Semi-supervised Graph Neural Network for Particle-level Noise Removal

Tianchun Li[§], **Shikun Liu**[§], **Pan Li**
Department of Computer Science
Purdue University
{li2657, liu2112, panli}@purdue.edu

Miaoyuan Liu
Department of Physics and Astronomy
Purdue University
liu3173@purdue.edu

Yongbin Feng[§], **Nhan V. Tran**
Department of Particle Physics
Fermi National Accelerator Laboratory
{yfeng, ntran}@fnal.gov

Abstract

The high instantaneous luminosity of the CERN Large Hadron Collider leads to multiple proton-proton interactions in the same or nearby bunch crossings (pileup). Advanced pileup mitigation algorithms are designed to remove this pileup particle noise and improve the performance of physics observables crucial to the science goals. This study applies the semi-supervised graph neural network to particle-level pileup noise removal, by identifying the particles produced from pileup. The graph neural network is trained on charged particles with well-known labels, which can be obtained from simulation truth information or measurements from data, and inferred on neutral particles of which such labeling is missing. This semi-supervised approach does not depend on the ground truth information from simulation and thus allows us to perform training directly on real data. The performance with this approach is found to be consistently better than widely-used domain algorithms and comparable to a fully supervised training approach. The study serves as the first attempt at applying semi-supervised learning on pileup mitigation, and opens up a new direction of fully data-driven pileup mitigation techniques.

1 Introduction

The high instantaneous luminosity of the CERN Large Hadron Collider (LHC) enables studies of the deep mysteries of our universe, such as the nature of the Higgs boson and dark matter as well as the origin of the matter-antimatter asymmetry. The enormous amount of data coming from increasing noisy particle collisions, recorded by increasingly complex detectors, pose various challenges to the data collection and analysis. Multiple collisions in the same or nearby proton beam crossings lead to overlapping particle interactions is referred to as pileup (PU). To achieve the desired physics sensitivity with the LHC data, the PU noise needs to be identified and mitigated effectively in order to identify signals of interest, i.e., those from the primary interaction associated with the leading vertex (LV). Various PU mitigation techniques have been developed over the past decade to tackle this important yet challenging task.

The current widely-used domain algorithms for PU mitigation include SoftKiller (1) and PUPPI (2), both of which depend on manually designed heuristics. More recently, deep learning (DL) techniques (3; 4; 5; 6) have been applied to extract particle features for the noise removal task. By

[§]The three authors contributed equally to this paper.

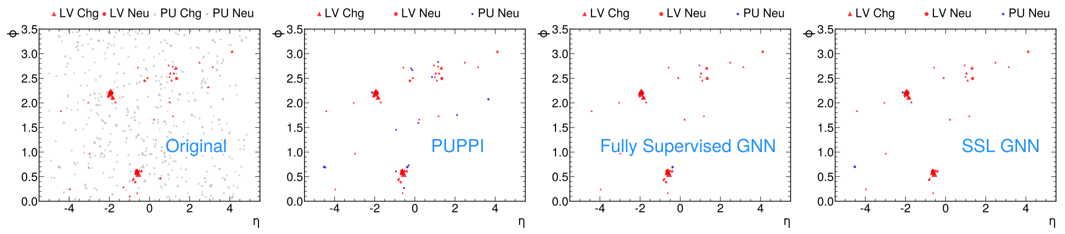


Figure 1: Visualization of the particle distributions in the $\eta - \phi$ space of one event. The four plots from left to right are with all particles, after PUPPI cleaning, after the GNN trained fully supervised, and after the GNN trained semi-supervised.

exploring a higher dimensional phase space of particle-level and event-level features with different network architectures, these DL algorithms are shown to perform better than the classical domain ones. However, these algorithms are trained with fast simulation datasets, fully supervised with the knowledge of the LV (signal) and PU (noise) labels as the ground truth information. This approach has two primary shortfalls: (1) in more realistic simulations and real data, the label information is harder or impossible to retrieve and define and (2) accurate modeling of the pileup noise interactions are difficult to obtain, and any discrepancies between simulation and data can lead to additional biases or performance degradation in downstream physics observable (7; 8; 9).

In this study, we explore a novel approach of applying a semi-supervised machine learning technique (SSL) in pileup mitigation, taking advantage of the fact that, although LV/PU labels are not available for all particles, they can be precisely determined for the charged subset of the particles. A graph neural network (GNN) is firstly trained on labeled charged particles, with the input features from both labeled charged particles and unlabeled neutral particles. The trained GNN is then inferred on neutral particles to estimate the probability of each being produced from LV. A randomized feature-masking technique is developed to handle the features that are not shared by charged and neutral particles. In contrast to the expert-level fixed form heuristics in SoftKiller and PUPPI, the GNN-based approach allows a more powerful, flexible, and expressive formula that is purely data-driven with parameters directly learned from data.

The effectiveness of the SSL approach is studied and confirmed by comparing its performance against methods with a fully-supervised GNN with the same architecture and the expert-level heuristics. We conduct our studies in different experimental conditions with different PU/noise levels, and find no significant performance drop between the fully supervised training and semi-supervised training. In all cases, the GNNs achieve better performance than the expert-level heuristics. Figure 1 is an example of one event visualization that shows GNN methods remove PU neutral particles more effectively than PUPPI. The SSL approach has no dependence on any ground truth information from simulations, and can be applied directly to real collision data, avoiding data-simulation differences. Our initial results are promising and encouraging, and we hope to adopt and validate this approach in the real LHC collision data in the near future.

2 Formulating Pileup Mitigation as a Semi-supervised Learning Problem

Semi-supervised learning (SSL) is a machine learning technique that takes advantage of not only the training samples but also the features of testing samples to train a model (10). The pileup mitigation problem can be formulated as an SSL problem by utilizing features of both charged particles (with labels, for training) and neutral ones (without labels, for testing) given their geometric relationship. For each proton bunch crossing, multiple charged and neutral particles are produced and scattered in the pseudorapidity and azimuthal angle $\eta - \phi$ space. Because of the interaction dynamics, particles from the LV (signal label) tend to be more localized. Therefore, an effective learning procedure of a model should not only depend on the self features of a particle, but also the features of its neighboring charged and neutral particles.

To leverage neighboring particle features, one graph is constructed per proton-bunch-crossing event to establish the relations between particles and their neighbors: Particles are viewed as nodes and two particles are linked if their distance in the $\eta - \phi$ space, $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$, is smaller than a threshold. $\Delta R = 0.8$ is chosen for the experiments in this study.In practice, multiple events are collected and processed, and each of them forms a graph.

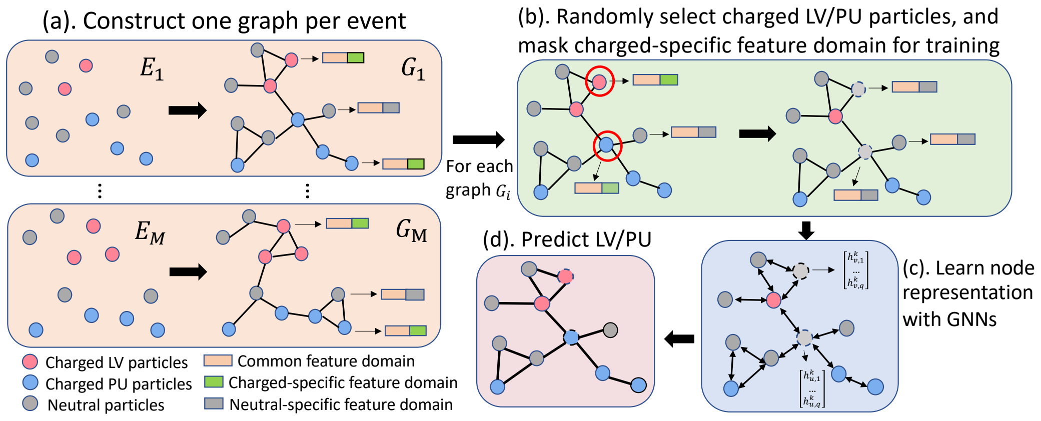

(a). Construct one graph per event
E_1
G_1
E_M
G_M
(b). Randomly select charged LV/PU particles, and mask charged-specific feature domain for training
For each graph G_i
(c). Learn node representation with GNNs
(d). Predict LV/PU
Charged LV particles
Charged PU particles
Neutral particles
Common feature domain
Charged-specific feature domain
Neutral-specific feature domain

Figure 2: A diagram illustrating the SSL model training flow.

There is a particular challenge in pileup mitigation as opposed to traditional SSL problems, where the training samples (charged particles) and testing samples (neutral particles) may have their specific features, such as the encoding of particle types. These features can not be naively removed as they provide useful information to determine neighbors' labels. To ensure the model trained on charged particles can be well applied to neutral particles, a dedicated training strategy is needed.

Formally the problem setting is as follows. There are M graphs $\{G_1, \cdots, G_M\}$. Each observed graph contains two types of nodes: charged particles V_C and neutral particles V_N. For each charged particle $v \in V_C$, both the features x_v and its label y_v are observed, while for a neutral particle $u \in V_N$, only its features x_u are observed. Our goal is to infer the labels of neutral particles. Each node feature can be categorized into two parts, where the first part contains feature values shared by charged and neutral particles, while the rest contain the charged-or-neutral-specific feature values.

3 The Proposed Model

Training the model consists of four steps as illustrated in Figure 2. Step (a) graph construction has been introduced. Next, we focus on steps (b)-(d).

Step (b) Random Selection and Feature Masking. The main goal of step (b) is to deal with the potential features not shared between charged and neutral particles. These features in our simulation datasets are LV-or-PU label information where only charged particles hold. In real experiments, they may include other features such as particle types as charged and neutral particles by nature are in different categories. We term these features as charged-specific features and neutral-specific features respectively. To prevent the model from overfitting such features, in each graph each training epoch, a certain portion of charged LV and PU particles are randomly selected as training samples. The charged-specific features of these particles will be replaced by neutral-specific features of randomly selected neutral particles. Note that such a portion should neither be too large nor too small, as otherwise, either the neighboring features of a particle get corrupted, or too many epochs are needed to fully utilize all labeled particles for training. About 10% of charged particles are selected in order to balance preserving neighboring structures and training convergence/speed. Table 1 includes the numbers of selected charged LV and PU particles per graph per epoch and the total number of charged LV and PU particles per graph. The effect of the selected charged LV and PU particles ratio has been studied by varying it in a certain region around known charged LV and PU particles ratio, and the effect is found to be small.

Step (c) GNN Encoding and Step (d) Prediction. Any GNN architecture, like (11; 12; 13), can be applied to our SSL framework, though we focus on a variant of the gated GNN model (14). Because, as shown in Fig. 1, there are some LV particles surrounded by PU particles, we use gates to control the messages from the neighbors to better fit the problem. Let h_v^k denote the node v representation at

$\mathrm{n_{PU}}$	# Particles (in total)				# Selected Particles (for training)	
	Charged LV	Charged PU	Neutral LV	Neutral PU	Charged LV	Charged PU
80	85 ± 30	1600 ± 300	50 ± 20	800 ± 140	10	160
140	60 ± 14	3000 ± 350	30 ± 13	1420 ± 200	6	282

Table 1: The first four columns include the average numbers of four types of particles under different pileup conditions per graph. The last two columns indicate the number of charged particles being randomly selected for training per graph per epoch.

k-th layer. The GNN follows

$$\begin{aligned}
\text{Edge message:}\quad & m_{uv} = \left[h_u^{k-1}, h_v^{k-1}, \Delta\eta_{uv}, \Delta\phi_{uv}, \Delta R_{uv}\right],\\
\text{Aggregation:}\quad & m_v = \textstyle\sum_{u\in N(v)} g_{uv} m_{uv}, \text{ where } g_{uv} = \text{Sigmoid}(W_1 m_{uv} + b_1)\\
\text{Node-level gate:}\quad & q_v = \text{Sigmoid}(W_2[h_v^{k-1}, m_v] + b_2)\\
\text{Node update:}\quad & h_v^k = \text{ReLU}(q_v(W_3 h_v^{k-1} + b_3)) + (1-q_v)(W_4 m_v + b_4)),
\end{aligned}$$

where $\Delta\eta, \Delta\phi, \Delta R$ are the geometric features that characterize the difference between the spatial coordinates η, ϕ of two particles and their distance $\sqrt{\Delta\eta^2 + \Delta\phi^2}$ respectively. The node representations are initialized as particle features that in our experiments include the particle transverse momentum p_{T} and one-hot label encoding, that is, $(1, 0, 0)$ for LV charged particles, $(0, 1, 0)$ for PU charged particles, $(0, 0, 1)$ for neutral particles and masked charged particles. The node representations of the selected particles in the final layer of GNN are put through a multi-layer perceptron with two hidden layers to make the final prediction.

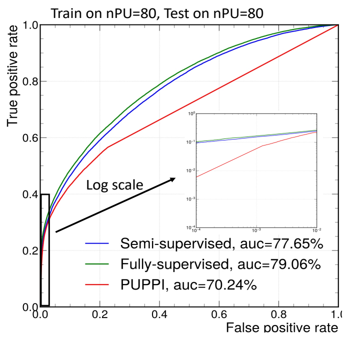
4 Results

The datasets used in this study are generated using PYTHIA 8.223 (15) and DELPHES 3.3.2 (16). Two pileup conditions are chosen to be studied: the number of pileup interactions $\mathrm{n_{PU}} = 80$ and 140, where the latter represents a more noisy experimental environment. Table 1 shows the average numbers of different types of particles under different pileup conditions.

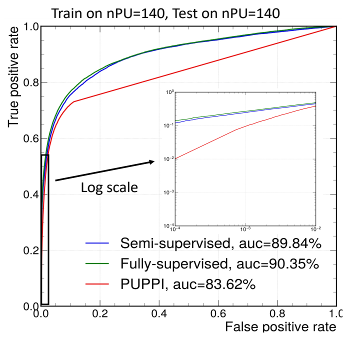
The experiments are designed to demonstrate the effectiveness of the model trained via SSL and its ability to be adapted to different $\mathrm{n_{PU}}$ levels. The model is trained and tested under three scenarios: (a) training the model on $\mathrm{n_{PU}} = 80$ with inference on $\mathrm{n_{PU}} = 80$, (b) training the model on $\mathrm{n_{PU}} = 140$ with inference on $\mathrm{n_{PU}} = 140$, and (c) training the model on $\mathrm{n_{PU}} = 80$ with inference on $\mathrm{n_{PU}} = 140$. Experimental settings (a) and (b) are to demonstrate that the model trained on charged particles with SSL can perform well on neutral particles in testing and work under different $\mathrm{n_{PU}}$ levels. Specifically, the SSL model is compared with the fully supervised model, which has the same architecture but is trained directly on neutral labels, and with the baseline algorithm PUPPI. Experiment (c) is to check the adaptation ability of the model in different $\mathrm{n_{PU}}$ conditions.

To evaluate the fully supervised model performance, the particles used for training and testing must come from different events, though this is not necessary in the SSL training. Therefore, for experiments on $\mathrm{n_{PU}} = 80$, there are 3000/1000/1000 events for training/validation/testing. When $\mathrm{n_{PU}} = 140$, 1000/400/800 events are used for training/validation/testing. For the $\mathrm{n_{PU}} = 140$ scenario, there are more particles per event, so the total number of events is reduced to maintain reasonable memory usage. During training, the model is trained until convergence, which normally takes about 5 epochs.

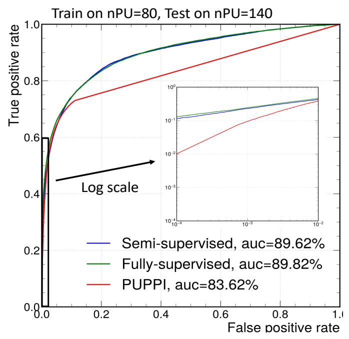
The testing results demonstrate the success of the model, and are shown in Fig. 3 ROC curves with AUC scores. In Fig. 3, (a) and (b) both indicate the SSL model outperforms the baseline algorithm PUPPI (by 7.41% under $\mathrm{n_{PU}}$=80 and 6.22% under $\mathrm{n_{PU}}$=140). Furthermore, the SSL performance is very similar to the fully-supervised model(decays by 1.41% under $\mathrm{n_{PU}}$=80 and 0.51% under $\mathrm{n_{PU}}$=140). The comparison between (a) and (c) in Fig. 3 demonstrates the model can adapt between different $\mathrm{n_{PU}}$s since the model only degrades by 0.22% under SSL from (b) to (c) in Fig. 3.



(a)



(b)



(c)

Figure 3: The ROC curves for the gated GNN on neutral particles for SSL, fully-supervised learning, and the domain PUPPI algorithm.

References

[1] M. Cacciari, G. P. Salam, and G. Soyez, "SoftKiller, a particle-level pileup removal method," *Eur. Phys. J. C*, vol. 75, no. 2, p. 59, 2015.

[2] D. Bertolini, P. Harris, M. Low, and N. Tran, "Pileup Per Particle Identification," *JHEP*, vol. 10, p. 059, 2014.

[3] P. T. Komiske, E. M. Metodiev, B. Nachman, and M. D. Schwartz, "Pileup Mitigation with Machine Learning (PUMML)," *JHEP*, vol. 12, p. 051, 2017.

[4] J. Arjona Martínez, O. Cerri, M. Pierini, M. Spiropulu, and J.-R. Vlimant, "Pileup mitigation at the Large Hadron Collider with graph neural networks," *Eur. Phys. J. Plus*, vol. 134, no. 7, p. 333, 2019.

[5] V. Mikuni and F. Canelli, "ABCNet: An attention-based method for particle tagging," *Eur. Phys. J. Plus*, vol. 135, no. 6, p. 463, 2020.

[6] B. Maier, S. M. Narayanan, G. de Castro, M. Goncharov, C. Paus, and M. Schott, "Pile-Up Mitigation using Attention," 7 2021.

[7] A. M. Sirunyan *et al.*, "Pileup mitigation at CMS in 13 TeV data," *JINST*, vol. 15, no. 09, p. P09018, 2020.

[8] A. M. Sirunyan *et al.*, "Identification of heavy, energetic, hadronically decaying particles using machine-learning techniques," *JINST*, vol. 15, no. 06, p. P06005, 2020.

[9] "Calibration of light-flavour jet b-tagging rates on ATLAS proton-proton collision data at $\sqrt{s} = 13$~TeV," 4 2018.

[10] *Semi-supervised learning*, ser. Adaptive computation and machine learning. MIT Press, 2006.

[11] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2018.

[12] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2018.

[14] Y. Li, R. Zemel, M. Brockschmidt, and D. Tarlow, "Gated graph sequence neural networks," in *Proceedings of ICLR'16*, April 2016. [Online]. Available: https://www.microsoft.com/en-us/research/publication/gated-graph-sequence-neural-networks/

[15] T. Sjöstrand, S. Ask, J. R. Christiansen, R. Corke, N. Desai, P. Ilten, S. Mrenna, S. Prestel, C. O. Rasmussen, and P. Z. Skands, "An introduction to PYTHIA 8.2," *Comput. Phys. Commun.*, vol. 191, pp. 159–177, 2015.

[16] J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi, "DELPHES 3, A modular framework for fast simulation of a generic collider experiment," *JHEP*, vol. 02, p. 057, 2014.