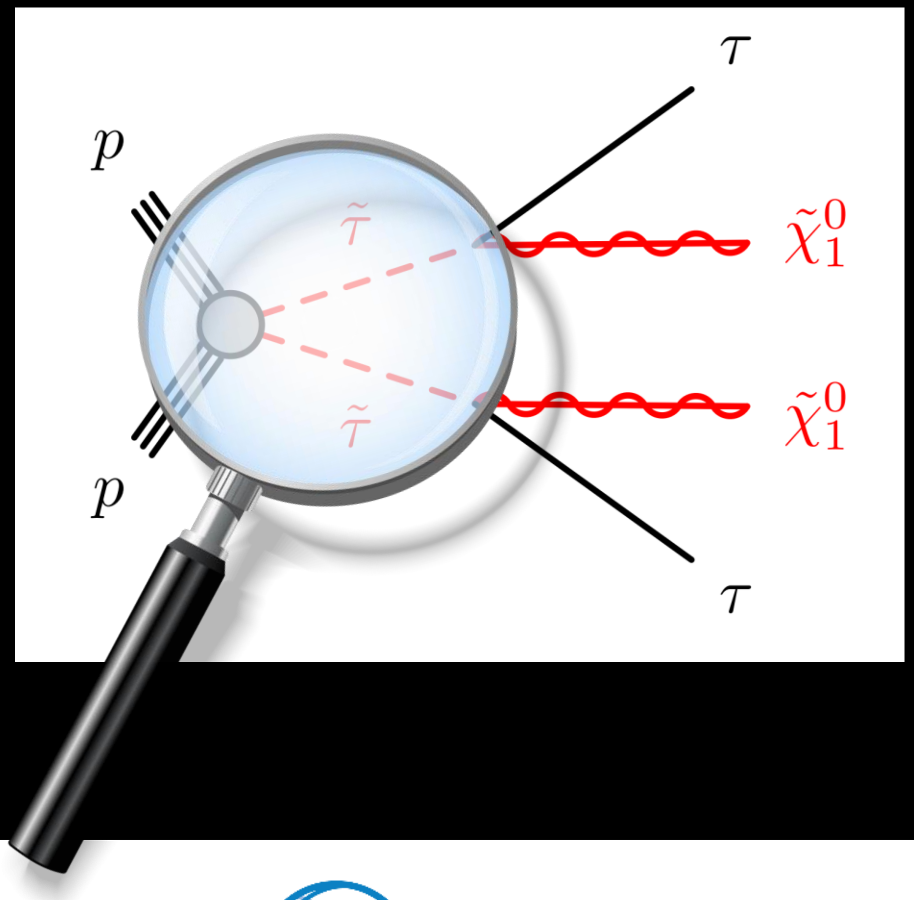
Direct Stau Production

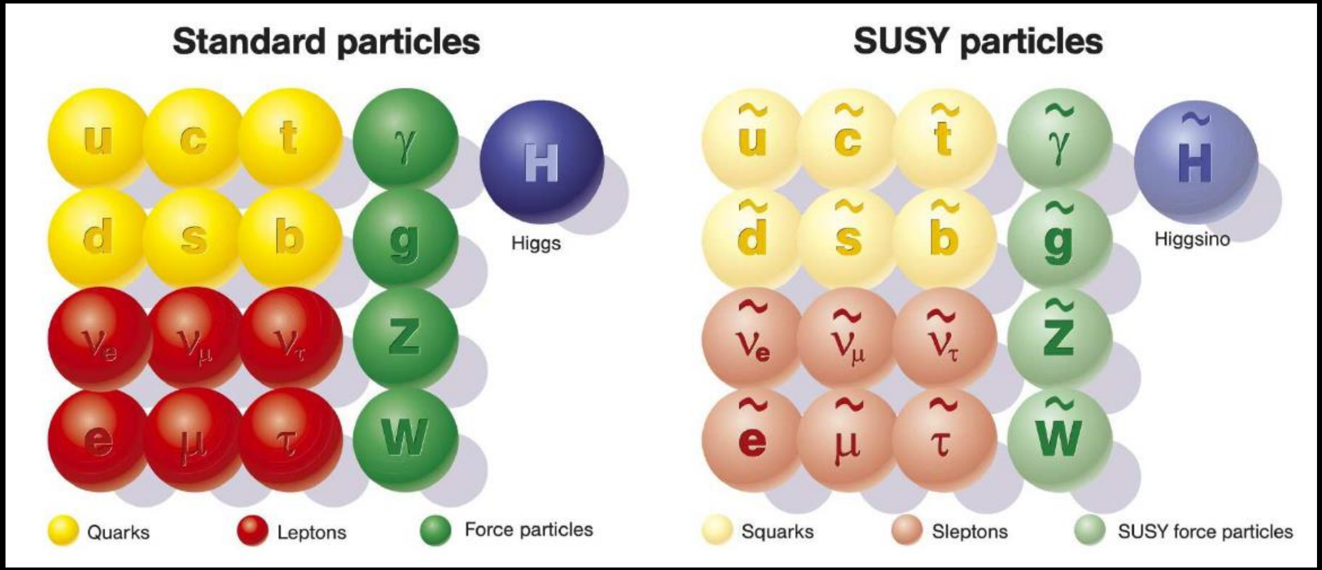
Searching for moderately compressed stau scenarios in the lepton-hadron final state using graph convolutional networks

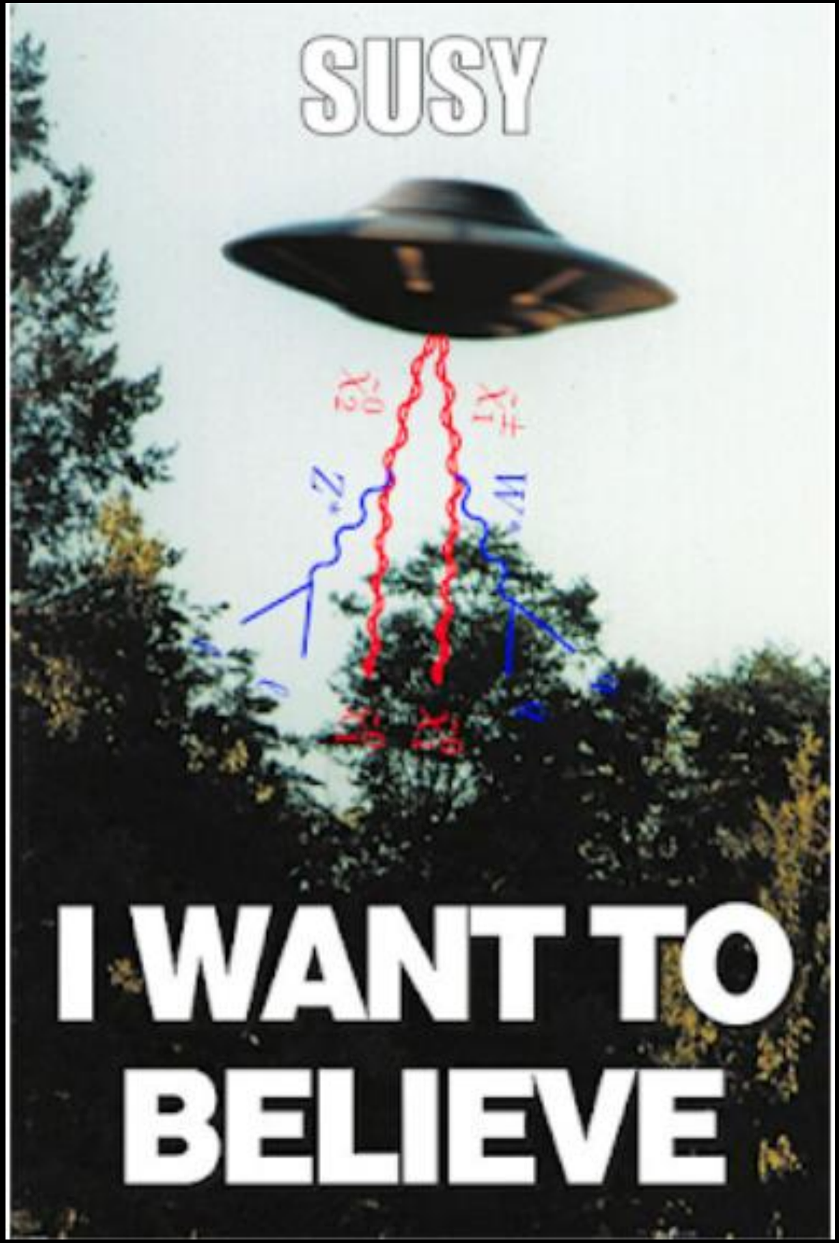
Sebastian Rutherford Colmenares - Tina Potter - Holly Pacey

April 2024

IoP APP, HEPP & NP



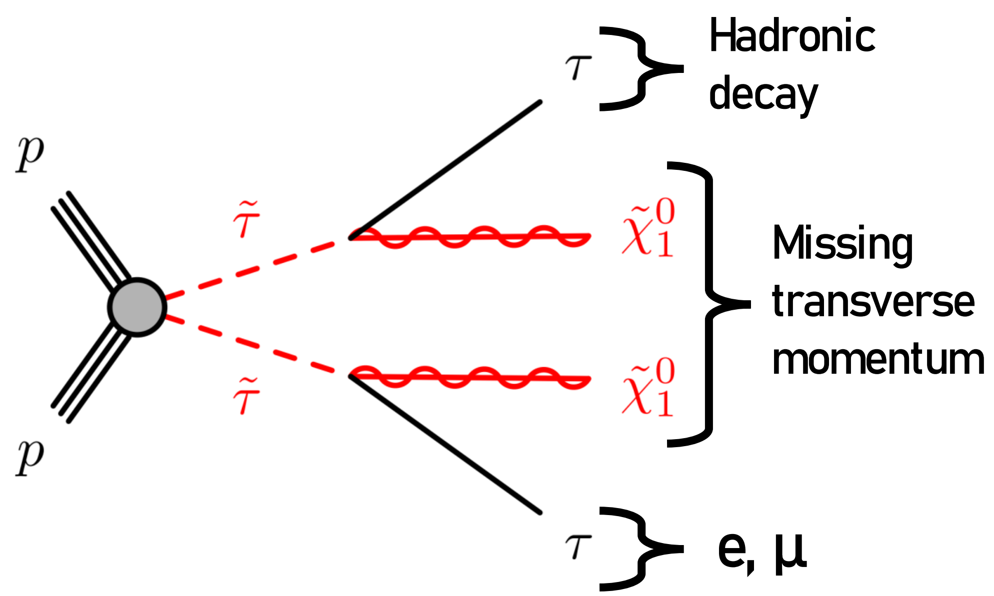


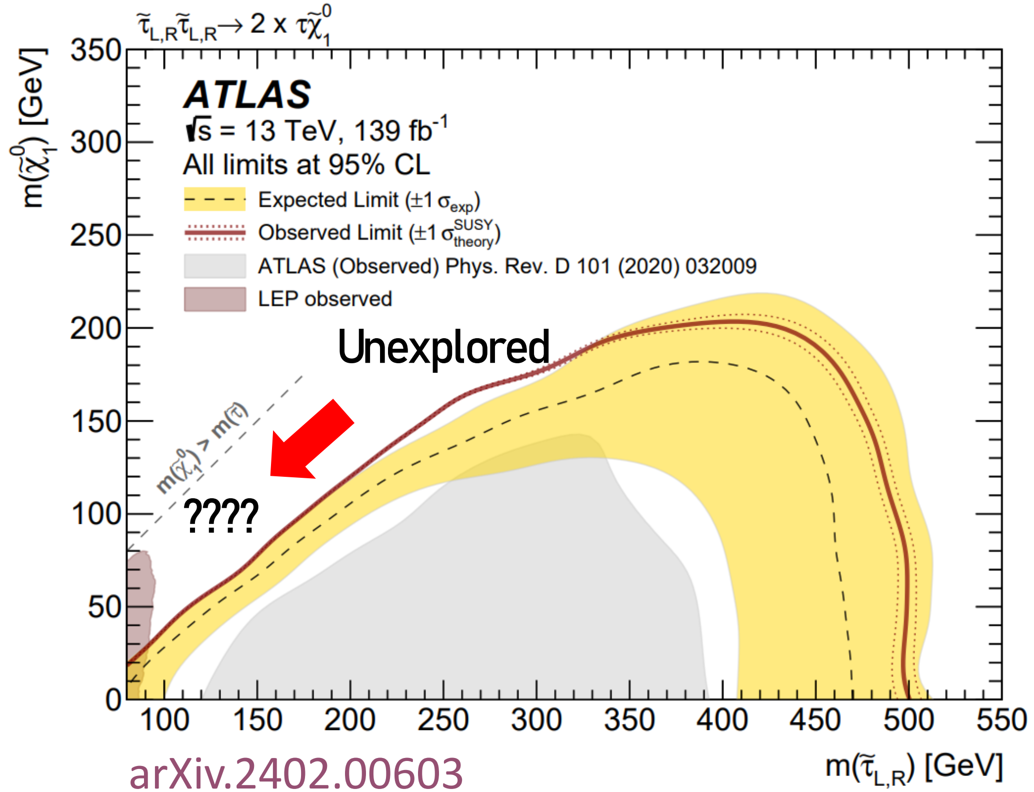


- Coleman–Mandula theorem *exception*
- Hierarchy
- Dark Matter
- Harmony of spin states

Staus status

- So far $\tau_{had}\ \tau_{had}$ is the main channel studied (BR = 42%)
  - Triggering on the two taus
- Large uncovered region in for models with $\Delta m(\tilde{\tau}, \tilde{\chi}_1^0) < \sim 60$ GeV
- Explore $\tau_{had}\ \tau_{lep}$ channel (BR = 46%)
  - lepton triggers have lower p_T thresholds
- Drawback: large W+jets and Z+jets backgrounds
- Make use of additional Run 3 data

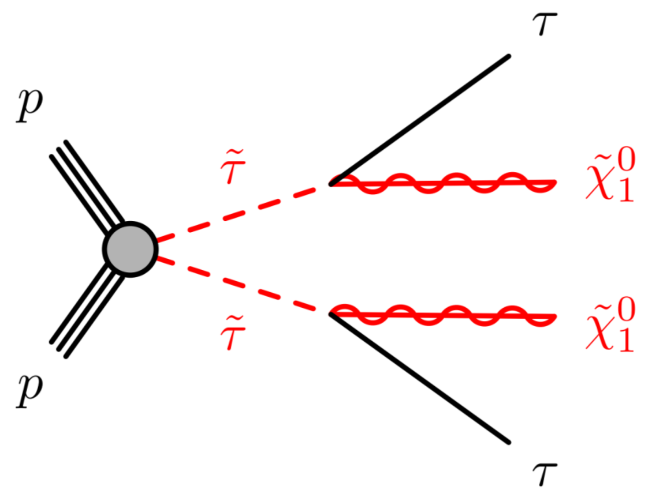


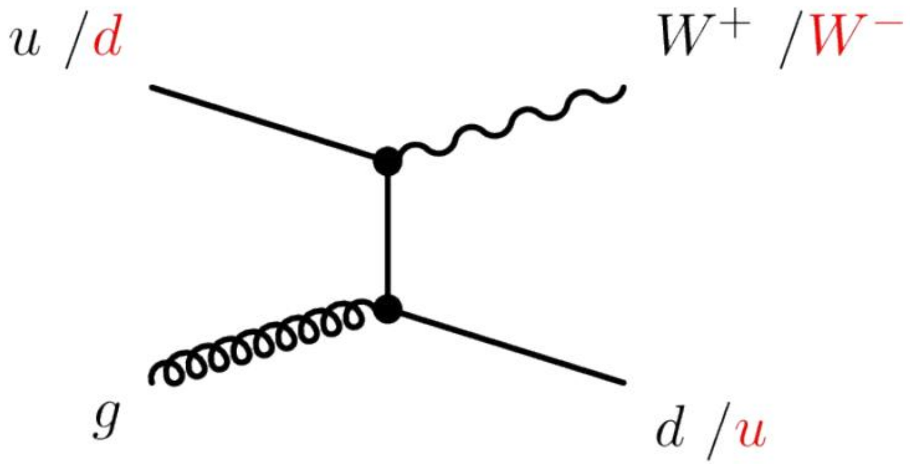


arXiv.2402.00603

Lep-Had Channel

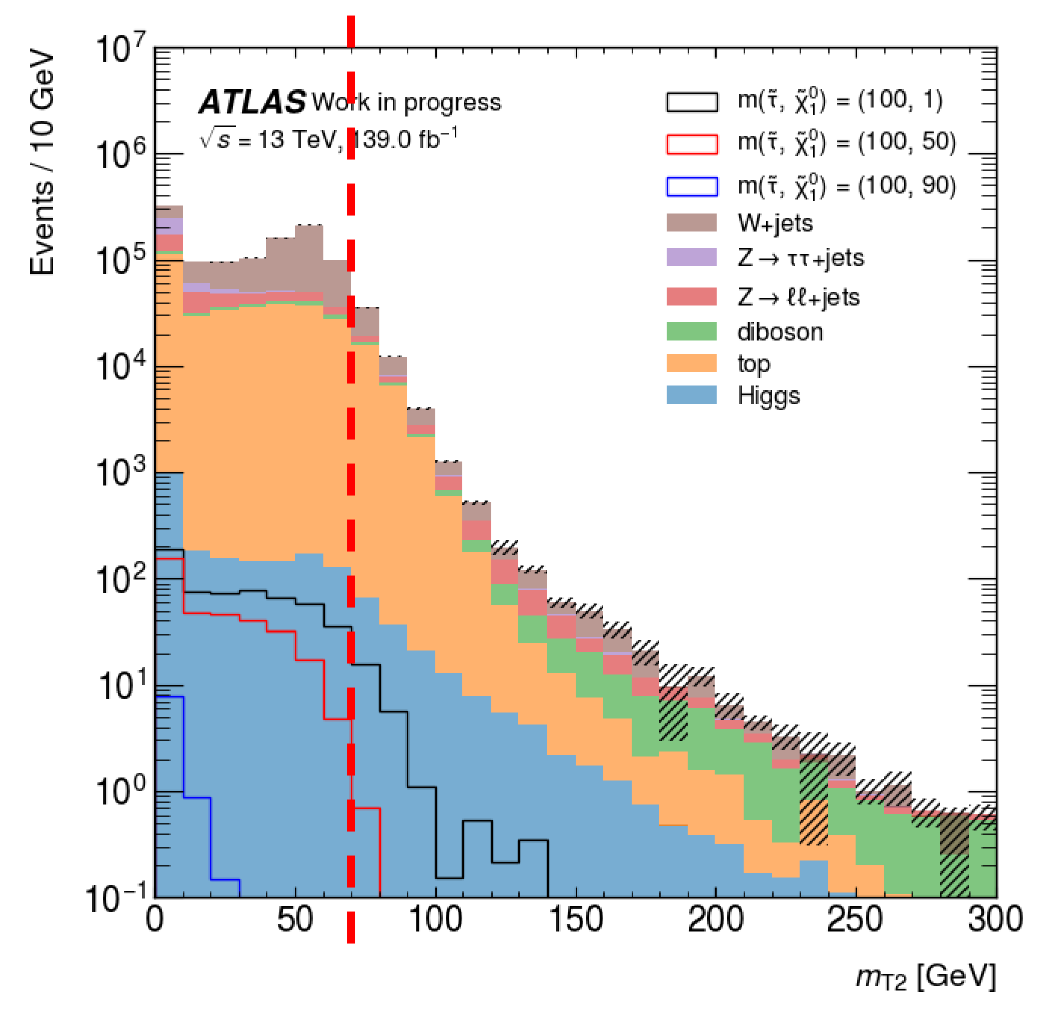
- Use single e/μ triggers (p_T > 27 GeV)
- W+jets background swamps 1 lepton and 1 tau selection
  - Fake tau predominantly comes from quark-initiated jet
- Signal is opposite sign (OS) lep-tau
- Fake taus not charge agnostic, N(OS) ~ 2xN(SS)





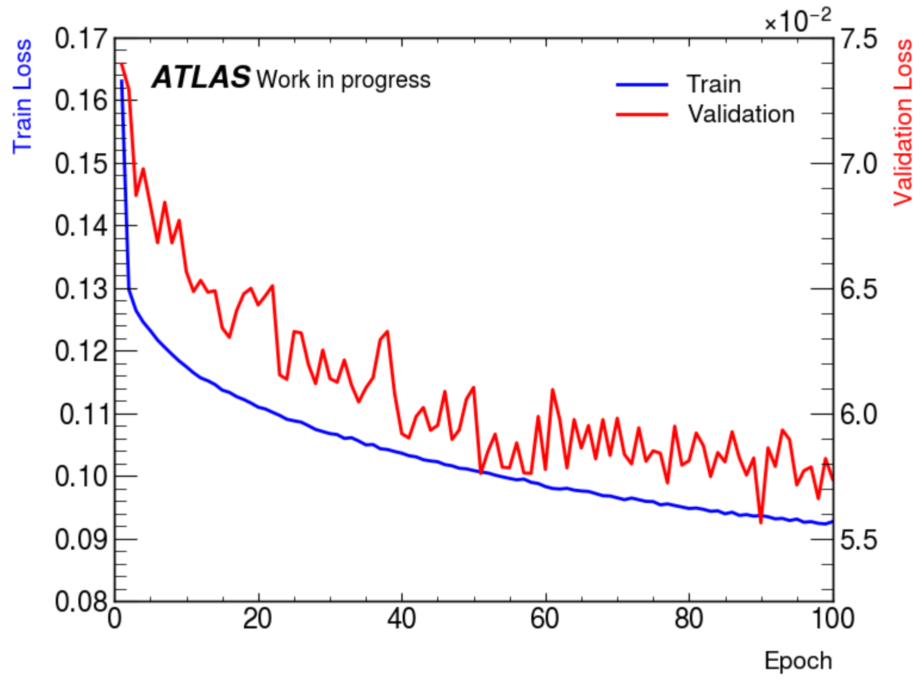
Cut & Count attempt (...towards ML preselection)

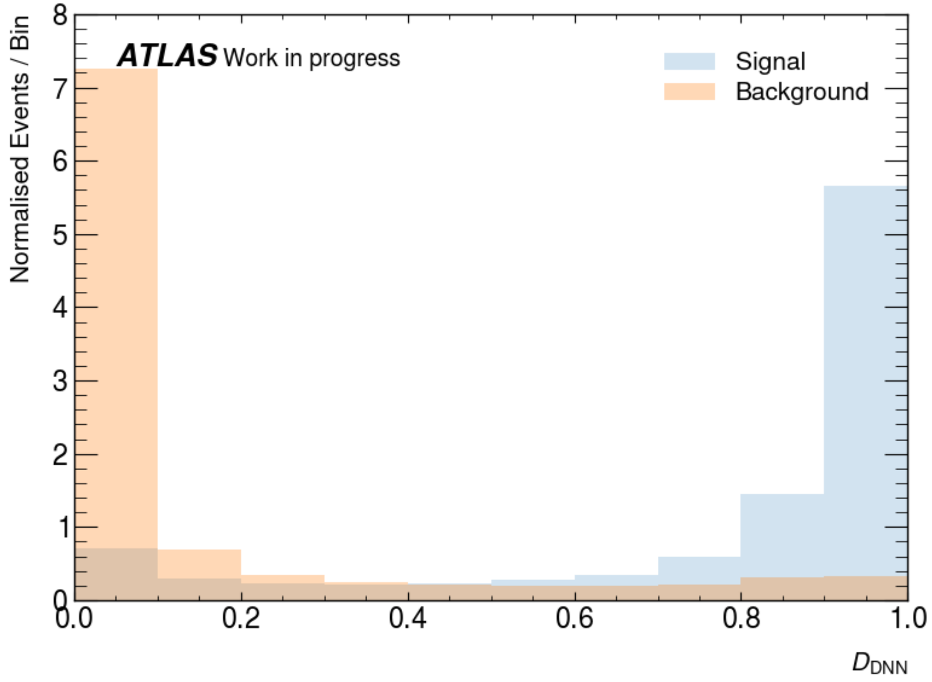
- Prospect studies looking at Run2
- Applied simple event selection to isolate signal and reduce background (see backup)
- The $\tau_{had}\,\tau_{had}$ analysis cut on 70 GeV in the stransverse mass (m_{T2})
  - SUSY signal extends beyond kinematic endpoint of background
- However, in small Δm scenarios, m_{T2} cut kills our signal



ML approach w/ DNN

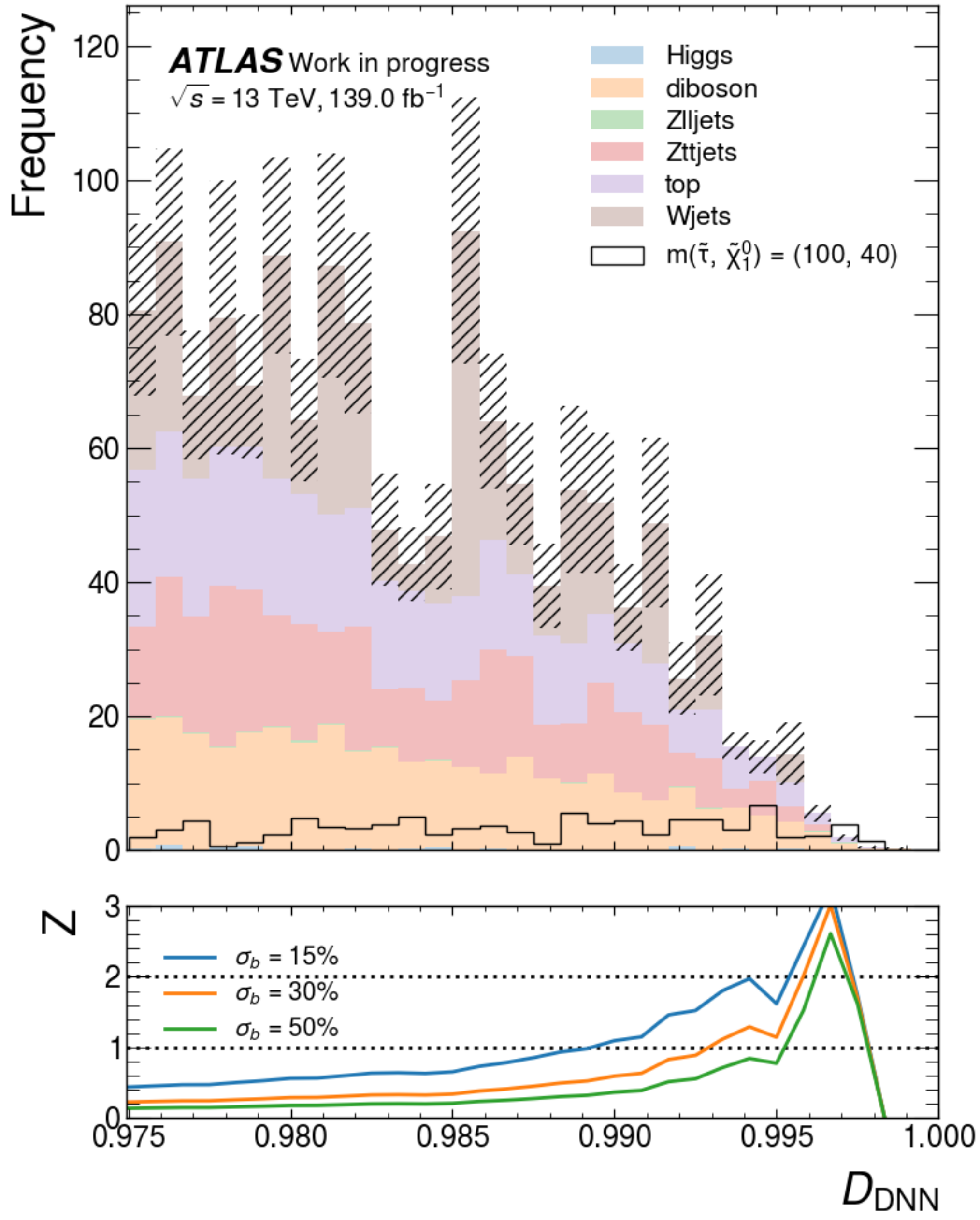
- Train deep neural network (DNN) to separate signal from background and cut on network score
- Use multiclass DNN with 7 output classes
  - (signal + 6 backgrounds: W+jets, $Z \rightarrow \ell\ell$+jets, $Z \rightarrow \tau\tau$+jets, top, diboson, and Higgs)
- Input variables (full list in backup):
  - Basic object variables: p_T, η, MET, charge etc.
  - High-level variables: ΔR, $\Delta\eta$, $\Delta\phi$, m_{T2}, M_{eff}, M_{inv}, Σm_T, m_{CT}, balance + more!
- (Full training details in backup)





ML approach w/ DNN

- Despite good signal/background separation, backgrounds remain large
- Background is swamping signal-like score region
- Only potential sensitivity at extreme tails of the score distribution

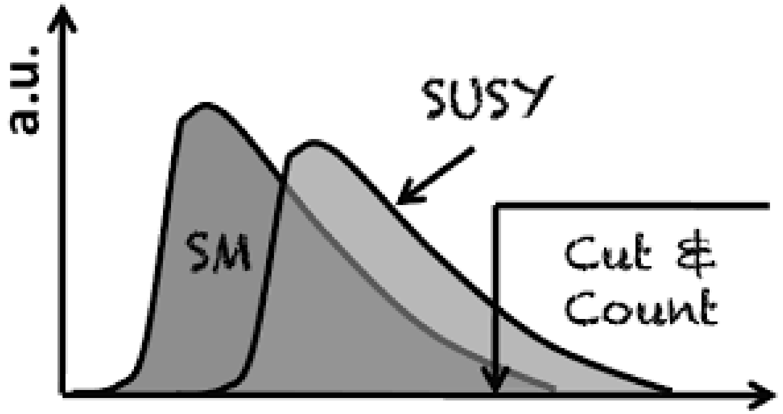


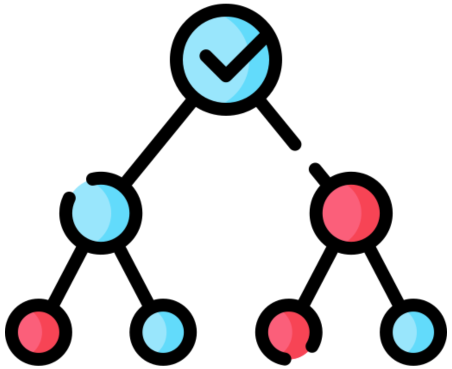
How we search for SUSY

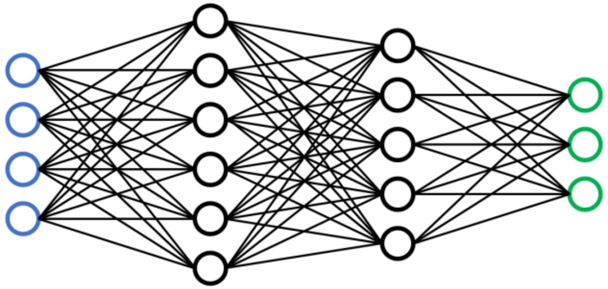
1. CUT & COUNT

2. MACHINE LEARNING
   - Boosted Decision Tree
   - Deep Neural Network

3. ???
   - Can we get additional signal background separation?

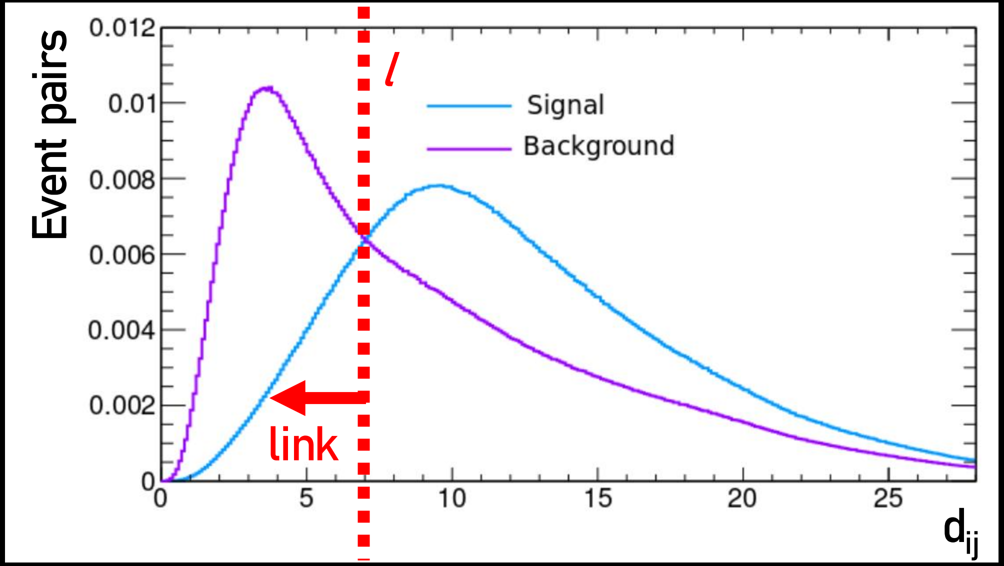




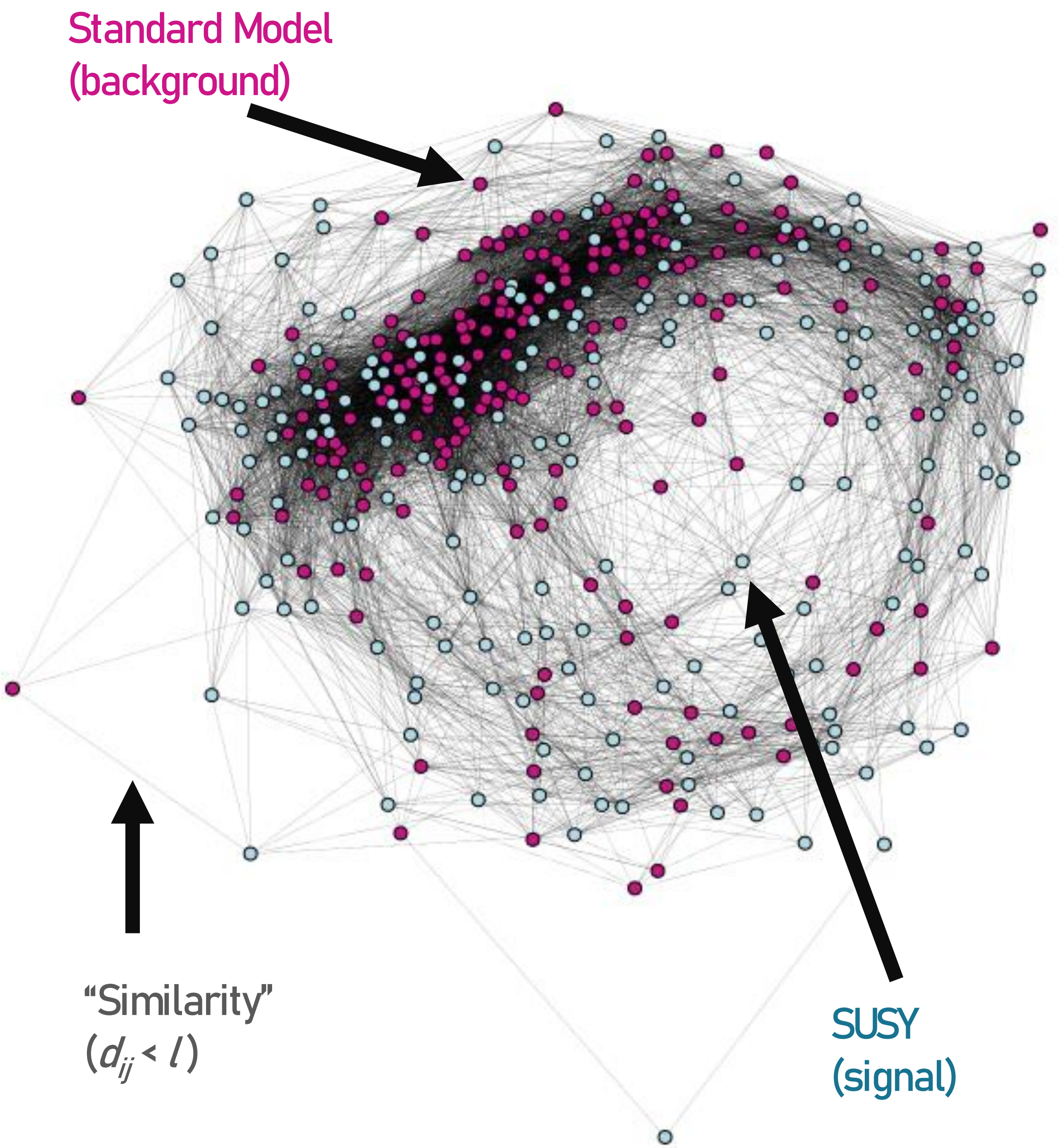


Does SUSY have friends?

- Compare pairs of events: how "similar" are they?
- Define distance d_{ij} between events based on kinematics
- Choose linking length l,



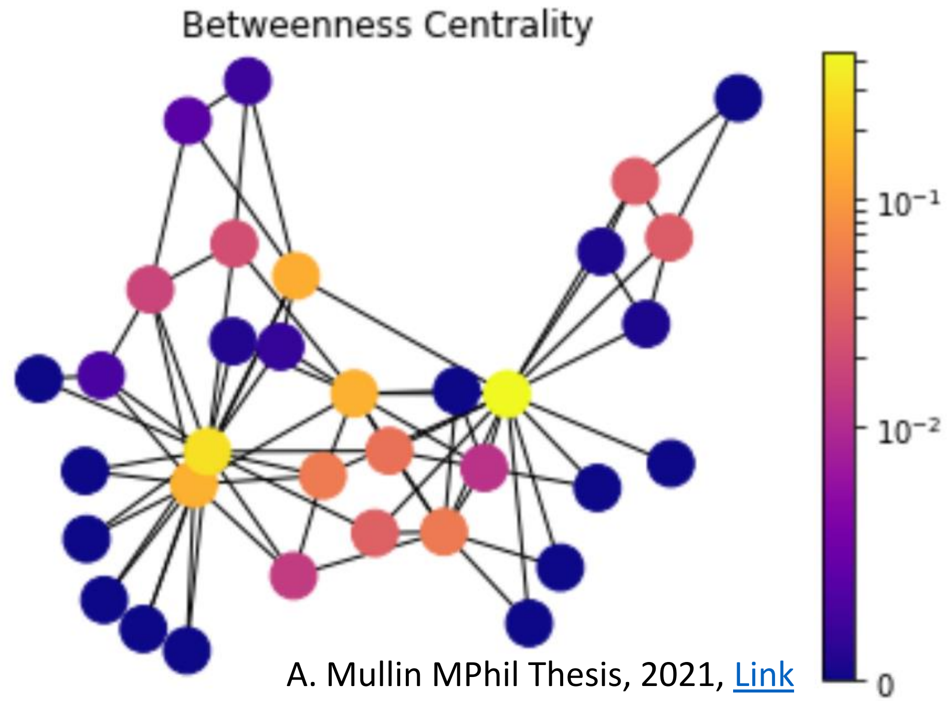
- Form graph of entire dataset, every event is a node



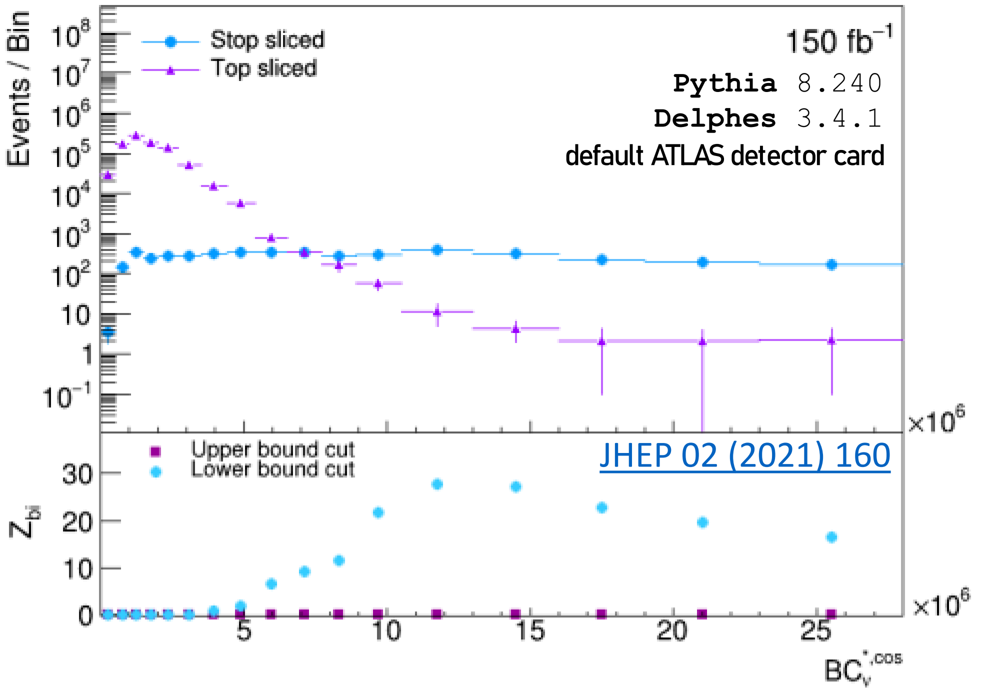
A. Mullin, S. Nicholls, H. Pacey, M. Parker, M. White, S. Williams: [JHEP 02 (2021) 160]

Does SUSY have friends?

- Use node properties from graph theory
- Construct new event variables (e.g. Betweenness Centrality *BC*)
- New distributions can be used to cut out background (or feed into ML)
- BIG PICTURE: *Additional discriminating information to be gained from looking at similarity between events*



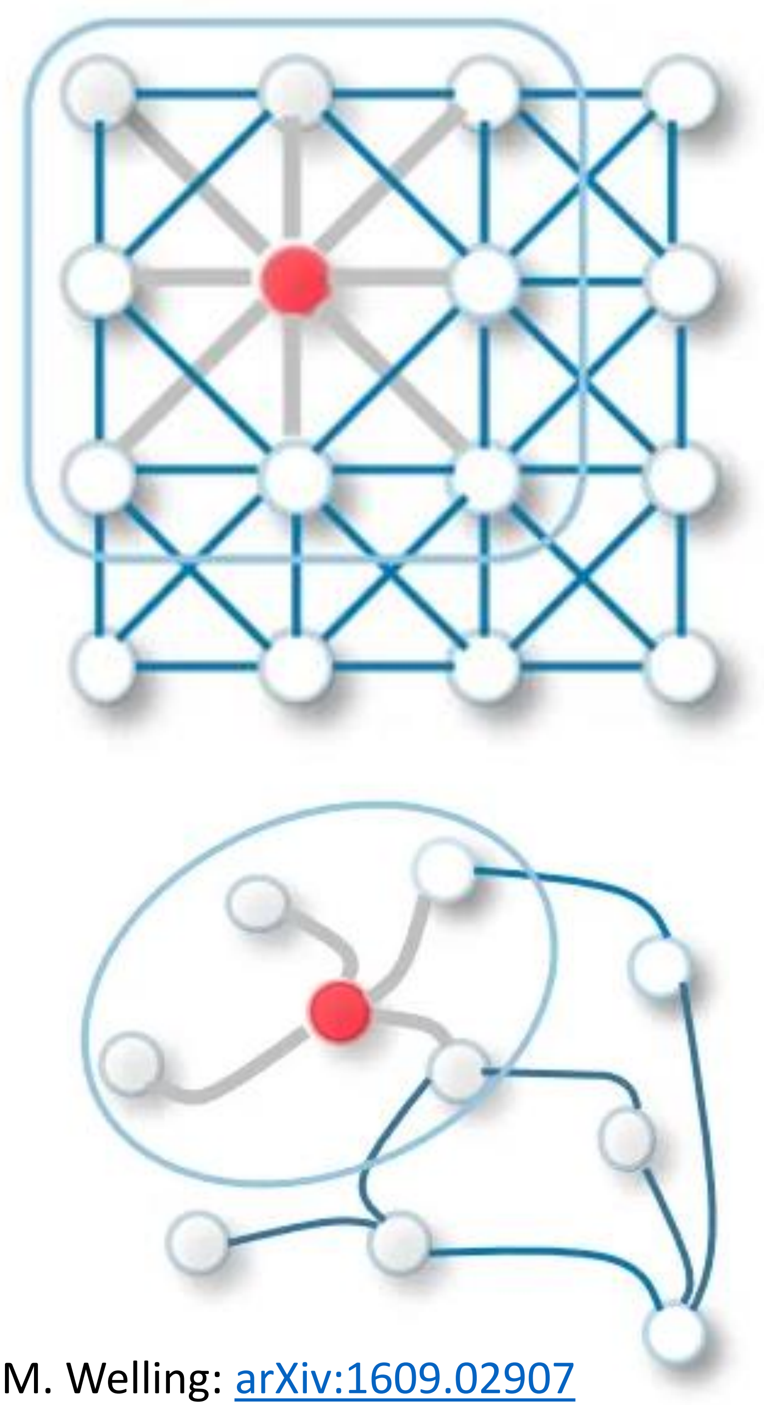
A. Mullin MPhil Thesis, 2021, [Link]



[JHEP 02 (2021) 160]

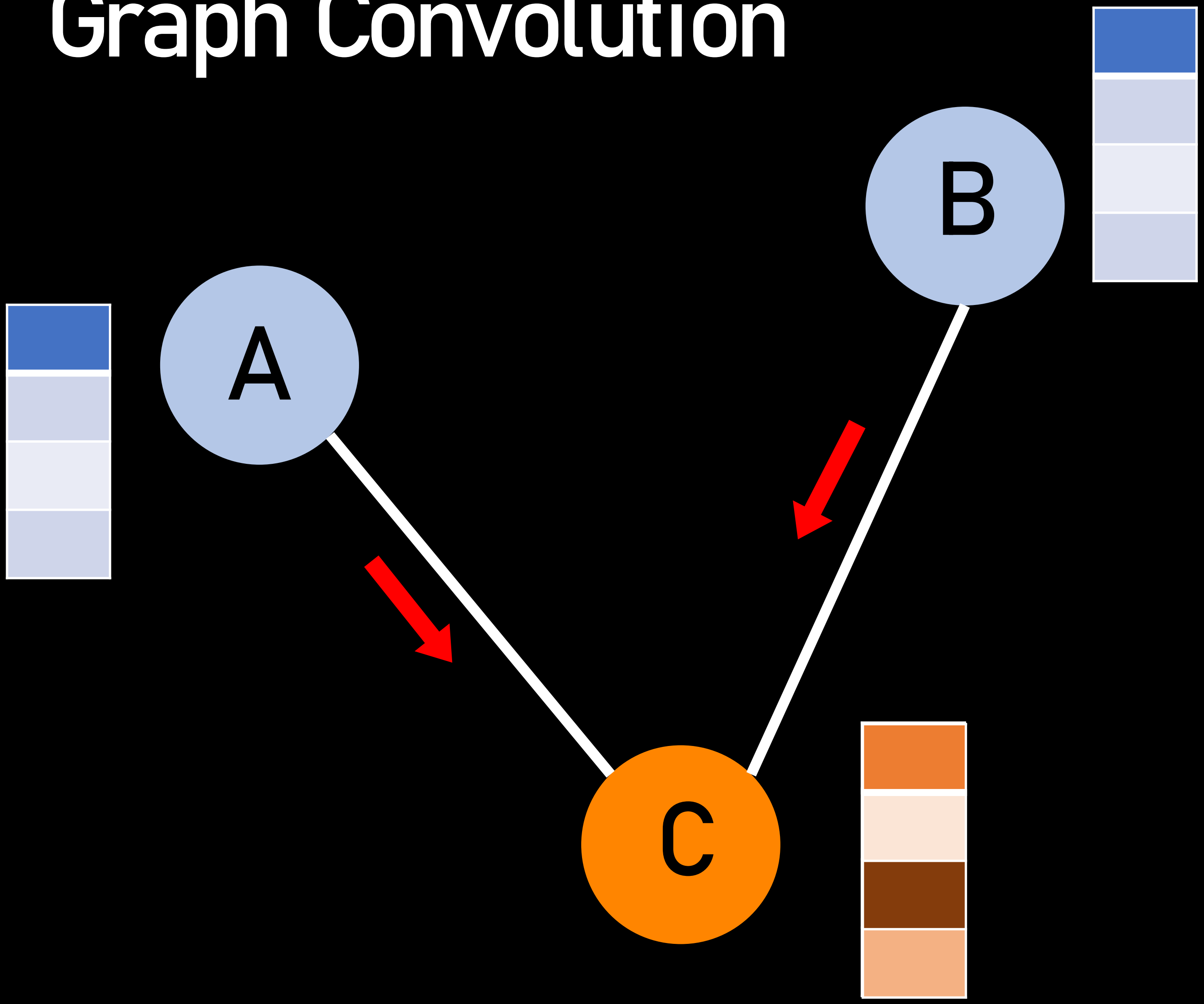
Graph Convolutional Networks (GCNs)

- Variant of Graph Neural Network (GNNs)
- Similar to Convolutional Neural Networks (CNNs)
  - Pixels learn features from neighbouring cells
- GCNs generalise to non-Euclidean, irregular data
- Typical use within ATLAS for *Graph Classification*
  - E.g. GN1 flavour tagging, each jet is graph
- Our idea is to make entire dataset into graph with events as nodes: *Node Classification*



T. N. Kipf, M. Welling: arXiv:1609.02907

Graph Convolution



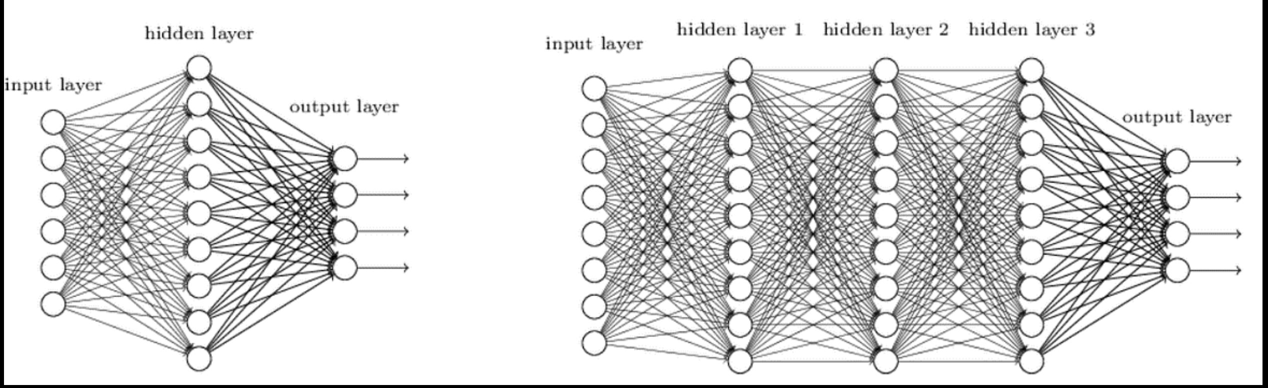
- Vector of kinematic variables for each event "updated" during a graph convolution:

$$C' = \mathrm{conv}(A,B,C)$$

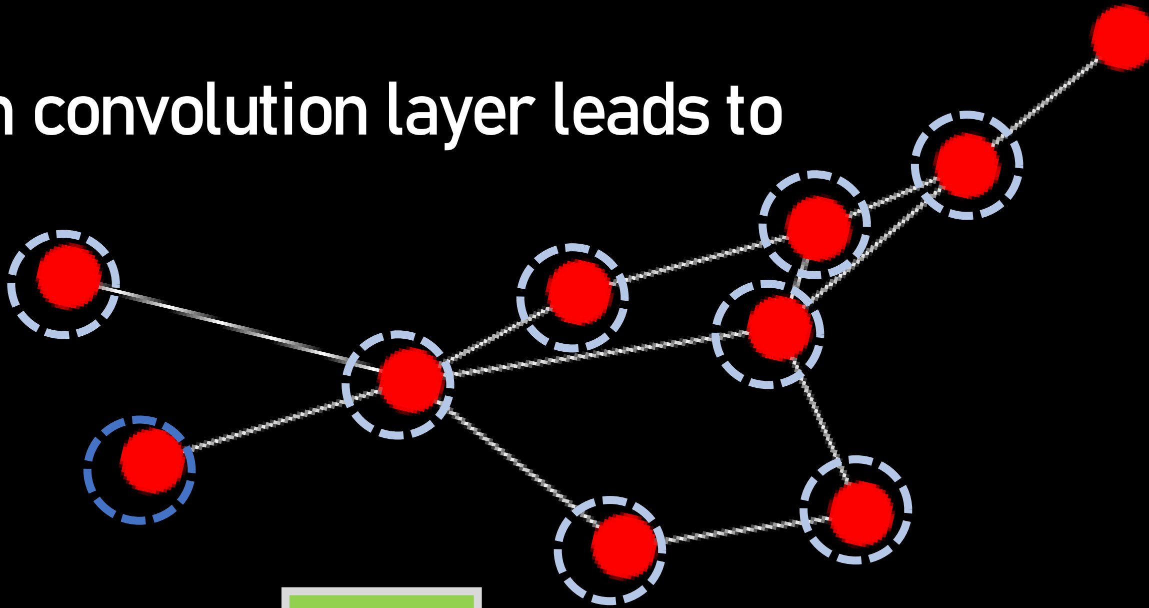
- Convolution operation can be a sum, max, average etc.

- Updates value of event C based on connected events A and B

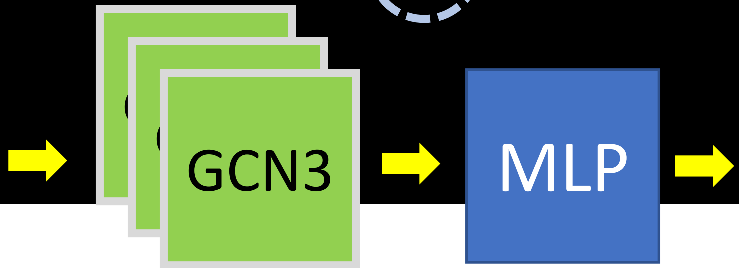
Depth?



- Deep learning based on more layers leads to improve performance
  - Problem eventually becomes computational cost and/or overfitting
- With deeper GCNs, each additional graph convolution layer leads to seeing further away neighbours

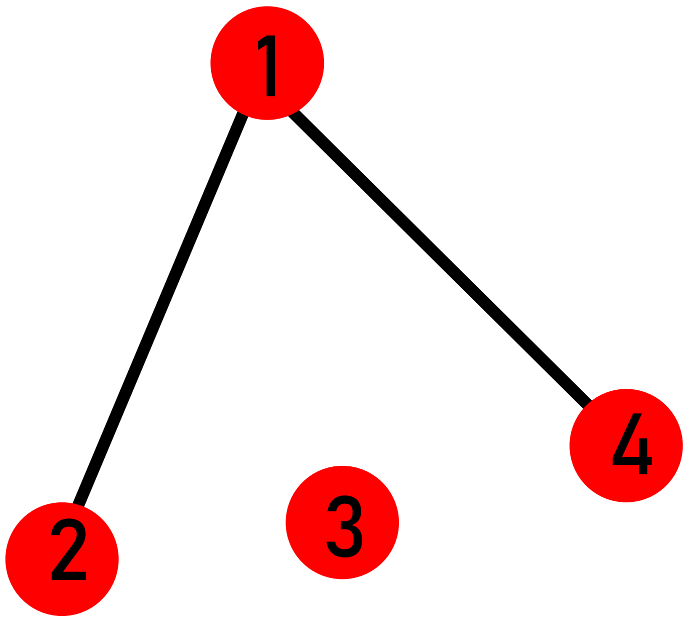


- Still open question: is depth needed?
  - *Over-smoothing*
- Can Multi-hop operators be used instead?



Some things to consider & potential issues

- Which distance metric?
- Which kinematic variables for distance calculation (and weights)?
- Linking length selection
- How to treat MC weights (and negative weights)?
- Scalability (graph Adjacency Matrix *A* scales as N^2 for N nodes)



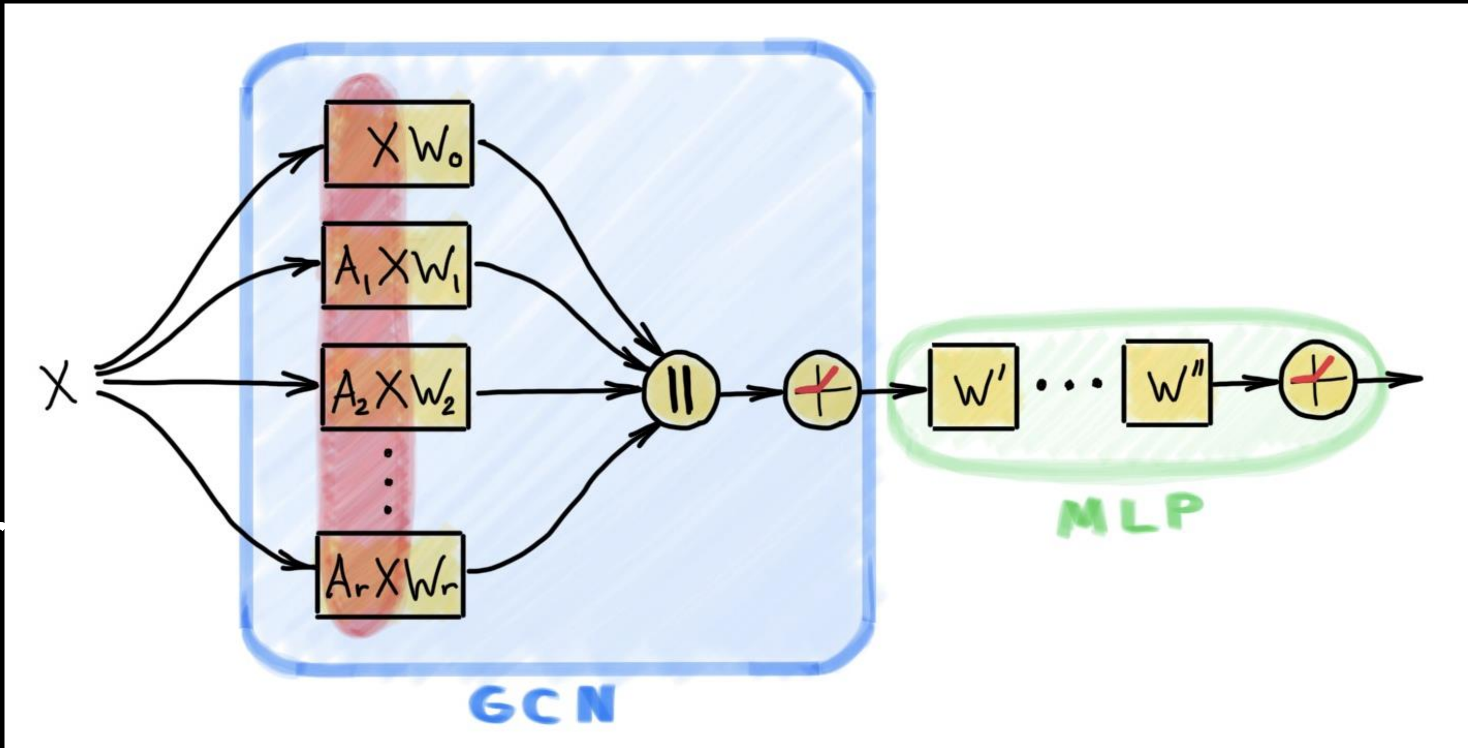
$$A = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

Scalable Inception-like Graph Network (SIGN)

- Incorporates graph convolution step in pre-processing, *before* training:

$$Y = \mathrm{softmax}(\mathrm{ReLU}(\ldots\mathrm{ReLU}(XW_0 \mid A_1XW_1)\, W')\ldots\, W'')$$

- Where A_1X is pre-computed
- Overall complexity ~MLP
- Higher order operators
    - Aggerate further hops
    - Reach further neighbours
- Network can be made deeper
    - Adding more layers to MLP



F. Frasca et al. arXiv:2004.11198

Pre-computation and MC weights

- Never need to hold all of A_1 operator in memory,
  - Can compute a few rows at a time
  - Only need to store A_1X which is updated version of dataset X
- Can apply MC weights in computation

Adjacency A_1

Ws
0.5
3
-2
10
1

1	0	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	1	1	0
0	0	0	0	1

N x N

X

Dataset X

tauPt	MET	Nbjets
20	80	0
40	90	0
100	50	2
30	140	1
80	100	0

N x M

=

Update A_1X

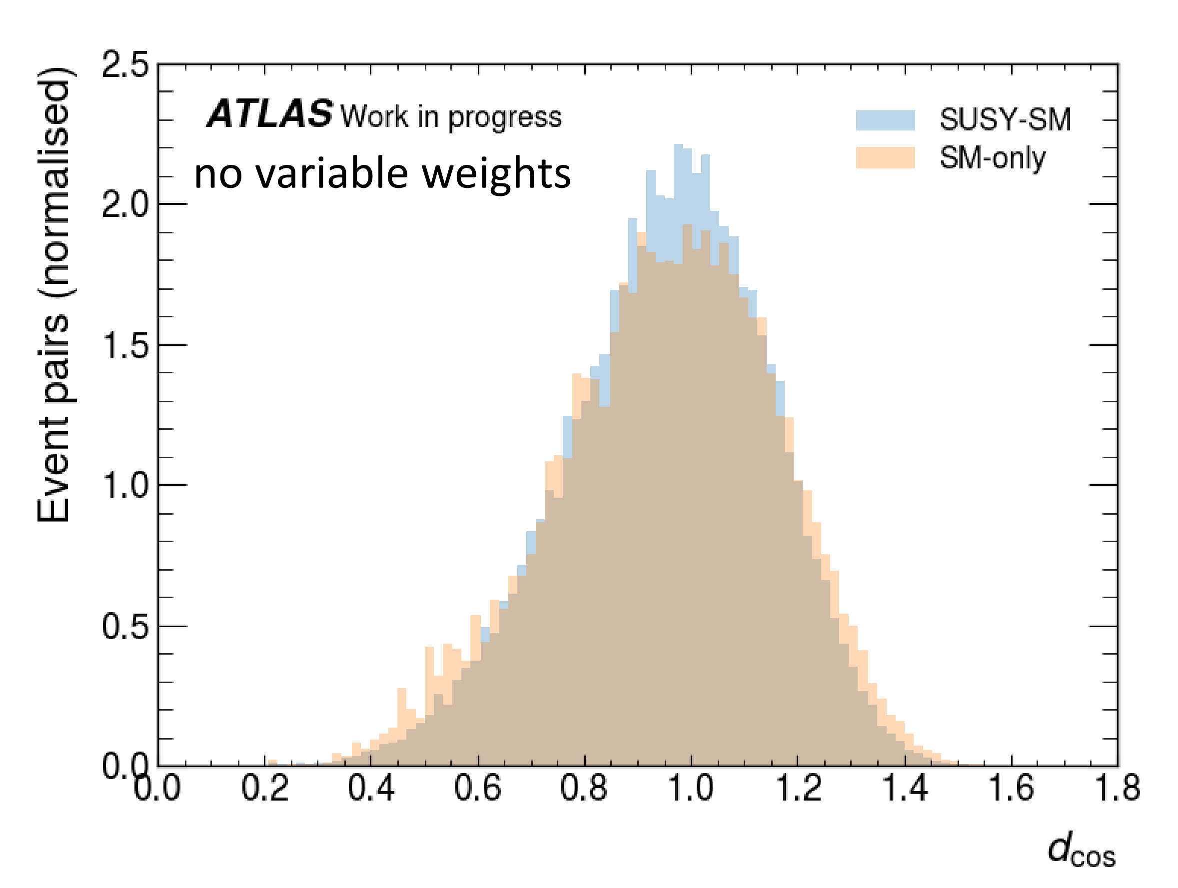
tauPt	MET	Nbjets
(0.5)20 + (10)30	(0.5)80+(10)140	(0.5)0 + (10)1
(3)40 + (-2)100	(3)90 + (-2)50	(3)0 + (-2)2
…	…	…
…	…	…
…	…	…

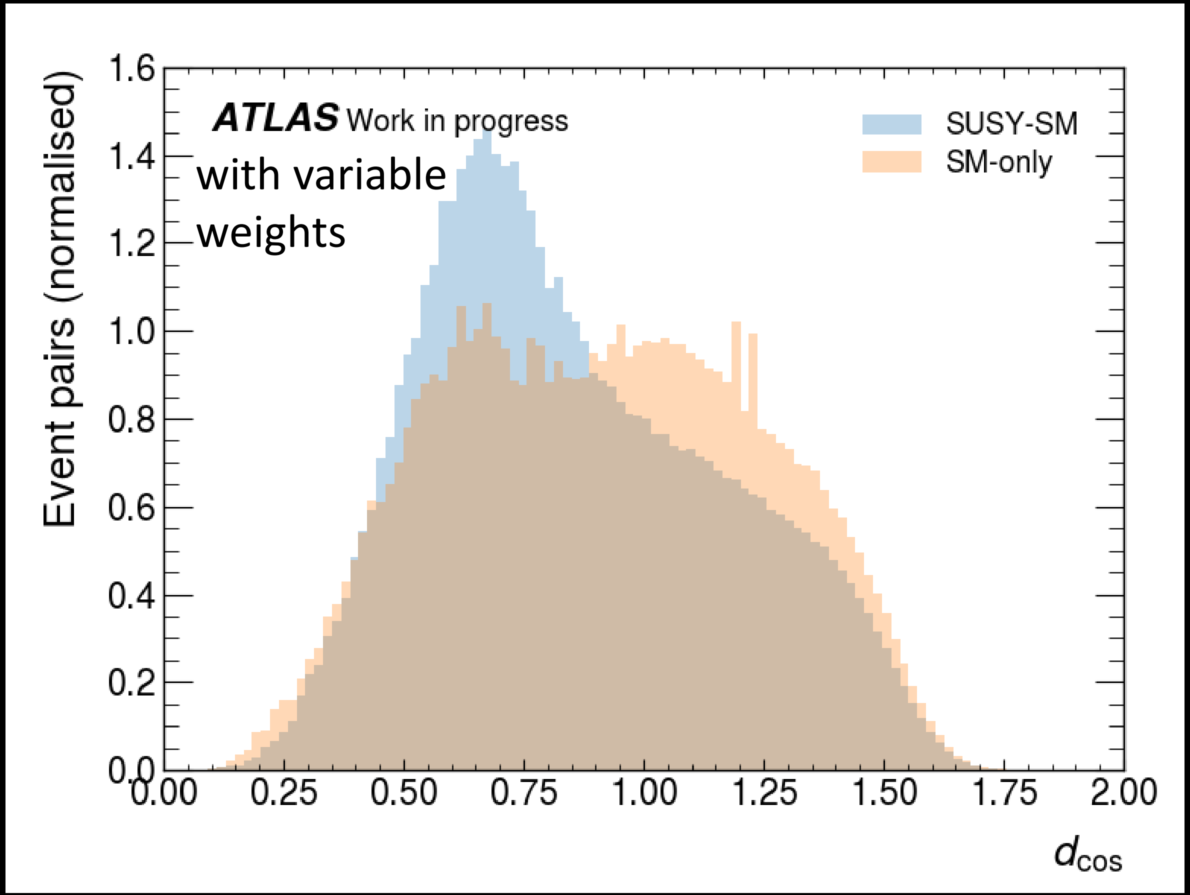
N x M

Distance calculation studies

- So far mainly studied cosine distance for computational reasons
- Weighing variables using *feature importance helped improve separation

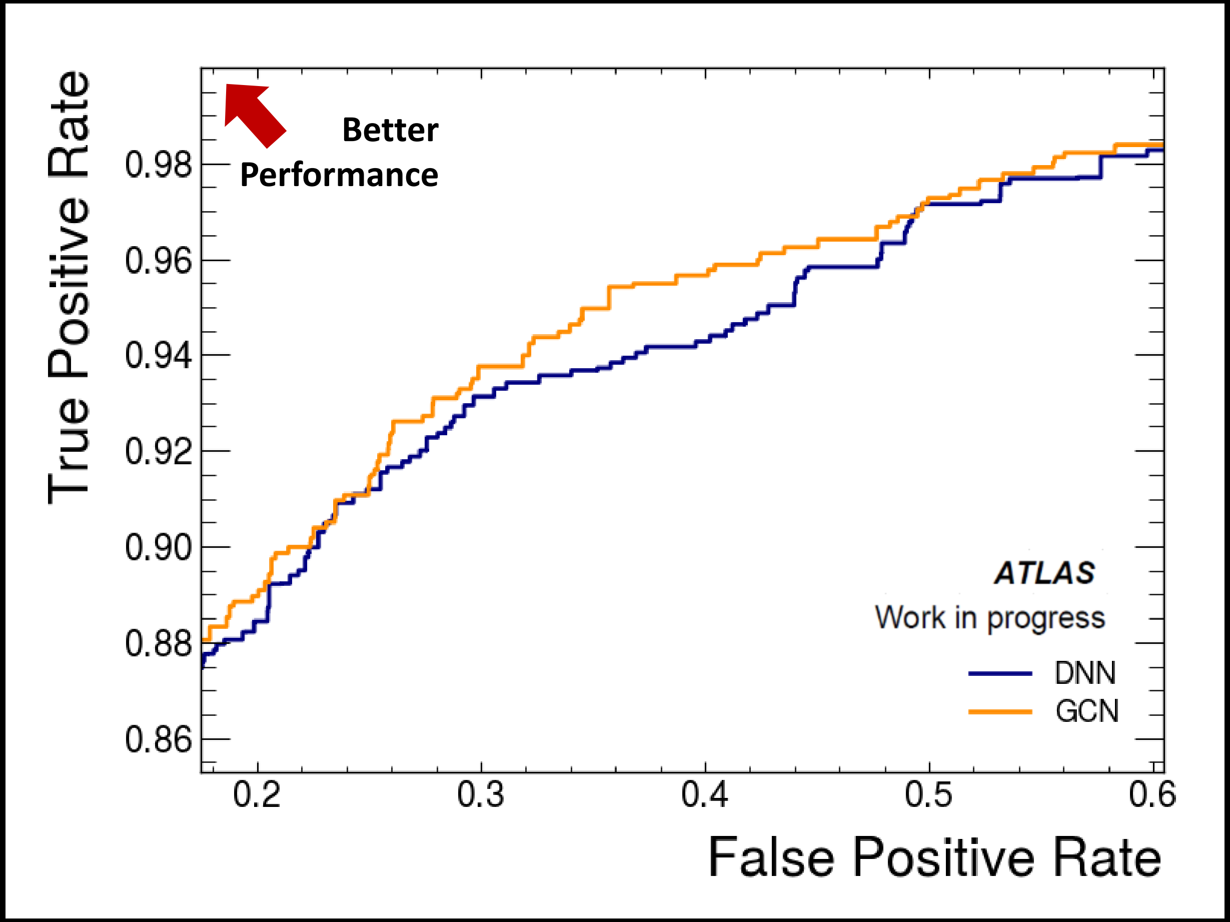
$$d_{\cos} = 1 - \frac{u \cdot v}{\sqrt{u \cdot u}\sqrt{v \cdot v}}$$



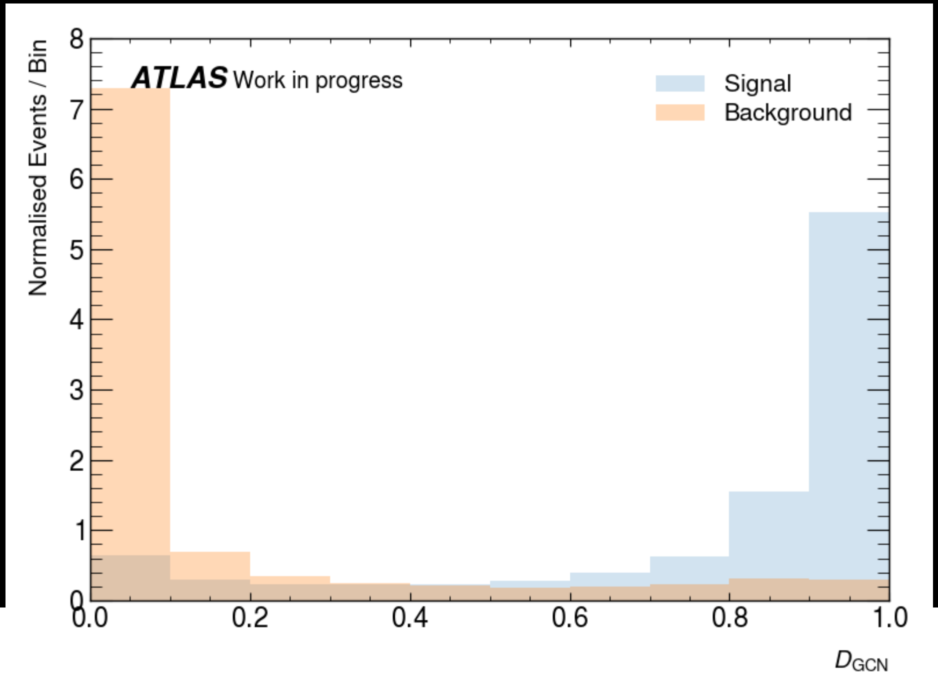


*(based on SHAP scores from regular DNN model, see backup)

Prototype results and analysis



- Initial results look promising
- Marginal improvement in performance over regular DNN
  - Reflecting current separation in distance calculation -> optimise this!



Conclusions & Next Steps

- Trying to improve SUSY/SM discrimination using information about where each events sits in the context of the wider data using GCN
  - Based on pheno paper with graph networks
- Found architecture that scaled well to large dataset, initial prototype results encouraging but could be improved through optimisation
- Test different distance metrics and/or input variable combinations (and weights) to further improve separation
- Any questions/suggestions?

Backup

ML preselection and 'Cut & Count' attempt

- Implemented simple selection with basic clean-up cuts
  - 1L1T
  - lepton p_T > 27 GeV
  - OS
  - b-veto (cleanup ttbar) (*dropped for ML*)
  - Tight ID tau
  - MET>50 (cleanup Z+jets)
- Tried additional cuts based on had-had analysis
  - dPhi(lep,tau) > 1.0
  - dR(lep,tau) < 3.2
  - mT(lep,MET)+mT(tau,MET) > 150 GeV
  - m(lep,tau) > 75 GeV
  - mT2 > 70 GeV

Training details

- Use multiclass DNN with 7 output classes (signal + 6 backgrounds)
- Trained with $m(\tilde{\tau}, \tilde{\chi}_1^0)$ = (100, 40) GeV signal mass point
- Softmax final layer and categorical cross entropy loss
- Validated with 5 folds
- Adam, lr = 0.001
- Use MC weights for each event* and scale signal as whole up to background
- Trained two separate networks
    - 0J network: no jets with pT > 40 GeV
    - wJ network: at least 1 jet with pT > 40 GeV

*with additional treatment for negative weights

Input Variables

Category	Variable	0 jet	>=1 jet
MET	MET	X	X
	TST	X	X
	MET_significance	X	X
tau	tauPt	X	X
	tauEta	X	X
	tauNtracks	X	X
	tauM	X	X
lepton	lep1Pt	X	X
	lep1Eta	X	X
	lep1D0	X	X
	lep1D0Sig	X	X
	lep1Z0	X	X
	lep1Z0SinTheta	X	X
	lep1Flavour	X	X
	lep1Charge	X	X
Delta phi	dphi_met_tst	X	X
	dphi_met_lep	X	X
	dphi_met_tau	X	X
	dphi_met_jet		X
	dphi_tst_lep	X	X
	dphi_tst_tau	X	X
	dphi_tst_jet		X
	dphi_lep_tau	X	X
	dphi_lep_jet		X
	dphi_tau_jet		X

Category	Variable	0 jet	>=1 jet
Delta eta	dEta_lep_tau	X	X
	dEta_lep_jet		X
	dEta_tau_jet		X
Delta R	dR_lep_tau	X	X
	dR_lep_jet		X
	dR_tau_jet		X
angular extra	sum_cos	X	X
	MET_centrality	X	X
	cos(phi1)	X	X
	cos(phi2)	X	X
	cos_theta_star	X	X
balance	tau_lep_balance	X	X
	met_balance	X	X
mass extra	mT_tau_met	X	X
	mT_lep_met	X	X
	mT_sum	X	X
	mCT_tau_lep	X	X
	M_invariant_tau_lep	X	X
	MeffInc40	X	X
jets	jetPt (leading)		X
	jetEta (leading		X
	jetM (leading)		X
	Ht40		X
	n_jets		X
	n_b_jets	X	X

Category	Variable	0 jet	>=1 jet
mT2_{invisible_mass}	mT2_0	X	X
	mT2_10	X	X
	mT2_20	X	X
	mT2_30	X	X
	mT2_40	X	X
	mT2_50	X	X
	mT2_60	X	X
Reco tau	rtau1Pt	X	X
	rtau2Pt	X	X
	dphi_met_rtau1	X	X
	dphi_met_rtau2	X	X
	dphi_tst_rtau1	X	X
	dphi_tst_rtau2	X	X
	dphi_rtau1_rtau2	X	X
	dphi_rtau1_jet		X
	dphi_rtau2_jet		X
	dR_rtau1_rtau2	X	X
	dR_rtau1_jet		X
	dR_rtau2_jet		X
	rtaus_balance	X	X
	M_invariant_rtaus	X	X
	mCT_rtaus	X	X
	Cos_theta_star_rtaus	X	X

Model architecture

wJ

Layer (type)	Output Shape	Param #
normalization (Normalization	(None, 74)	149
dense (Dense)	(None, 256)	19200
batch_normalization (BatchNo	(None, 256)	1024
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 512)	131584
batch_normalization_1 (Batch	(None, 512)	2048
dropout_1 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 256)	131328
batch_normalization_2 (Batch	(None, 256)	1024
dropout_2 (Dropout)	(None, 256)	0
dense_3 (Dense)	(None, 128)	32896
batch_normalization_3 (Batch	(None, 128)	512
dropout_3 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 128)	16512
batch_normalization_4 (Batch	(None, 128)	512
dense_5 (Dense)	(None, 64)	8256
batch_normalization_5 (Batch	(None, 64)	256
dense_6 (Dense)	(None, 48)	3120
batch_normalization_6 (Batch	(None, 48)	192
dense_7 (Dense)	(None, 32)	1568
batch_normalization_7 (Batch	(None, 32)	128
dense_8 (Dense)	(None, 16)	528
batch_normalization_8 (Batch	(None, 16)	64
dense_9 (Dense)	(None, 8)	136
batch_normalization_9 (Batch	(None, 8)	32
dense_10 (Dense)	(None, 7)	63

Total params: 351,132
Trainable params: 348,087
Non-trainable params: 3,045

0J

Layer (type)	Output Shape	Param #
normalization_1 (Normalizati	(None, 57)	115
dense_11 (Dense)	(None, 512)	29696
batch_normalization_10 (Batc	(None, 512)	2048
dropout_4 (Dropout)	(None, 512)	0
dense_12 (Dense)	(None, 256)	131328
batch_normalization_11 (Batc	(None, 256)	1024
dropout_5 (Dropout)	(None, 256)	0
dense_13 (Dense)	(None, 128)	32896
batch_normalization_12 (Batc	(None, 128)	512
dropout_6 (Dropout)	(None, 128)	0
dense_14 (Dense)	(None, 64)	8256
batch_normalization_13 (Batc	(None, 64)	256
dropout_7 (Dropout)	(None, 64)	0
dense_15 (Dense)	(None, 48)	3120
batch_normalization_14 (Batc	(None, 48)	192
dense_16 (Dense)	(None, 32)	1568
batch_normalization_15 (Batc	(None, 32)	128
dense_17 (Dense)	(None, 16)	528
batch_normalization_16 (Batc	(None, 16)	64
dense_18 (Dense)	(None, 8)	136
batch_normalization_17 (Batc	(None, 8)	32
dense_19 (Dense)	(None, 7)	63

Total params: 211,962
Trainable params: 209,719
Non-trainable params: 2,243

SHAP Values

- Used absolute value of SHAP scores to rank feature importance to regular DNN
- Applying as weights to features in cosine distance calculation helped improve distance separation