Proceedings of CTD 2020
PROC-CTD2020-25
August 19, 2020

Displaced Event Classification Using Graph Networks

Kim Albertsson[1,2] Federico Meloni[3]

[1]*CERN,* [2]*Luleå University of Technology,* [3]*DESY*

ABSTRACT

Long-lived massive particles, predicted in numerous Standard Model extensions, are a particularly difficult target for online event reconstruction. This work presents a study of detecting the presence of displaced vertices in a collider experiment in several environmental conditions. In particular Graph Neural Networks performing classification on input hit-level data are shown to perform well in the task of separating prompt against displaced events with results translating, with some degradation, into more busy environments. Furthermore, promising results are shown for identifying events from a benchmark supersymmetric process with future work investigating higher pileup environments.

PRESENTED AT

Connecting the Dots Workshop (CTD 2020)
April 20-30, 2020

1 Introduction

LHC searches for new physics involving massive long-lived particles (LLPs) often rely on the LLP decay products to select events online. Current selection methodologies optimised for prompt decays can become very inefficient when these decay products have low energy. The direct identification of displaced decays is limited by the fact that running a full track and vertex reconstruction at a sufficient rate during active data-taking, is computationally infeasible. Identifying the presence of displaced decays from raw detector hits would allow to mitigate the selection inefficiency.

In the Standard Model, heavy particles like top quarks and weak bosons decay very close to the interaction point. However, there are several sources of displaced decays of light particles, such as B and K mesons, and interactions of primary hadrons with the detector material. Many theories for physics beyond the Standard Model (BSM) predict LLPs that could decay producing high-mass displaced vertices within the inner tracking volume stressing the need to accurately discriminate between light and heavy secondary vertices [1].

The problem will be studied using simulated detector hits and represented in cylindrical coordinates (r, ϕ, z). A supersymmetric (SUSY) particle production process, shown in Figure 1, was chosen as benchmark signal, while the Standard Model multi-jet production was used to model the backgrounds. Event-level classification will be used to identify of an interesting event features, such as the presence of a secondary vertex. This is a challenging problem as the per-event topology is highly variable both in size and connectivity. Recently, Graph Neural Networks have shown promise in track reconstruction and will be further investigated here in the context of event selection.

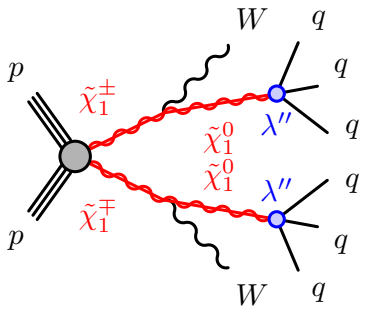


Figure 1: The principal process under study is the pair production of supersymmetric charginos, denoted $\chi_1^{\pm}$, which decay into two neutralinos, χ_1^0, and two W bosons. Because of their weak coupling (λ'') to quarks, the neutralinos decays into three quarks which fragment and hadronise into jets. For this study both supersymmetric particles are simulated with a mean life-time of 1 ns and are thus long-lived.

2 Graph Neural Networks

To effectively discriminate different types of events, an algorithm needs to consider the event topology. Due to the underlying physics, the number of final state particles is highly variable each event and as a consequence both the topology and the number of detector hits are as well. This sparse and dynamic structure is challenging for many deep learning approaches such as dense (e.g. as investigated in our previous work [2]), and convolutional neural networks.

Convolutional neural networks have recently seen successful application to calorimeter data, for example in jet classification, due to the image-like structure of the data. However, particular care needs to be taken due to the sparsity of the data [3]. The tracking detector modeled in this work can be construed as image planes, similar to a typical image interpretation of a calorimeter, however, in a tracking detector sparsity issues are more pronounced.

Both recurrent and convolutional neural networks have seen recent successes in online event classification using reconstructed particles as inputs [4]. However, since it relies on reconstructed inputs its applicability to displaced signatures is limited.

Recently, graph neural networks coupled with an explicit graph construction step have been showing promise in processing sparse, structured data by directly operating only on active elements allowing the architecture to be applied to low level detector hits. Ferrel et al. demonstrated its potential for application to primary track reconstruction in [5].

A strategy similar to that of Ferrel el al. was adopted for this work with the addition of the aggregation network from [6] to provide a per-event binary classification output for event selection. In [5] the per-segment output is scalar, we use a multidimensional output (3–4 depending on the target). For this experiment internal dense networks are all single layer networks with the same width, yielding a single hyper-parameter to control the entire network capacity. A schematic overview of the architecture can be seen in Figure 2. The network design, training and data loading was implemented using pytorch (v1.1.0).

The graph building step consists of generating segment candidates from the input hits based on the angle between two hits and the origin. The segment candidate is additionally extrapolated to the beamline crossing, and if the deviation in z is too large the candidate is excluded.

The training operates on graph data, taking as input an event after the graph building step. Two targets are used during training: a per-event output (Marked A in Figure 2) classifies the event as either displaced, or as containing an LLP, depending on the task; and a per-segment output (Marked E in Figure 2) is used to guide the hidden representation to classify segments into categories fake, prompt, displaced, and optionally, whether the segment was generated by a supersymmetric particle.

To evaluate the architecture described in this section, three tasks of increasing difficulty were defined: Task 0, the initial task, prods the general applicability of the approach to separating displaced events from prompt ones in a simplified 2D environment; Task 1, the second task, introduces realistic physics processes and particle propagation in a 3D environment; Task 2, the third and final task, further looks at separating events containing a supersymmetric signature from events lacking such a signature putting increased requirements on the network to .

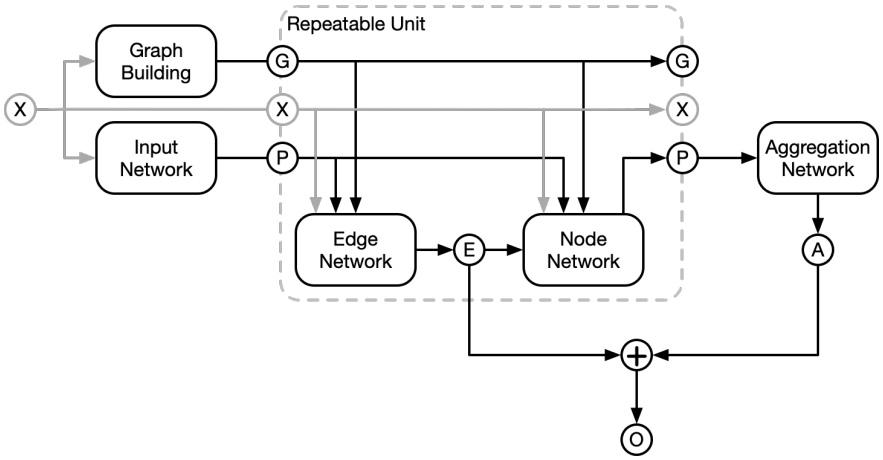


Figure 2: The network architecture consists of four dense networks. The input network transforms the initial input hits, labelled X, to a hidden representation, labelled P. A separate graph construction step yields candidate segments for the input hits. The collection of segments is labelled G. The edge network outputs classification predictions for each segment, labelled E, while the node network refines the hidden representation for each hit. The application of the edge and node networks are repeated for a set number of times. Finally the hidden representation is passed to an aggregation network producing an event wide prediction, labelled A. The final output comprises the edge predictions and the event prediction during training during training while during evaluation the output consists solely of the aggregation network output.

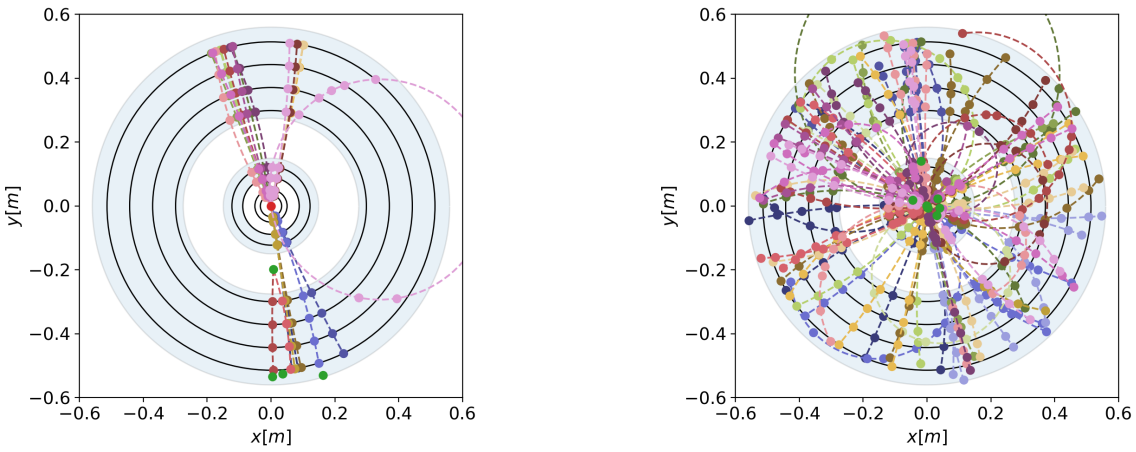
3 Datasets and generation

For this experiment a dedicated simulation setup was developed to describe a simplified collider detector. The geometry and the layout was inspired by the silicon part of the ATLAS inner detector geometry. Eight barrel layers and 12 end-cap layers were included, corresponding to the silicon-pixel and silicon-strip components, with the active surfaces modeled as perfect cylinders and disks respectively. The interaction point between a simulated particle track and the detector surface was recorded as an exact hit in three dimensions, r, φ, and z. Particles were propagated in a uniform 2 T magnetic field. The simulation did not include multiple scattering, hadronic interactions, nor hit resolution effects. MadGraph5 aMC@NLO 2.7.2 [7] was used to generate the hard-scatter event, and parton showering was performed with Pythia8 [8]. Finally Delphes [9], with a custom module, was used for particle propagation. For the simulated event samples including the effects of pile-up interactions in proton-proton collisions, a varying number of simulated minimum-bias interactions were overlaid on the hard-scattering event, following a Poisson distribution. These interactions were generated using Pythia 8.2.

Three datasets were generated to perform the studies presented in this document, including two different SUSY production modes: an electron-positron collision environment with low track-multiplicity, and a proton-proton collision environment with larger track-multiplicity.

- $e^+e^- \to \chi^+\chi^-$ — A low track-multiplicity environment used for training. Events have on average 26 tracks and 200 hits. A total of 10^6 events were generated, and after pre-processing 10^5 were used in further experiments.
- $pp \to \chi^+\chi^-$ — The proton-proton collision version of the process is busier and was generated both without and with pileup ($\mu = 10$) used for evaluation. Events have typically 200–300 (900–950) tracks and 1100–1500 (4400–4900) hits in the $\mu = 0$ ($\mu = 10$) dataset depending on the value of the supersymmetric particle masses. A total of 10^5 (10^4) events were generated for the $\mu = 0$ ($\mu = 10$) dataset.
- $pp \to jj$ — A typical multijet background used for evaluation. A total of 10^4 events were generated with a mean of about 200 tracks and 1000 hits per event.

Example events from these processes can be seen in Figure 3. The mean lifetime of the supersymmetric particles is set to 1 ns and their masses lie in the 100-300 GeV range (see task descriptions in Section 4 for exact values).



(a) $e^+e^- \to \chi_1^+\chi_1^-$ (b) $pp \to \chi_1^+\chi_1^-$

Figure 3: View of the xy-plane of a typical event in the simulated detector. True particle tracks are visible as dashed lines, solid markers are detector hits. Two marker colors have specialised meaning: A red marker indicates the position of the primary vertex, and a green marker indicates a displaced vertex.

For the electron-positron dataset a pre-processing step was conducted by: excluding trivial events with only 2 visible tracks, and, as in our previous work, introducing a gap of 5 cm between prompt and displaced events to facilitate learning. This means no events containing a secondary vertex in the range $0 < R < 5$ [cm] are retained.

In addition to the three dimensional datasets defined above, a two dimensional dataset was used for an initial step of the investigation. This dedicated dataset simulates a slice in the detector xy-plane and is concerned only with in-plane particle-like tracks. Tracks are generated independently and no event-wide energy-momentum conservation is required. Eight barrel layers were used in the simulation, similar to the three dimensional setup. The generation process can be tuned to produce a number of tracks from the primary vertex. For displaced vertex generation, a set number of tracks originating from the primary vertex are selected and a secondary vertex is placed along their trajectory. Secondary vertices are required to appear inside the tracking volume and the selected primary tracks are subsequently removed.

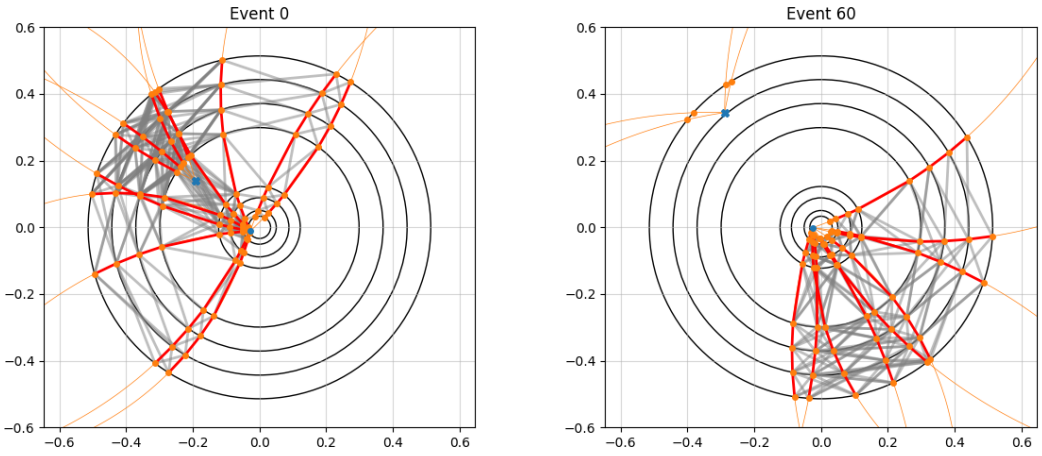
For all datasets, besides the two-dimensional toy data, a pre-processing step was applied limiting the number of hits per track to the ten first, including only hits from the barrel region. This as a simple way to limit control the momentum of considered tracks, and to simplify the graph construction step.

4 Case studies

Three tasks of increasing difficulty were defined to investigate the efficacy of the network. The first two, denoted Task 0 and Task 1, investigate how the network scales from a simple to a more realistic collision environment. The last task , most interesting for future application, investigates the network's ability to discriminate displaced vertices coming from different processes.

4.1 Task 0 – 2D environment

The initial task investigated involved separating prompt events (no displaced vertex) from those with a displaced vertex. A simpler version of the full architecture described in Section 2 was applied to the two dimensional dataset as an initial test of the suitability of the approach.



(a) Signal-like event

(b) Background-like event

Figure 4: Showing two displaced events, in (a) correctly identified, and in (b), with an incorrect prediction. Solid orange lines indicate true particle path. For segments, red solid lines indicate a true segment, while grey indicate a fake segment. A blue round marker indicates the primary vertex, and a blue cross indicates a secondary vertex. Note that the background-like displaced event contains no track segments from the displaced tracks.

The network was trained using a small, $O(10^3)$ events, dataset with an equal distribution of prompt and displaced events. For evaluation an independent sample generated with the same process as the training data was used.

The network accurately classified all events containing a segment originating from a displaced vertex. Interestingly the network misclassified events where a displaced vertex lay within the detector volume and generated hits, but failed to generate detectable segments, as can be seen in Figure 4.

The failure mode is not expected to significantly impact inference in a production environment, but highlights a limitation of the network to exploit hits lacking connectivity information.

4.2 Task 1 – Displaced event classification

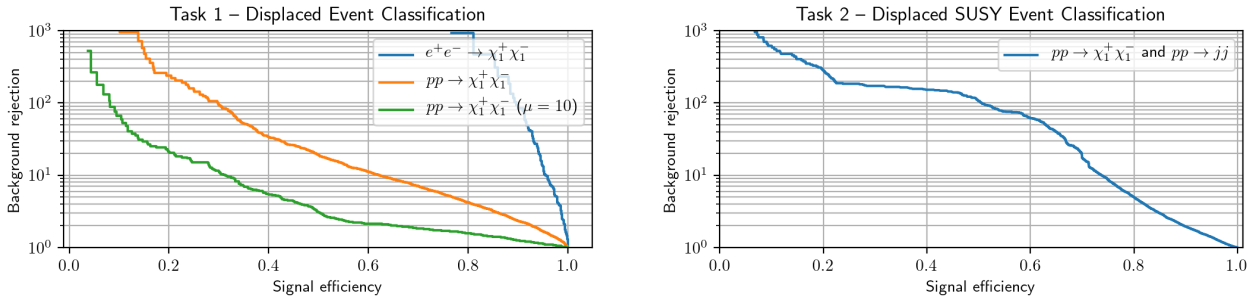
The second task investigates how well the network performance translates to different levels of pileup by training in a low track multiplicity process and evaluating in increasingly more busy environments. The classification target was to separate prompt events from displaced events.

For training the electron-positron dataset was used, while for evaluation an independent part of the electron-positron data was used together with the proton-proton version of the SUSY process, both with and without pileup. When included, the pileup is set to $\mu = 10$. The datasets span a range of track multiplicities of 30–900, about half an order of magnitude.

The mass for the chargino was set to 181 GeV, while the mass of the neutralino was set to 96 GeV.

The performance of network on the three different datasets can be seen in Figure 5. The background rejection is measured as the inverse efficiency of selecting a background event as signal. The network can learn the simple environment well, but the performance is degraded for the two proton-proton datasets with the high pileup environment performing only slightly better than random chance.

The degradation across processes is driven primarily by two factors: The electron-positron sample has a margin of 5 cm around the interaction point not present in the proton-proton samples, and the different track multiplicities in the environments. The relatively stable performance retained across a seven-fold increase in track multiplicity is promising. The performance degradation could be recovered with a dedicated training targeting proton-proton collisions at a higher pileup level.



(a) (b)

Figure 5: ROC curves from Task 1 and 2. In (a): Outcome of Task 1, discriminating displaced from prompt events across three different physics processes of increasing difficulty. The network was trained on data similar to the electron-positron set shown here in blue, where the best performance is reached. When transitioning to process environments not explicitly trained for, some performance is retained, but at a pileup of $\mu = 10$ most separation power is lost. There is an approximately seven-fold difference in track multiplicity between subsequent processes. In (b): Outcome of Task 2, separating a displaced SUSY event from a multijet background. Note that the good performance is mostly driven by a difference in track multiplicity between the two samples.

4.3 Task 2 – Displaced SUSY event classification

The final task studies how well a SUSY process can be isolated from both prompt and other displaced events.

The network was again trained on electron-positron data where prompt events constitutes the majority of the background, with a smaller amount of events where the SUSY particle decayed close to the interaction point and gave rise to further Standard Model decays. For evaluation, the proton-proton SUSY process (without pileup) was combined with the multijet dataset.

The mass for the chargino was set to 281 GeV, while the mass of the neutralino was set to 196 GeV, an increase of 100 GeV from the previous task to increase the detector activity in the simulated data.

Note that in the previous two tasks, the signal and background were both drawn from the same dataset. In Task 2 however, they are drawn from different sets. As a result there is an imbalance with the signal sample having a higher track multiplicity due to the SUSY particles decaying into jets.

The result after training on the evaluation data can be seen in Figure 5. A high background rejection (a factor $\sim$100) can be seen up to a signal efficiency of about 50 percent.

The problem setting in task 2 is the most relevant for a fully implemented algorithm. The output of the classifier could be used as an independent trigger channel, or incorporated into an experiment trigger-menu.

Future studies will be aimed improving the independence of the performance on track multiplicity as this is one of the main discriminants that the algorithm is currently is exploiting. However, it can be noted that as the pileup level is increased, the difference between the signal and background will diminish.

5 Conclusion

This paper investigates the application of Graph Neural Networks towards identifying high-mass displaced vertices from raw detector hits without an explicit track and vertex reconstruction step. The investigation is carried out in three steps of increasing complexity. The first task investigates the applicability of graph networks to characterise displaced events in a trigger environment while the final two tasks investigate, respectively, how the performance of the setup translates to different detection environments, and the performance identifying a specific signal process against a multi-jet background. Examples have been shown of a retained ($\sim$20%) signal efficiency in the regime of high background rejection (a factor $\sim$100).

Interesting directions for future research include a full time complexity analysis, the network training using proton-proton processes with realistic levels of pileup and simulating a more detailed detector environment.

Acknowledgements

The authors would like to extend a heartfelt thank you to Andreas Hoecker for his support. This work was partly supported financially by the Productive 4.0 project (EU ARTEMIS JU grant agreement No. 737459).

References

[1] Lee, L. et al., "Collider Searches for Long-Lived Particles Beyond the Standard Model", Prog. Part. Nucl. Phys. vol. 106 p. 210-255 (2019).

[2] Albertsson, K., and Meloni F. "Towards Fast Displaced Vertex Finding", Connecting the Dots and Workshop on Intelligent Trackers, arXiv:1910.10508, (2019).

[3] Guest, D., Cranmer, K. and Whiteson, D. "Deep Learning and Its Application to LHC Physics", Annu. Rev. Nucl. Part. Sci. 68:161–81, (2018).

[4] Nguyen, T. et al. "Topology classification with deep learning to improve real-time event selection at the LHC", Computing and Software for Big Science volume 3 (2019).

[5] Farrell, S. et al. "Novel deep learning methods for track reconstruction", 4th International Workshop Connecting The Dots 2018, arxiv:1810.06111, (2018).

[6] Battaglia, P. W., et al. "Interaction Networks for Learning about Objects." Relations and Physics 6 (2016).

[7] Alwall J. et al., "The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations," JHEP 1407 (2014) 079.

[8] Sjöstrand, T. et al., "An Introduction to PYTHIA 8.2," Comput.Phys.Commun. 191 (2015) 159-177.

[9] DELPHES 3, de Favereau, J., Delaere, C. et al., "DELPHES 3: a modular framework for fast simulation of a generic collider experiment," J. High Energ. Phys. (2014) 2014: 57.