Reconstruction of Semi-Leptonic Top Anti-top Pair Production with Deep Learning

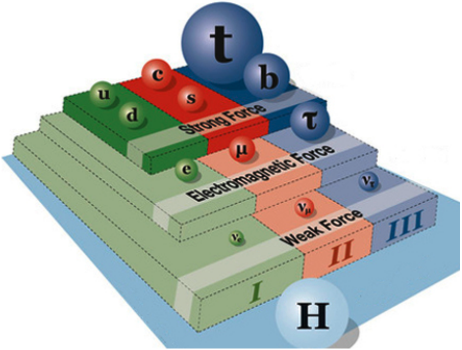
Jenna Chisholm

Supervisor: Alison Lister

THE UNIVERSITY OF BRITISH COLUMBIA
Vancouver, British Columbia, Canada

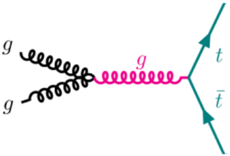
February 2023

- Heaviest known fundamental particle ($m_t \approx 172.5$GeV)
  - First place a new particle could be observed, particularly if it couples to mass
- Extremely short lifetime ($\sim 5 \times 10^{-25}$s)
  - Decays semi-weakly (t→Wb), before hadronization can occur
  - Only place to study properties of a "bare" quark
- Precise measurements enhance our sensitivity to possible beyond SM effects



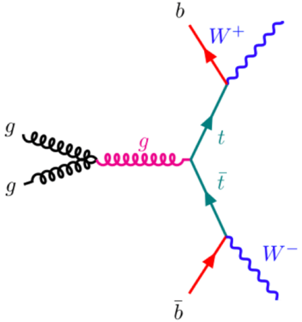
Background

Top-Antitop Pair Production (ttbar)



Background

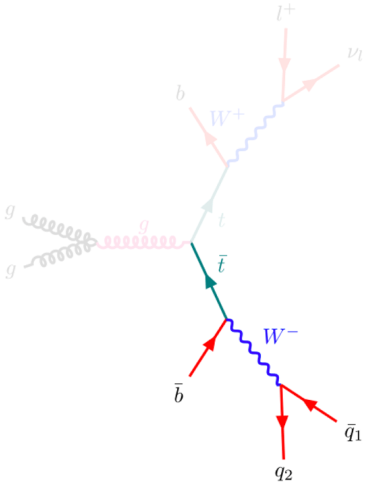
Top-Antitop Pair Production (ttbar)



- Top quark decays to b and W $\sim$99% of the time

Background

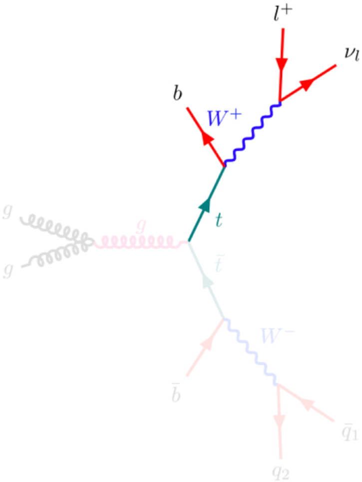
Top-Antitop Pair Production (ttbar)



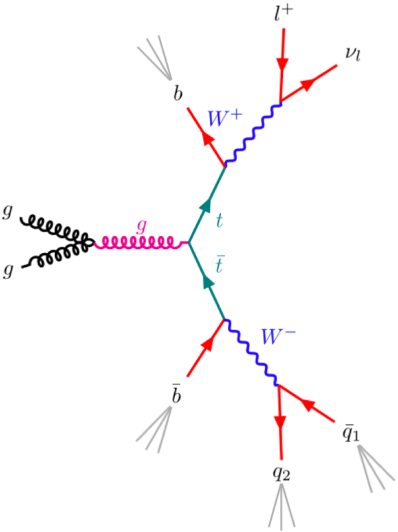
- Top quark decays to b and W $\sim$99% of the time
- W decays **hadronically** with $\sim$70% branching ratio and leptonically with $\sim$30%

Background

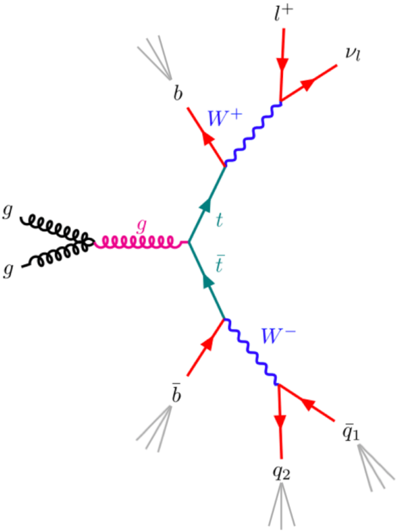
Top-Antitop Pair Production (ttbar)



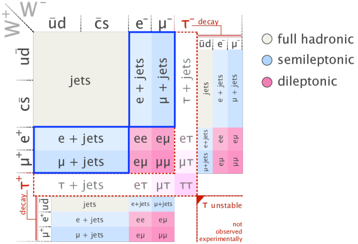
- Top quark decays to *b* and *W* $\sim$99% of the time
- *W* decays hadronically with $\sim$70% branching ratio and **leptonically** with $\sim$30%



- Top quark decays to b and W $\sim$99% of the time
- W decays hadronically with $\sim$70% branching ratio and leptonically with $\sim$30%



- Top quark decays to b and W $\sim$99% of the time
- W decays hadronically with $\sim$70% branching ratio and leptonically with $\sim$30%
- Focus on *semi-leptonic* decays ($\sim$30% branching ratio)



Objective

Algorithms:

- Currently well-established and widely used
- Determines the best permutation of detector-level jets to particle-level jets by:
  - Employing kinematic constraints
  - Sometimes aiming to maximize a likelihood or minimize a chi-squared
  - Assuming a four-jet system
- Reconstruct the top and anti-top 4-vectors from this permutation
- E.g. Kinematic Likelihood Fitter (KLFitter), TtresChi2 (Chi2), and PseudoTop (PT)

Objective

Algorithms:

- Currently well-established and widely used
- Determines the best permutation of detector-level jets to particle-level jets by:
  - Employing kinematic constraints
  - Sometimes aiming to maximize a likelihood or minimize a chi-squared
  - Assuming a four-jet system
- Reconstruct the top and anti-top 4-vectors from this permutation
- E.g. Kinematic Likelihood Fitter (KLFitter), TtresChi2 (Chi2), and PseudoTop (PT)

Deep Neural Networks:

- Determines weights and functions (through training) that will map the typical detector-level objects to the expected parton-level objects
- Could be more precise, more efficient, and less model dependant
- 3 slight variations we're working on: TRecNet, TRecNet+ttbar, and TRecNet+ttbar+JetPretrain

Objective

Algorithms:

- Currently well-established and widely used
- Determines the best permutation of detector-level jets to particle-level jets by:
  - Employing kinematic constraints
  - Sometimes aiming to maximize a likelihood or minimize a chi-squared
  - Assuming a four-jet system
- Reconstruct the top and anti-top 4-vectors from this permutation
- E.g. Kinematic Likelihood Fitter (KLFitter), TtresChi2 (Chi2), and PseudoTop (PT)

Deep Neural Networks:

- Determines weights and functions (through training) that will map the typical detector-level objects to the expected parton-level objects
- Could be more precise, more efficient, and less model dependant
- 3 slight variations we're working on: TRecNet, TRecNet+ttbar, and TRecNet+ttbar+JetPretrain

Goal:

Design a deep neural network to reconstruct $t\bar{t}$ better than current algorithms!

Setup

"Truth"

- Generate hard-scattering with *POWHEG* (parton-level)
- Simulate parton shower and hadronization with *Pythia8* (particle-level)

Setup

"Truth"

- Generate hard-scattering with *POWHEG* (parton-level)
- Simulate parton shower and hadronization with *Pythia8* (particle-level)

$\rightarrow$

"Measured" / "Reco Input"

- Detector response simulated by *Geant4* (detector/reco-level)
  - Jets: (p_T, η, ϕ, E), b_{tag}
  - Lepton: $(p_{T_{lep}}, \eta_{lep}, \phi_{lep})$
  - Missing Transverse Energy: E_T, ϕ_{E_T}

Setup

"Truth"

- Generate hard-scattering with *POWHEG* (parton-level)
- Simulate parton shower and hadronization with *Pythia8* (particle-level)

→

"Measured" / "Reco Input"

- Detector response simulated by *Geant4* (detector/reco-level)
  - Jets: (p_T,η,ϕ,E), b_{tag}
  - Lepton: ($p_{T_{lep}}$,η_{lep},ϕ_{lep})
  - Missing Transverse Energy: E_T,ϕ_{E_T}

↓

"Predictions" / "Reco Output"

- Previous fitting algorithms vs. Top Reconstruction Neural Networks
  - Hadronic Top: ($p_{T_{t_h}}$,η_{t_h},ϕ_{t_h},m_{t_h})
  - Leptonic Top: ($p_{T_{t_l}}$,η_{t_l},ϕ_{t_l},m_{t_l})
  - ttbar: ($p_{T_{t\bar{t}}}$,$\eta_{t\bar{t}}$,$\phi_{t\bar{t}}$,$m_{t\bar{t}}$)

Setup

"Truth"

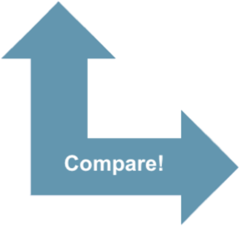
- Generate hard-scattering with *POWHEG* (parton-level)
- Simulate parton shower and hadronization with *Pythia8* (particle-level)

"Measured" / "Reco Input"

- Detector response simulated by *Geant4* (detector/reco-level)
  - Jets: (p_T, η, ϕ, E), b_{tag}
  - Lepton: $(p_{T_{lep}}, \eta_{lep}, \phi_{lep})$
  - Missing Transverse Energy: E_T, ϕ_{E_T}

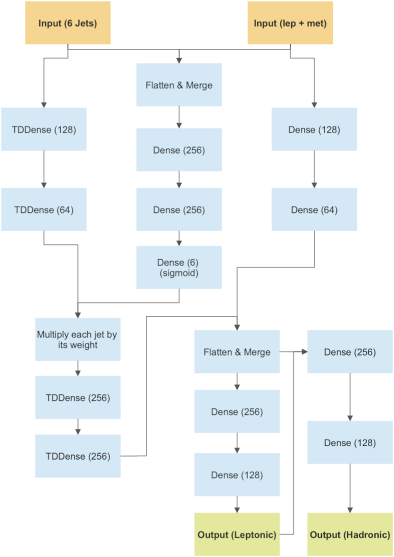
"Predictions" / "Reco Output"

- Previous fitting algorithms vs. Top Reconstruction Neural Networks
  - Hadronic Top: $(p_{T_{t_h}}, \eta_{t_h}, \phi_{t_h}, m_{t_h})$
  - Leptonic Top: $(p_{T_{t_l}}, \eta_{t_l}, \phi_{t_l}, m_{t_l})$
  - ttbar: $(p_{T_{t\bar{t}}}, \eta_{t\bar{t}}, \phi_{t\bar{t}}, m_{t\bar{t}})$



Architectures

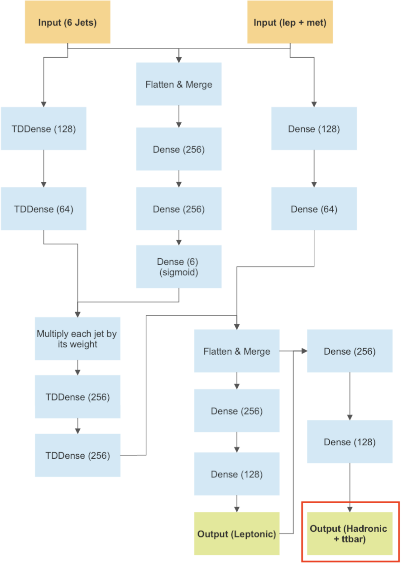
Model #1: TRecNet



- Input: pre-processed jets (6) and other (lep, met) variables
- First attempts to learn which jets are relevant to $t\bar{t}$ process
- Predicts leptonic 4-vectors (t_l, W_l) first, since their classification is easier, and then uses this information to help inform predictions on the hadronic 4-vectors (t_h, W_h)

Architectures

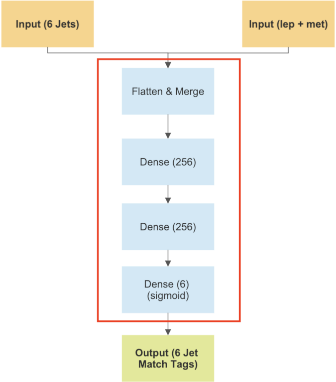
Model #2: TRecNet+ttbar

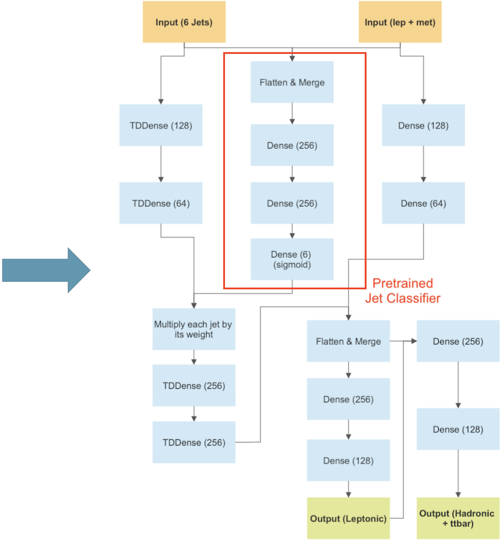


- Input: pre-processed jets (6) and other (lep, met) variables
- First attempts to learn which jets are relevant to $t\bar{t}$ process
- Predicts leptonic 4-vectors (t_l, W_l) first, since their classification is easier, and then uses this information to help inform predictions on the hadronic 4-vectors (t_h, W_h) and $t\bar{t}$ variables

Architectures

Model #3: TRecNet+ttbar+JetPretrain





- Obtain weights for the jet classifier using matched jet tags
- *Freeze these weights into this section of the overall model*

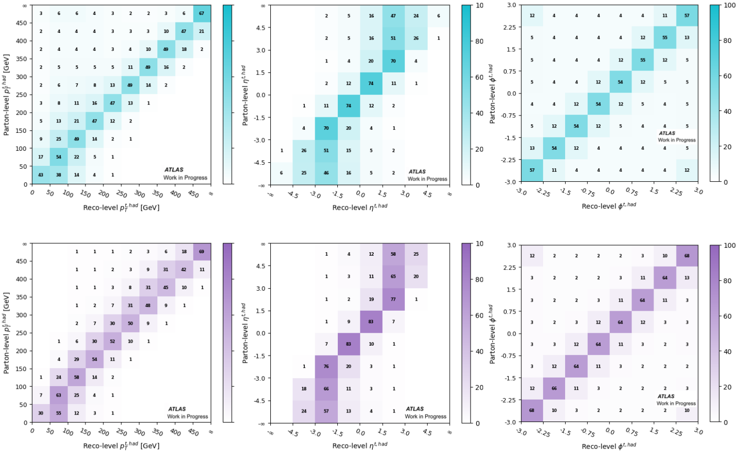
Hadronic Top Results

Response Matrices

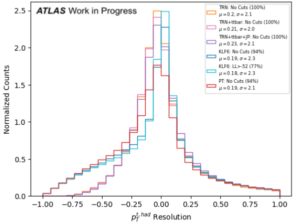
(a) Hadronic Top p_T (b) Hadronic Top η (c) Hadronic Top ϕ

KLFitter (6 jets flag) ($LL > -52$ $\rightarrow$ 77%):

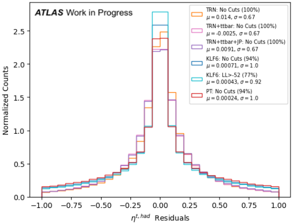
TRecNet+ttbar +JetPretrain (No Cuts $\rightarrow$ 100%):



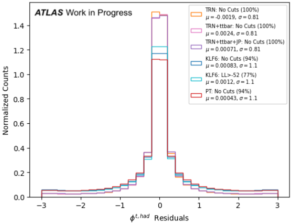
TRecNet+ttbar+JP is more diagonal than KLFitter $\Longrightarrow$ improved precision!



(a) Hadronic Top p_T



(b) Hadronic Top η



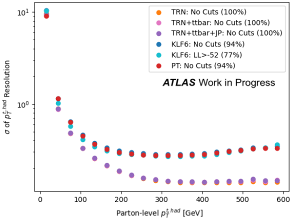
(c) Hadronic Top ϕ

TRecNet+ttbar+JP:	TRecNet+ttbar+JP:	TRecNet+ttbar+JP:
$\mu = 0.23, \sigma = 2.1$	$\mu = 0.0091, \sigma = 0.67$	$\mu = 0.00071, \sigma = 0.81$
KLF6($LL > -52$: 77%):	**KLF6($LL > -52$: 77%):**	**KLF6($LL > -52$: 77%):**
$\mu = 0.18, \sigma = 2.3$	$\mu = 0.00043, \sigma = 0.92$	$\mu = 0.0012, \sigma = 1.1$

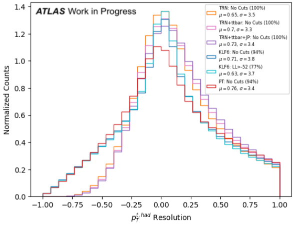
TRecNet+ttbar+JP is more narrow and less skewed than KLFitter $\implies$ improved precision!

Hadronic Top Results

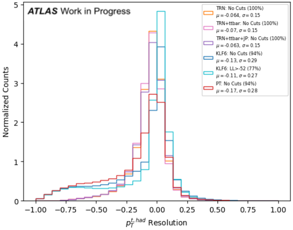
p_T Resolutions at Different Momenta



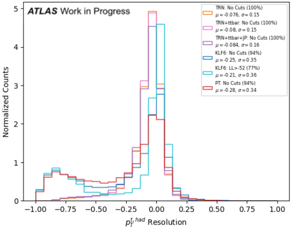
- Neural networks completely remove the extra bump at high p_T!
  - Jets become more difficult to resolve at high p_T (events occur with more or less than jets)
  - Neural networks use all jet info, but algorithms use only best permutation of 4 out of 6



$p_T < 100$ GeV



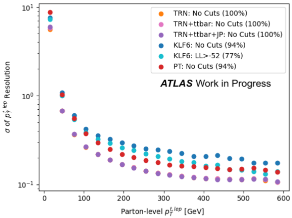
$250 < p_T < 500$ GeV



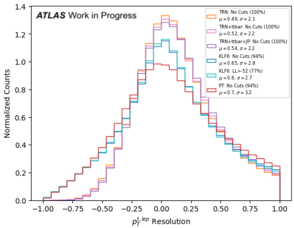
$p_T > 500$ GeV

Leptonic Top Results

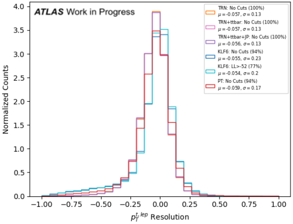
p_T Resolutions at Different Momenta



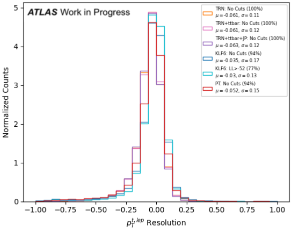
- No extra bump at high p_T on leptonic side!
  - Only one b-jet to resolve
- But neural networks still have better resolution over range of p_T



$p_T < 100$ GeV

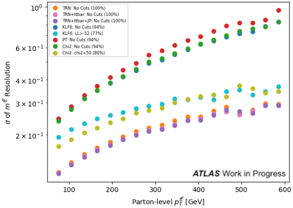


$250 < p_T < 500$ GeV

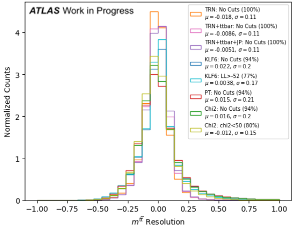


$p_T > 500$ GeV

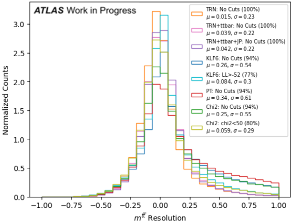
$m_{t\bar{t}}$ Resolutions at Different Momenta



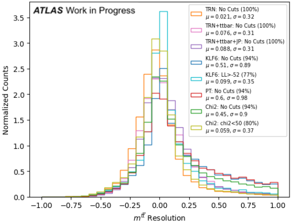
- Neural networks improve upon reconstruction of mass of $t\bar{t}$ system
- Adding $t\bar{t}$ variables to the neural network helped improve precision for $m_{t\bar{t}}$



$p_T < 100$ GeV



$250 < p_T < 500$ GeV



$p_T > 500$ GeV

Conclusions and Outlook

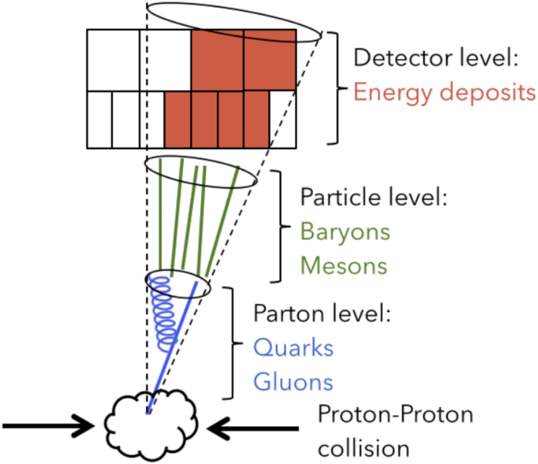
- Advantages of the neural networks:
  - Appear to improve upon results of from likelihood-based algorithms
  - Perform more efficiently
  - Flexibility to handle events with more or less than 4 jets (and thus performs better than previous methods in the boosted topology)
- Future possibilities and outlook:
  - Widen model to consider more jets (e.g. 7 or 8)
  - Unfreeze the jet pre-training weights to fine-tune TRecNet+ttbar+JetPretrain model
  - Measure model dependency
  - Include systematics to obtain a more quantitative measure of the neural network's improvement

Thanks to . . .

- Dr. Alison Lister
- Dr. Zhengcheng Tao
- Tao Zhang
- The ATLAS Collaboration
- NSERC

Background

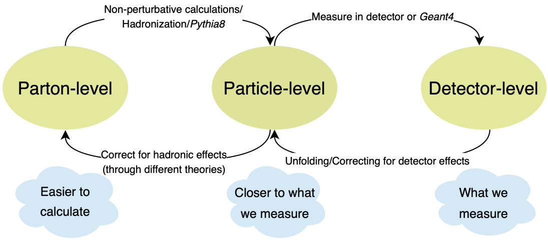
Parton-level vs. Particle-level vs. Detector-level

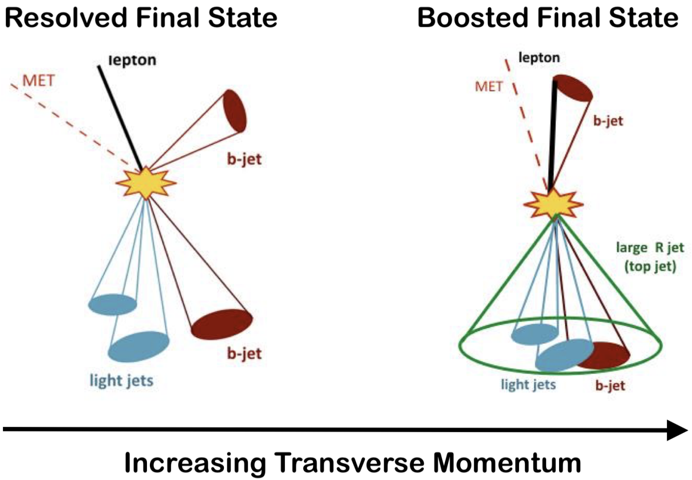


Background

Parton-level vs. Particle-level vs. Detector-level

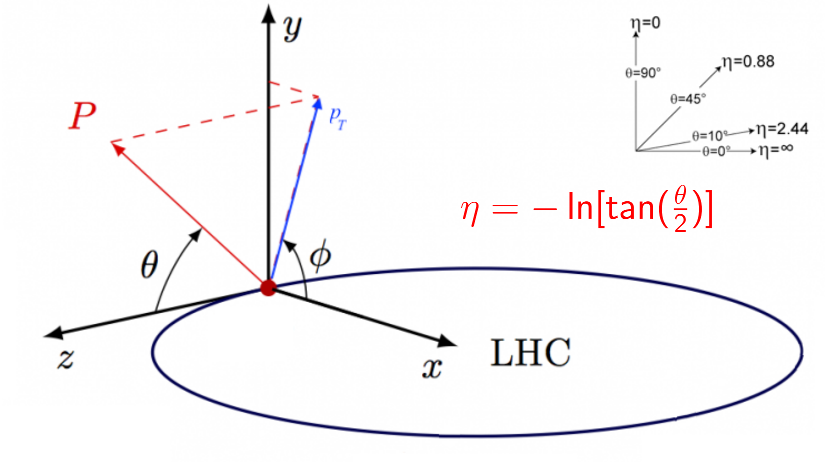
- **Parton-level**: Only includes perturbative matrix element calculations
  - E.g. hard scattering events generated by *POWHEG*
- **Particle-level**: Includes both perturbative and non-perturbative matrix element calculations
  - E.g. parton shower/hadronization components handled by *Pythia8*
- **Detector-level**: What we measure
  - E.g. data or simulated data from *Geant4*
  - The top reconstruction algorithms we're using are at this level




Resolved Final State
Boosted Final State
lepton
MET
b-jet
b-jet
light jets
lepton
MET
b-jet
large R jet
(top jet)
light jets
b-jet
Increasing Transverse Momentum

Background

Coordinate System



$$\eta = -\ln[\tan(\frac{\theta}{2})]$$

Reconstruction Algorithms

Kinematic Likelihood Fitter (KLFitter)

- Best permutation of jets determined using kinematics and likelihood calculations:

$$\mathcal{L} = \mathcal{B}(m_{q_1 q_2 q_3}|m_t, \Gamma_t) \cdot \mathcal{B}(m_{q_1 q_2}|m_W, \Gamma_W) \cdot \mathcal{B}(m_{q_4 \ell \nu}|m_t, \Gamma_t) \cdot \mathcal{B}(m_{\ell \nu}|m_W, \Gamma_W) \cdot \prod_{i=1}^{4} W_{\text{jet}}(E_{\text{jet},i}^{\text{meas}}|E_{\text{jet},i}) \cdot W_{\ell}(E_{\ell}^{\text{meas}}|E_{\ell}) \cdot W_{\text{miss}}(E_x^{\text{miss}}|p_x^{\nu}) \cdot W_{\text{miss}}(E_y^{\text{miss}}|p_y^{\nu})$$

  - Breit-Wigner terms ($\mathcal{B}$) $\rightarrow$ quantify agreement of known masses with measured decay products
  - Transfer function terms (W) $\rightarrow$ quantify agreement of fitted energies and missing transverse momentum components with measured values (detector-specific and representative of experimental resolutions)
- Likelihood calculated for each possible association of detector-level jets to particle-level jets, where m_t, $E_{jet,i}$, E_ℓ, and $\vec{p}_\nu$ are treated as parameters varied to maximize the likelihood
- Retain permutation with highest likelihood (called the "best permutation")
- Can make cuts on $\log \mathcal{L}$ to separate well- and poorly-reconstructed events

Breit-Wigner Function:

$$\mathcal{B}(E|M,\Gamma) = \frac{k}{(E^2 - M^2)^2 + M^2\Gamma^2}$$

where,

$$k = \frac{2\sqrt{2}M\Gamma\gamma}{\pi\sqrt{M^2+\gamma}}$$

and

$$\gamma = \sqrt{M^2(M^2+\Gamma^2)}$$

Transfer Function:

$$W(E) = \frac{Y(E)}{X(E)}\bigg|_{\text{initial conditions} = 0}$$

where,

$$Y = \text{laplace transform of output}$$

and

$$X = \text{laplace transform of input}$$

- Best permutation of jets determined using kinematics and chi-squared calculation:

$$\chi^2 = \left[\frac{m_{jj} - m_{W_h}}{\sigma_{W_h}}\right]^2 + \left[\frac{m_{jjb} - m_{jj} - m_{t_h - W_h}}{\sigma_{t_h - W_h}}\right]^2 + \left[\frac{m_{b\ell\nu} - m_{t_\ell}}{\sigma_{t_\ell}}\right]^2 + \left[\frac{(p_{T,jjb} - p_{T,b\ell\nu}) - (p_{T,t_h} - p_{T,t_\ell})}{\sigma_{p_{T,t_h} - p_{T,t_\ell}}}\right]^2$$

  - Constraint on dijet mass to form hadronic W
  - Constraint on three jets to form hadronic top – contribution of hadronic W subtracted to decouple first two terms, since m_{jj} and m_{jjb} are highly correlated
  - Constraint on remaining jet, lepton and neutrino (met) to form leptonic top
  - Constraint on transverse momentum balance between the two top quarks (p_T should be similar, as expected in a resonance)
- Expected values of parameters m_{W_h}, $m_{t_h - W_h}$, m_{t_ℓ}, $p_{T,t_h} - p_{T,t_\ell}$ as well as their uncertainties σ_{W_h}, $\sigma_{t_h - W_h}$, σ_{t_ℓ}, $\sigma_{p_{T,t_h} - p_{T,t_\ell}}$ are obtained from the simulated Z' events by matching reconstructed objects to truth partons
- Can make cuts on χ^2 to separate well- and poorly-reconstructed events

Reconstruction Algorithms

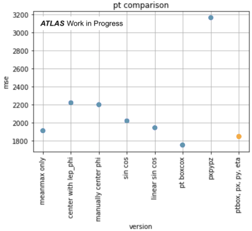
PseudoTop

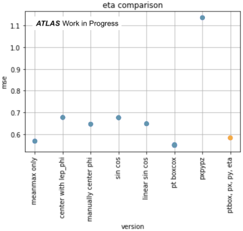
- Uses lepton, jet, and missing transverse energy measurements, as well as known mass of W boson
- Only two b-tagged jets with highest p_T are considered part of the system

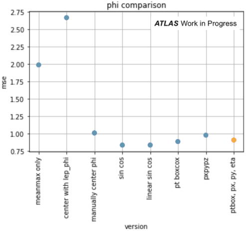
Algorithm:

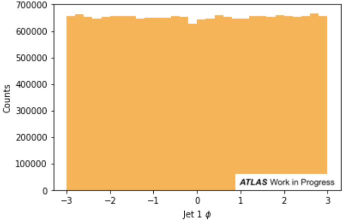
1. Reconstructs neutrino 4-momentum
   - p_x and p_y obtaining from met
   - p_z calculated by conservation of momentum
2. Reconstruct leptonic W from lepton and neutrino
3. Reconstruct leptonic top from leptonic W and b-tagged jet closest in $\Delta R = \sqrt{\Delta\phi^2 + \Delta\eta^2}$ to lepton
4. Reconstruct hadronic W from the two light-flavoured jets whose invariant mass is closest to mass of W boson
5. Reconstruct hadronic top from hadronic W and remaining b-tagged jet

- Model performance was evaluated on validation data using mean-squared error ($mse = \langle truth - prediction \rangle^2$)
- Mean/variance scaling $\left(x_i^{\text{scaled}} = \frac{x_i - \bar{x}}{\sigma(x)}\right)$ vs. mean/max scaling $\left(x_i^{\text{scaled}} = \frac{x_i - \bar{x}}{\max(|x|)}\right)$
  - Standard procedure for allowing the network to focus on each variable equally
- Encoding ϕ with $\sin(\phi)$ and $\cos(\phi)$ vs. triangle wave of $\sin(\phi)$ and $\cos(\phi)$ vs. p_x and p_y
  - Former two produced edge peaks that the network has trouble predicting
- Boxcox transformation of p_T $\left(p_T = \frac{p_T^{\lambda} - 1}{\lambda}\right)$ vs. $(p_x,\ p_y)$ vs. p_T
  - Boxcox did better on average, but poorly reconstructed low p_T events
  - p_x and p_y difficult to predict, resulting in large compounding error for p_T

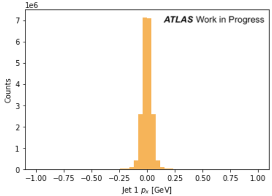


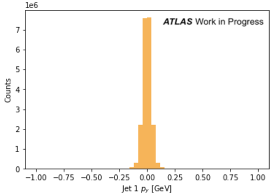






ϕ encoding (with mean/max scaling)





- Final procedure:
  - Encode ϕ_T with $\sin(\phi_T)$, $\cos(\phi_T)$ and all other ϕ with p_x and p_y
  - All inputs (except b_{tag}) undergo mean/max scaling
  - Model predicts (p_T, p_x, p_y, η, m) for top quarks and Ws in mean/max scale
  - Invert mean/max scaling and ϕ encoding to return predictions to original scale

Training Features

Loss Function

Training Feature	TRecNet Models	Jet Pre-training
Loss Function	Mean absolute error	Binary cross entropy
Optimizer	Adam	Adam
Learning Rate	Polynomial decaying from 10^{-3} to 5×10^{-5} with power of 0.25 and decay steps of 10000	Polynomial decaying from 10^{-2} to 5×10^{-4} with power of 0.25 and decay steps of 10000
Activation Function	ReLU excpet for one sigmoid layer and linear output layer	ReLU excpet for sigmoid output layer
Regularization	Early stopping (monitor=val_loss, patience=10)	Early stopping (monitor=val_loss, patience=10)
Events, Batch Size	∼23 Million, 1000	∼23 Million, 1000

- Loss function: quantifies error for current state of model – want to change weights to reduce this loss on next evaluation
- E.g. Binary cross entropy loss function:
  - Default loss function for binary classification problems
  - Calculates a score between $[0, 1]$ that summarizes average difference between true and predicted, and tries to minimize this score through training
  - Used for jet-pretraining model
- E.g. Mean absolute error (MAE) loss function:
  - Calculates average absolute difference between true and predicted
  - Often most appropriate in regression problems where target distributions are mostly Gaussian but may have outliers, since it punishes larger mistakes from outliers less harshly than, for example, MSE
  - Used for TRecNet models

Training Features

Optimizer

Training Feature	TRecNet Models	Jet Pre-training
Loss Function	Mean absolute error	Binary cross entropy
Optimizer	Adam	Adam
Learning Rate	Polynomial decaying from 10^{-3} to 5×10^{-5} with power of 0.25 and decay steps of 10000	Polynomial decaying from 10^{-2} to 5×10^{-4} with power of 0.25 and decay steps of 10000
Activation Function	ReLU excpet for one sigmoid layer and linear output layer	ReLU excpet for sigmoid output layer
Regularization	Early stopping (monitor=val_loss, patience=10)	Early stopping (monitor=val_loss, patience=10)
Events, Batch Size	∼23 Million, 1000	∼23 Million, 1000

- Optimizer: Method or algorithm by which we change weights of network in order to locate minima of loss function
- E.g. Stochastic gradient descent (SGD):
  - Estimates gradient of loss function with randomly selected subset of data
  - Uses estimated gradient to choose direction to move in search space (with step size determined by learning rate)
- E.g. Adam:
  - Particular type of SGD where learning rate is non-static – individual adaptive learning rates are computed for different parameters from estimates of first and second moments of the gradients
  - Used for TRecNet models and jet pre-training

Training Features

Learning Rate

Training Feature	TRecNet Models	Jet Pre-training
Loss Function	Mean absolute error	Binary cross entropy
Optimizer	Adam	Adam
Learning Rate	Polynomial decaying from 10^{-3} to 5×10^{-5} with power of 0.25 and decay steps of 10000	Polynomial decaying from 10^{-2} to 5×10^{-4} with power of 0.25 and decay steps of 10000
Activation Function	ReLU excpet for one sigmoid layer and linear output layer	ReLU excpet for sigmoid output layer
Regularization	Early stopping (monitor=val_loss, patience=10)	Early stopping (monitor=val_loss, patience=10)
Events, Batch Size	$\sim$23 Million, 1000	$\sim$23 Million, 1000

- Learning rate: Step size that optimization algorithm uses at each iteration to move towards the minima
  - Parameter that can be fine-tuned to optimize model performance
  - Can modulate how learning rate changes over training
- E.g. Polynomial decay rate:
  - Begin with larger learning rate $\rightarrow$ take larger steps and train faster
  - Gradually move to smaller learning rate $\rightarrow$ take smaller steps and fine-tune optimization
  - Used for TRecNet and jet pre-training (which slight differences)

Training Features

Activation Function

Training Feature	TRecNet Models	Jet Pre-training
Loss Function	Mean absolute error	Binary cross entropy
Optimizer	Adam	Adam
Learning Rate	Polynomial decaying from 10^{-3} to 5×10^{-5} with power of 0.25 and decay steps of 10000	Polynomial decaying from 10^{-2} to 5×10^{-4} with power of 0.25 and decay steps of 10000
Activation Function	ReLU excpet for one sigmoid layer and linear output layer	ReLU excpet for sigmoid output layer
Regularization	Early stopping (monitor=val_loss, patience=10)	Early stopping (monitor=val_loss, patience=10)
Events, Batch Size	∼23 Million, 1000	∼23 Million, 1000

- Activation function: Defines how weighted sum of input to a node is transformed to output from that node
  - Allows network to handle more complex patterns and non-linear problems → large impact on capability and performance of network
  - Can have different activation functions for different layers
- E.g. ReLU (Rectified Linear Function): $\max(0, x)$
  - Popular for hidden layers
  - Easy to implement, quick, computationally light, and less susceptible to the vanishing gradient problem
  - Used for almost all of our hidden layers
- E.g. Sigmoid (or Logistic) Function: $1/(1 + e^{-x})$
  - Popular for hidden and output layers
  - Use for output from jet classifier

Training Features

Regularization

Training Feature	TRecNet Models	Jet Pre-training
Loss Function	Mean absolute error	Binary cross entropy
Optimizer	Adam	Adam
Learning Rate	Polynomial decaying from 10^{-3} to 5×10^{-5} with power of 0.25 and decay steps of 10000	Polynomial decaying from 10^{-2} to 5×10^{-4} with power of 0.25 and decay steps of 10000
Activation Function	ReLU excpet for one sigmoid layer and linear output layer	ReLU excpet for sigmoid output layer
Regularization	Early stopping (monitor=val_loss, patience=10)	Early stopping (monitor=val_loss, patience=10)
Events, Batch Size	∼23 Million, 1000	∼23 Million, 1000

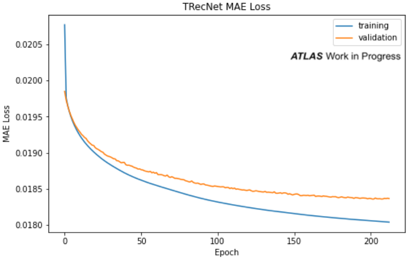
- Regularization: Techniques to prevent over- or under-fitting
- E.g. Early stopping (monitor=val_loss,patience=10):
  - End training after 10 epochs of no improvement in loss for the validation data
  - Used for TRecNet and jet pre-training

Training Features

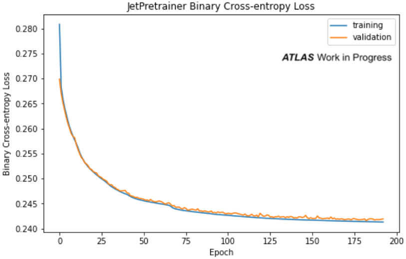
Events and Batch Size

Training Feature	TRecNet Models	Jet Pre-training
Loss Function	Mean absolute error	Binary cross entropy
Optimizer	Adam	Adam
Learning Rate	Polynomial decaying from 10^{-3} to 5×10^{-5} with power of 0.25 and decay steps of 10000	Polynomial decaying from 10^{-2} to 5×10^{-4} with power of 0.25 and decay steps of 10000
Activation Function	ReLU excpet for one sigmoid layer and linear output layer	ReLU excpet for sigmoid output layer
Regularization	Early stopping (monitor=val_loss, patience=10)	Early stopping (monitor=val_loss, patience=10)
Events, Batch Size	$\sim$23 Million, 1000	$\sim$23 Million, 1000

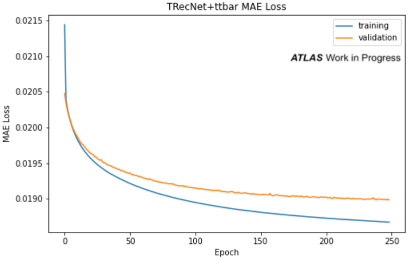
- Events: 33 million
  - 70% to training
  - 15% to validation
  - 15% to testing
- Batch Size: Number of events processed before model is updated
  - Used batch size = 1000 for all models



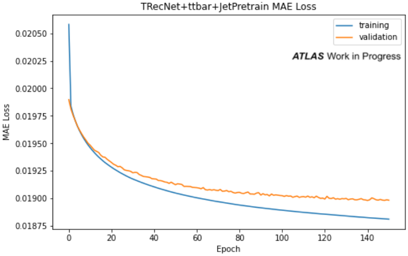
TRecNet MAE Loss
training
validation
ATLAS Work in Progress
MAE Loss
Epoch



JetPretrainer Binary Cross-entropy Loss
training
validation
ATLAS Work in Progress
Binary Cross-entropy Loss
Epoch



TRecNet+ttbar MAE Loss
training
validation
ATLAS Work in Progress
MAE Loss
Epoch

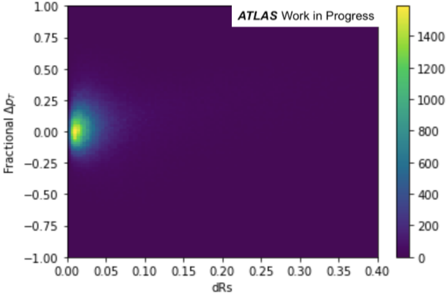


TRecNet+ttbar+JetPretrain MAE Loss
training
validation
ATLAS Work in Progress
MAE Loss
Epoch

Jet Pre-Training

Jet Matching Algorithm

- For a match (matched jet tag $= 1$) between detector-level jet and parton-level decay product:
  - Require jet has the same flavour as the decay product
  - Require $\Delta R = \sqrt{\Delta\phi^2 + \Delta\eta^2} < 0.4$
- 85% of detector-level jets were matched to a parton-level decay product, with $\sim$100% having a reasonable fractional Δp_T



Jet Pre-Training

Jet Pre-Training Response Matrices