelligent Laboratory Systems* **211** (2021) 104269.

[7] P. Baldi, P. Sadowski and D. Whiteson, *Searching for Exotic Particles in High-Energy Physics with Deep Learning*, *Nature Commun.* **5** (2014) 4308 [1402.4735].

[8] P. Baldi, K. Cranmer, T. Faucett, P. Sadowski and D. Whiteson, *Parameterized neural networks for high-energy physics*, *The European Physical Journal C* **76** (2016) .

[9] M.C. Romão, N.F. Castro, R. Pedro and T. Vale, *Transferability of deep learning models in searches for new physics at colliders*, *Phys. Rev. D* **101** (2020) 035042.

[10] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer et al., *The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations*, *JHEP* **07** (2014) 079 [1405.0301].

[11] T. Sjöstrand, S. Ask, J.R. Christiansen, R. Corke, N. Desai, P. Ilten et al., *An introduction to PYTHIA 8.2*, *Comput. Phys. Commun.* **191** (2015) 159 [1410.3012].

[12] M. Selvaggi, *DELPHES 3: A modular framework for fast-simulation of generic collider experiments*, *Journal of Physics: Conference Series* **523** (2014) 012033.

[13] G. Branco, P. Ferreira, L. Lavoura, M. Rebelo, M. Sher and J.P. Silva, *Theory and phenomenology of two-higgs-doublet models*, *Physics Reports* **516** (2012) 1.

[14] C. Degrande, *Automatic evaluation of UV and R2 terms for beyond the Standard Model lagrangians: A proof-of-principle*, *Computer Physics Communications* **197** (2014) .

[15] G. Altarelli, B. Mele and M. Ruiz-Altaba, *Searching for New Heavy Vector Bosons in $p\bar{p}$ Colliders*, *Z. Phys. C* **45** (1989) 109.

[16] B. Fuks and R. Ruiz, *A comprehensive framework for studying W′ and Z′ bosons at hadron colliders with automated jet veto resummation*, *Journal of High Energy Physics* **2017** (2017) 1.

[17] S.P. Martin, *A Supersymmetry primer*, *Adv. Ser. Direct. High Energy Phys.* **18** (1998) 1 [hep-ph/9709356].

[18] The ATLAS Collaboration, *The ATLAS experiment at the CERN large hadron collider*, *Journal of Instrumentation* **3** (2008) S08003.

[19] S. Brody, U. Alon and E. Yahav, *How Attentive are Graph Attention Networks?* , in *International Conference on Learning Representations*, 2022, https://openreview.net/forum?id=F72ximsx7C1.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez et al., *Attention is all you need*, in *Advances in Neural Information Processing Systems*, I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan et al., eds., vol. 30, Curran Associates, Inc., 2017, https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[21] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, p. 448–456, JMLR.org, 2015.

[22] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan et al., *Pytorch: An imperative style, high-performance deep learning library*, in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett, eds., pp. 8024–8035, Curran Associates, Inc. (2019), http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[23] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song et al., *Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks*, *arXiv preprint arXiv:1909.01315* (2019) .

[24] "heptrans." Available: https://github.com/ktomoe/heptrans, 2022.

[25] I. Loshchilov and F. Hutter, *SGDR: Stochastic Gradient Descent with Warm Restarts*, 2016. 10.48550/ARXIV.1608.03983.