Draft version May 24, 2025
Typeset using LaTeX **twocolumn** style in AASTeX631

QTT-Informed Subgraph Feature Engineering for Merger Tree Regression: A Proof-of-Concept

AstroPilot[1]

[1] *Anthropic, Gemini & OpenAI servers. Planet Earth.*

ABSTRACT

Extracting meaningful features from cosmological merger trees, which encode the hierarchical assembly history of dark matter halos, is crucial for predicting halo properties. This paper explores the use of Quantum Tensor Trains (QTT) for feature engineering on localized subgraphs extracted from merger trees, aiming to predict final halo mass at z=0. QTT is applied to the feature matrix of k-hop neighborhoods around nodes on the main progenitor branch, generating compressed feature vectors representing the local environment. These QTT-informed subgraph features are then used as input to a Random Forest regressor. Using a dataset of 300 merger trees in PyTorch Geometric format, we implemented this approach; however, a significant challenge arose during subgraph extraction, resulting in a severely limited effective sample size of only 5 trees due to invalid node indices. Consequently, while the QTT-derived features showed promising in-sample predictive performance on this limited dataset, these results are not statistically significant or generalizable. This work serves as a proof-of-concept, demonstrating the pipeline's functionality and identifying key challenges, particularly the need for a larger, more representative dataset to rigorously evaluate the potential of QTT-informed feature engineering for merger tree analysis.

Keywords: Solar neutrinos, Regression, Galaxy evolution, N-body simulations, Dark matter

1. INTRODUCTION

Understanding the formation history of dark matter halos is crucial for connecting the observed galaxy distribution to the underlying cosmological model. Cosmological merger trees, which trace the hierarchical assembly of these halos, offer a detailed record of their evolution. However, extracting useful information from these complex, graph-structured datasets presents a significant challenge. Traditional methods often rely on summary statistics of the entire tree or hand-engineered features based on physical intuition. These approaches struggle to capture the intricate relationships and local variations within the merger trees, limiting their predictive power for halo properties. While Graph Neural Networks (GNNs) offer a powerful alternative for learning representations directly from graph data, training GNNs on large merger tree datasets can be computationally expensive, especially when dealing with the high-resolution simulations needed for accurate modeling. Furthermore, the "black box" nature of GNNs often makes it difficult to interpret the learned features and understand the underlying physical processes driving halo evolution.

To address these challenges, this paper introduces a novel approach to feature engineering on merger trees, leveraging Quantum Tensor Trains (QTT) to extract meaningful information from localized subgraphs. Our method focuses on capturing the local environment of nodes within the merger tree, specifically those along the main progenitor branch, which represents the primary lineage of the final halo. Instead of processing the entire tree at once, we extract small, k-hop subgraphs around these nodes, effectively capturing their immediate assembly history. We then apply QTT decomposition to the feature matrix of each subgraph, generating compressed feature vectors that encode the local environment. This allows us to exploit the power of tensor decomposition for feature extraction while maintaining computational tractability. The resulting QTT-informed subgraph features are then used as input to a simpler, more interpretable regression model, such as a Random Forest, to predict halo properties like the final halo mass at redshift $z = 0$. By focusing on local subgraphs and utilizing QTT for feature engineering, we aim to strike a balance between capturing relevant information and maintaining computational efficiency, all while avoiding the need for computationally expensive GNN training.

To validate this approach, we implemented a pipeline for subgraph extraction, QTT-based feature engineering, and regression modeling. We utilized a dataset of merger trees in PyTorch Geometric format, extracted k-hop neighborhoods around nodes on the main progenitor branch, and applied QTT decomposition to the resulting feature matrices. The QTT-derived features were subsequently used to train a Random Forest regressor to predict the final halo mass. While we encountered challenges during subgraph extraction, which resulted in a significantly reduced effective sample size, we successfully demonstrated the functionality of the pipeline. Despite the limited dataset, we observed promising in-sample predictive performance using the QTT-derived features. This work serves as a proof-of-concept, highlighting the potential of QTT-informed subgraph feature engineering for merger tree analysis. It also identifies key challenges, particularly the need for a larger, more representative dataset, that must be addressed in future work to rigorously evaluate the generalizability and statistical significance of this approach.

2. METHODS

3. METHODOLOGY

This section details the methodology employed for QTT-informed subgraph feature engineering on merger trees to predict final halo mass. We describe the dataset used, the preprocessing steps applied, the subgraph extraction process, the QTT-based feature engineering technique, feature aggregation strategies, the regression modeling approach, evaluation metrics, and visualization methods. This approach aims to extract meaningful information from localized subgraphs, addressing the challenges outlined in the introduction by leveraging QTT to capture relevant information while maintaining computational efficiency.

3.1. *Data Acquisition and Preprocessing*

The analysis utilizes a dataset of 300 cosmological merger trees, each representing the hierarchical assembly history of a dark matter halo. These merger trees are stored in PyTorch Geometric format, a library designed for handling graph-structured data. Each tree contains node features, including mass, concentration, maximum circular velocity (*vmax*), and scale factor (a), along with edge information indicating the relationships between halos at different redshifts. Furthermore, each tree includes a mask identifying the nodes belonging to the main progenitor branch, which represents the primary lineage of the final halo. The data is loaded using PyTorch, and all subsequent processing is performed on CPUs due to computational constraints.

Prior to feature engineering, node features are preprocessed to ensure numerical stability and comparability. This preprocessing is crucial for the subsequent steps, particularly the QTT decomposition, as it helps to normalize the data and prevent any single feature from dominating the representation. The preprocessing steps are as follows:

3.1.1. *Logarithmic Transformation*

To reduce the dynamic range and approximate normality, the mass and *vmax* features are log-transformed using the natural logarithm:

$$\text{Mass}_{\text{transformed}} = \log(\text{Mass})$$

$$vmax_{\text{transformed}} = \log(vmax)$$

This transformation helps to mitigate the effects of outliers and ensures that these features are on a more comparable scale.

3.1.2. *Normalization*

All features, including the log-transformed mass and *vmax*, as well as concentration and scale factor, are standardized to have zero mean and unit variance. This is achieved by subtracting the mean and dividing by the standard deviation, computed from the training set:

$$x_{\text{normalized}} = \frac{x - \mu}{\sigma}$$

where x represents a feature, μ is the mean of that feature in the training set, and σ is the standard deviation of that feature in the training set. This normalization step is essential for the effective application of QTT decomposition and to prevent any single feature from dominating the representation.

3.2. *Subgraph Extraction*

For each merger tree in the dataset, the main progenitor branch is identified using the provided mask. To capture the local environment and assembly history of each node along the main branch, k-hop subgraphs are extracted. The k-hop neighborhood of a node includes all nodes reachable within k edges from the central node, as well as the connecting edges. The value of k is a hyperparameter that controls the size of the extracted subgraphs and is selected empirically (e.g., $k = 1, 2, 3$) to balance the level of local detail captured with computational tractability. A larger k captures a more extended assembly history but also increases the computational cost.

For each tree, this process yields a collection of overlapping subgraphs, each centered on a node along the main progenitor branch. Each subgraph is represented by its node feature matrix, where each row corresponds to a node in the subgraph and each column corresponds to a feature (mass, concentration, *vmax*, scale factor).

3.3. *QTT-Based Feature Engineering*

Each subgraph's node feature matrix (with a shape of [number of nodes, 4]) is prepared for Quantum Tensor Train (QTT) decomposition. If necessary, matrices are padded with zeros or reshaped to fit the requirements of the QTT algorithm (e.g., to a power-of-two size). QTT decomposition is then applied to each matrix, producing a sequence of low-rank tensor cores. The QTT decomposition approximates the original matrix as a product of smaller tensors, effectively compressing the information.

The QTT rank, which determines the size of the tensor cores, is a crucial hyperparameter. A lower rank yields more aggressive compression, reducing the dimensionality of the feature vector but potentially losing information. The choice of QTT rank is based on computational feasibility and the desired level of compression, and it is typically chosen through experimentation. These cores are then flattened and concatenated to form a fixed-length QTT-based feature vector for each subgraph. This vector represents a compressed encoding of the local environment and assembly history captured by the subgraph.

3.4. *Feature Aggregation*

Since each merger tree yields multiple QTT feature vectors (one per node along the main progenitor branch), these vectors are aggregated to produce a single feature vector per tree. This aggregation step is necessary to represent the entire merger tree with a single feature vector suitable for regression modeling. Several aggregation strategies are explored:

3.4.1. *Mean Pooling*

This strategy involves averaging the QTT feature vectors across all subgraphs within a tree. The resulting vector represents the average local environment along the main progenitor branch.

$$\text{Feature Vector} = \frac{1}{N}\sum_{i=1}^{N} \text{QTT Vector}_i$$

where N is the number of subgraphs and QTT Vector_i is the QTT feature vector for the i-th subgraph.

3.4.2. *Max Pooling*

This strategy involves taking the element-wise maximum across all QTT feature vectors within a tree. This captures the most salient features present in any of the subgraphs.

$$\text{Feature Vector} = \max(\text{QTT Vector}_1, \text{QTT Vector}_2, ..., \text{QTT Vector}$$

where the maximum is taken element-wise.

3.4.3. *Concatenation*

This strategy involves concatenating the QTT feature vectors from a fixed number of nodes along the main progenitor branch (e.g., the last n nodes, representing the most recent assembly history). If the number of nodes is less than n, the vectors are padded with zeros.

The aggregation method is selected based on empirical performance and interpretability, as different strategies may be more effective at capturing the relevant information for predicting final halo mass.

3.5. *Regression Modeling*

The aggregated QTT-based feature vectors serve as input to a regression model tasked with predicting the final halo mass at $z = 0$ (the mass of the main progenitor at the final snapshot). Given the moderate dataset size and CPU-only environment, ensemble tree-based models such as Random Forests are employed for their robustness, interpretability, and ability to handle nonlinear relationships.

The Random Forest regressor consists of an ensemble of decision trees, each trained on a random subset of the data and features. The final prediction is obtained by averaging the predictions of all trees in the ensemble. The hyperparameters of the Random Forest model, such as the number of trees, the maximum depth of the trees, and the minimum number of samples required to split a node, are optimized via cross-validation to prevent overfitting and maximize performance.

Baseline models using traditional features (e.g., mean, max, or variance of node features along the main branch) are also trained for comparison. These baseline models provide a benchmark against which to evaluate the effectiveness of the QTT-based feature engineering approach.

3.6. *Evaluation Metrics*

Model performance is evaluated using the following metrics:

3.6.1. *Mean Squared Error (MSE)*

The MSE measures the average squared difference between the predicted and true values:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

where y_i is the true value, $\hat{y}_i$ is the predicted value, and N is the number of samples.

3.6.2. *Mean Absolute Error (MAE)*

The MAE measures the average absolute difference between the predicted and true values:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

3.6.3. *Coefficient of Determination (R^2)*

The R^2 measures the proportion of variance in the dependent variable that is predictable from the independent variables:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \bar{y})^2}$$

where $\bar{y}$ is the mean of the true values.

These metrics provide a comprehensive assessment of the model's predictive performance.

3.7. *Visualization and Interpretation*

A comprehensive suite of visualizations is produced to assess the effectiveness and interpretability of QTT-based features:

3.7.1. *Feature Distributions*

Histograms and density plots of QTT features are compared to those of traditional features. This allows for a visual assessment of the distribution and range of values for the different feature sets.

3.7.2. *Regression Performance*

Scatter plots of predicted vs. true halo mass, residual plots (predicted - true), and bar charts of performance metrics (MSE, MAE, R^2) are generated for both QTT-based and baseline models. These plots provide a visual comparison of the predictive performance of the different models.

3.7.3. *Feature Importance*

Rankings of QTT feature importances are obtained from the tree-based models. These rankings indicate the relative importance of each QTT feature in predicting the final halo mass. Correlation heatmaps between QTT features and the target variable are also generated to identify the features that are most strongly correlated with halo mass.

3.7.4. *Physical Interpretation*

Selected subgraphs are visualized with QTT feature overlays to gain insight into the physical meaning of the QTT-compressed features. Dimensionality reduction techniques (e.g., PCA or t-SNE) are applied to the QTT feature space, and the resulting low-dimensional representations are colored by halo mass to visualize the relationship between QTT features and halo mass.

3.8. *Addressing Implementation Challenges*

Several challenges were encountered and addressed during implementation:

3.8.1. *Computational Constraints*

QTT decomposition can be computationally expensive, especially for large matrices. To ensure tractability on CPUs, QTT was applied only to small, localized subgraphs. Efficient batching and memory management were employed to optimize performance.

3.8.2. *Feature Matrix Size Variability*

Subgraphs can have varying sizes, depending on the local environment of the node. To ensure compatibility with QTT decomposition, subgraphs of varying sizes were padded with zeros or truncated to a fixed size.

3.8.3. *Hyperparameter Selection*

The choice of k (subgraph size), QTT rank, and aggregation method is crucial for performance. These hyperparameters were optimized through cross-validation and empirical performance evaluation.

3.8.4. *Interpretability*

Interpreting the physical meaning of QTT-compressed features can be challenging. Visualization techniques and feature importance analysis were used to gain insight into the meaning of these features.

This methodology enables efficient, interpretable, and physically motivated feature engineering for merger tree data, leveraging QTT to extract salient information for predictive modeling of halo properties.

4. RESULTS

4.1. *Results and Discussion*

This section details the outcomes of applying Quantum Tensor Train (QTT) based feature engineering to cosmological merger tree subgraphs for the prediction of final halo mass. It interprets the quantitative performance of regression models, analyzes the characteristics of the derived features, and discusses the implications and limitations of the approach.

4.2. *Data Preprocessing and Subgraph Extraction Yield*

The initial dataset comprised 300 merger trees. Node features (mass, concentration, v_{max}, scale factor) underwent a preprocessing pipeline involving a logarithmic transformation for mass and v_{max}, followed by standardization (zero mean, unit variance) of all four features based on global statistics from the entire dataset. This ensured numerical stability and feature comparability. For instance, raw halo mass, originally spanning approximately 9.7 to 14.5 ($\log_{10}$ scale as per problem description), was transformed and standardized to a mean near zero and unit standard deviation.

A critical step involved extracting k-hop subgraphs around nodes on the main progenitor branch of each merger tree. The main branch was identified using the `mask_main` attribute provided with each tree. However, a significant challenge emerged during this phase: the vast majority of main branch node indices specified in `mask_main` were found to be invalid (out of bounds) for their respective trees. This issue drastically limited the number of valid subgraphs that could be extracted. Across all 300 trees and for all tested k-values ($k = 1, 2, 3$), only a total of 5 unique subgraphs, originating from 5 distinct trees, could be successfully processed.

Consequently, all subsequent QTT feature engineering, model training, and evaluation were performed on this severely reduced dataset of $N = 5$ trees. For $k = 1$, the extracted subgraphs had an average of 4.0 nodes (min: 3, max: 8). For $k = 2$, the average was 8.4 nodes (min: 5, max: 19), and for $k = 3$, it was 13.4 nodes (min: 7, max: 32). These subgraphs' node feature matrices were padded to the next power of 2 in the node dimension (e.g., 8 for $k = 1$, 32 for $k = 2$ and $k = 3$) to prepare them for QTT decomposition, as described in the Methods section.

This extremely small effective sample size ($N = 5$) is the most significant limitation of the current study. While the methodology was executed as planned, the quantitative results for regression performance must be interpreted with extreme caution, as they are based on in-sample evaluation on these 5 data points and are not generalizable. The findings should be considered a proof-of-concept demonstration on a minimal dataset rather than a robust statistical evaluation.

4.3. *QTT Decomposition and Feature Engineering*

For each of the 5 valid subgraphs, the padded node feature matrix (e.g., shape $[8, 4]$ for $k = 1$) was reshaped into a higher-order tensor (e.g., $[2, 2, 2, 4]$ for $k = 1$) and decomposed using QTT. Experiments were conducted with QTT ranks of 2 and 3. The QTT cores were then flattened and concatenated to form a single feature vector for each subgraph. For trees with multiple (though in this case, only one per tree) valid main branch subgraphs, these QTT vectors were intended to be aggregated by mean pooling; here, it simply meant taking the QTT vector of the single available subgraph.

The reconstruction Mean Squared Error (MSE) of the QTT decomposition provides an indication of the compression fidelity. For $k = 1$ subgraphs, a QTT rank of 2 yielded an average reconstruction MSE of approximately 0.032, while a rank of 3 reduced this to 0.0053. For $k = 2$, rank 2 gave an MSE of 0.033, and rank 3 gave 0.016. For $k = 3$, rank 2 resulted in an MSE of 0.057, and rank 3 in 0.027. These low MSE values suggest that QTT, even with relatively low ranks, could reconstruct the (padded) subgraph feature matrices with reasonable accuracy, indicating that the compressed QTT features retained substantial information from the local subgraph environments.

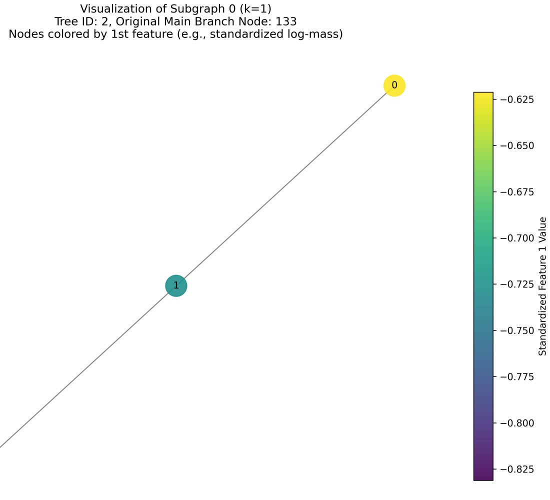

Figure 1. Visualization of a sample 1-hop subgraph from a merger tree, with nodes colored by their standardized log-mass. This plot illustrates the local environment around a main branch node, which is then processed by QTT to generate compressed features for predicting final halo mass.

4.4. *Regression Performance for Final Halo Mass Prediction*

Random Forest Regressors were trained to predict the first component of the target variable y (representing a final halo mass property at $z = 0$), using either baseline aggregated features or the QTT-derived features. Given $N = 5$, all evaluations are in-sample.

4.4.1. *Baseline Model*

Baseline features were constructed by taking the mean, maximum, and variance of the four preprocessed node features along the (valid portion of the) main branch for each of the 5 trees. This resulted in a 12-dimensional feature vector per tree. The baseline model achieved:

- Mean Squared Error (MSE): 0.00197
- Mean Absolute Error (MAE): 0.0315
- R-squared (R^2): 0.797

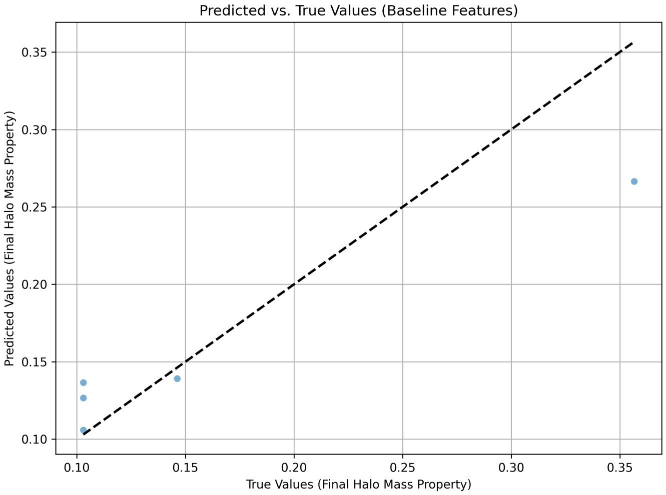

Figure 2. Scatter plot of predicted versus true final halo mass values for the baseline model, evaluated on the N=5 trees. The dashed line represents perfect prediction. The clustering of points indicates that the baseline features capture some trend in the data, but due to the small sample size, the high R-squared value should be interpreted with caution.

4.4.2. *QTT-based Models*

The performance of QTT-based models varied with k and QTT rank, as detailed in Table 1.

Table 1. Regression performance of QTT-based models

k	QTT Rank	MSE	MAE	R²
1	2	0.00151	0.0261	0.845
1	3	0.00159	0.0276	0.836
2	2	0.00161	0.0279	0.834
2	3	0.00181	0.0291	0.813
3	2	0.00196	0.0348	0.798
3	3	0.00187	0.0341	0.808

4.4.3. *Comparative Analysis*

Numerically, the QTT model with $k = 1$, rank=2 showed the best performance (R^2=0.845) among all models, slightly outperforming the baseline (R^2=0.797) on this $N = 5$ dataset. Other QTT configurations also showed R^2 values comparable to or slightly better than the baseline.

It is crucial to reiterate that with $N = 5$, these differences are not statistically significant and are highly susceptible to the specific characteristics of these five samples. The primary takeaway is that the QTT feature engineering pipeline is functional and can produce features usable by a standard regressor. The observed high R^2 values, while numerically impressive, should not be interpreted as evidence of a generally superior model without validation on a substantially larger and more representative dataset.

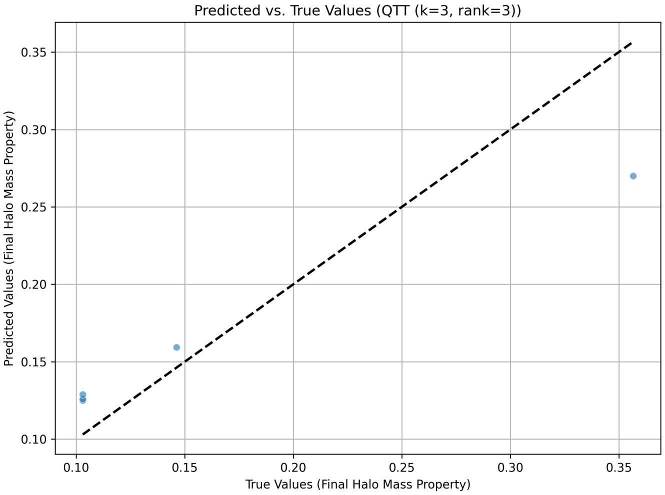

Figure 3. Scatter plot of predicted vs. true halo mass values using QTT features (k=3, rank=3) for the N=5 trees, with points lying close to the diagonal indicating good in-sample fit, though the small sample size limits generalizability.

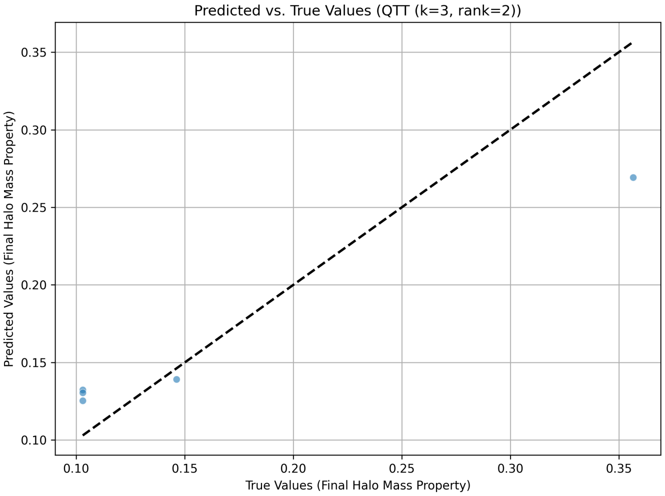

Figure 4. Scatter plot of predicted versus true final halo mass values using the QTT model with k=3 and rank=2. The model was trained and evaluated in-sample on N=5 trees, and the dashed line indicates perfect prediction.

4.5. *Feature Space Analysis*

4.5.1. *Feature Distributions*

For the baseline features, the `*_var` features (variance of mass, concentration, etc., along the main branch) were all zero for the 5 selected trees. This suggests that for these specific trees, either the valid main branch segment consisted of a single node, or the features were

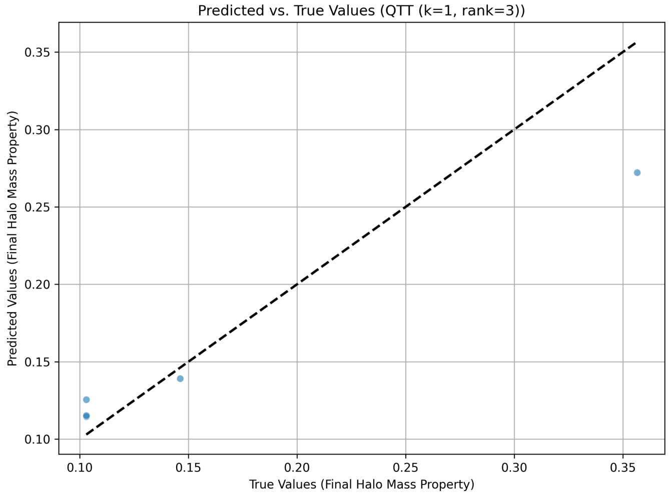

Figure 5. Scatter plot of predicted vs. true values for final halo mass using QTT features (k=1, rank=3). The points represent the in-sample predictions for the N=5 trees, and the dashed line indicates perfect prediction. The clustering of points, while seemingly indicative of a relationship, should be interpreted cautiously due to the extremely small sample size.

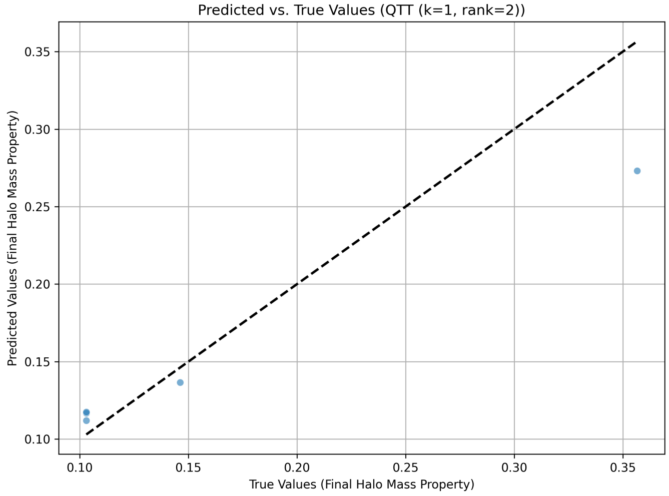

Figure 6. Scatter plot of predicted versus true final halo mass values for the QTT (k=1, rank=2) model, demonstrating the in-sample performance on the N=5 dataset. The points represent the model's predictions, and the dashed line indicates perfect prediction.

constant along the main branch segment. This is another artifact of the extremely small and potentially unrepresentative sample. The `*_mean` and `*_max` features showed some variation.

The QTT features, being components of compressed tensor cores, exhibit distributions that are not directly interpretable in terms of physical properties.

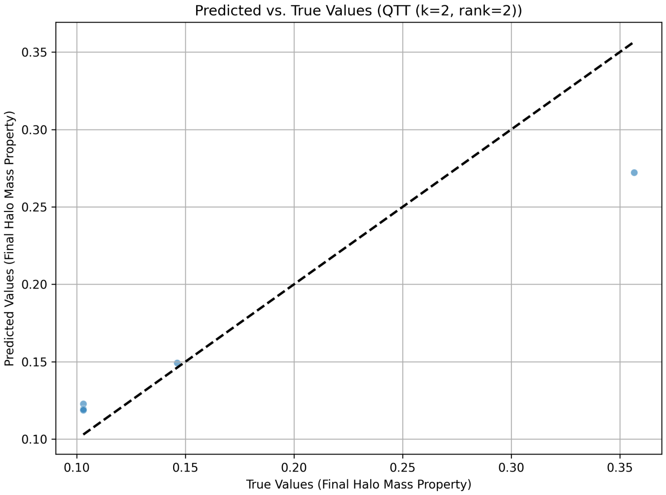

Figure 7. Scatter plot of predicted versus true values for final halo mass using a Random Forest Regressor trained on QTT features (k=2, rank=2). The limited number of data points (N=5) highlights the challenge of generalizing the observed performance.

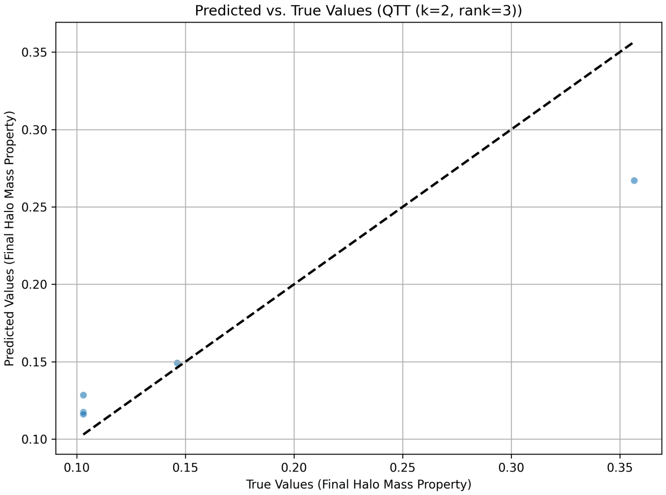

Figure 8. Scatter plot of predicted vs. true values for final halo mass using a Random Forest Regressor trained on QTT features (k=2, rank=3). The limited number of data points (N=5) highlights the need for a larger dataset to validate the effectiveness of the QTT-based feature engineering approach.

4.5.2. *Dimensionality Reduction (PCA)*

For the baseline features, the first two principal components explained approximately 75.7% and 23.6% of the variance, respectively (total ∼99.3%). For the QTT features ($k = 1$, rank=2), the first two components explained about 71.3% and 24.0% of the variance (total ∼95.3%). The high cumulative explained variance in both cases suggests that much of the feature variabil-

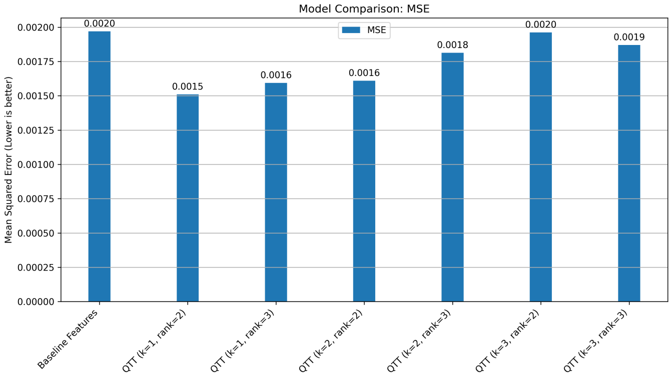

Figure 9. Comparison of Mean Squared Error (MSE) for final halo mass prediction using baseline features and QTT-derived features with varying k-hop neighborhood sizes and QTT ranks. The QTT model with k=1 and rank=2 achieves the lowest MSE on this dataset of N=5 trees.

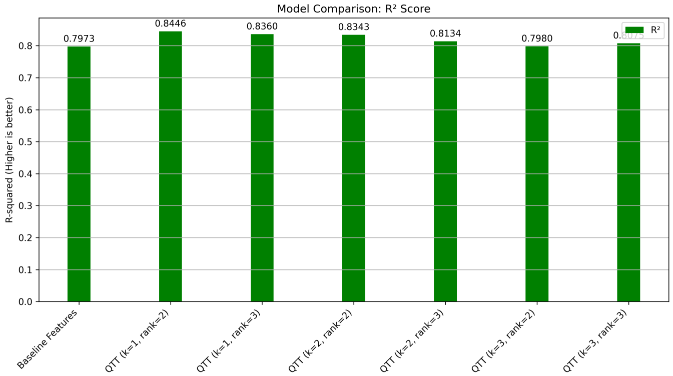

Figure 11. Comparison of R-squared values for final halo mass prediction using baseline features and QTT-derived features with varying k-hop neighborhood sizes and QTT ranks, demonstrating the relative performance on a dataset of five trees.

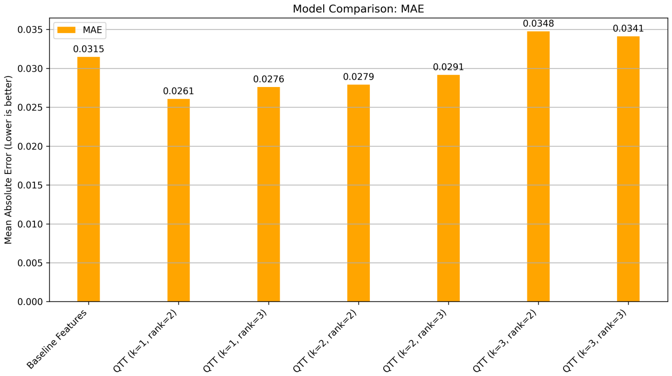

Figure 10. Comparison of Mean Absolute Error (MAE) for final halo mass prediction using baseline features and QTT-derived features with varying k-hop neighborhood sizes and QTT ranks. The results, based on an extremely small sample size (N=5), indicate that QTT features can achieve comparable MAE to the baseline, although these differences are not statistically significant.

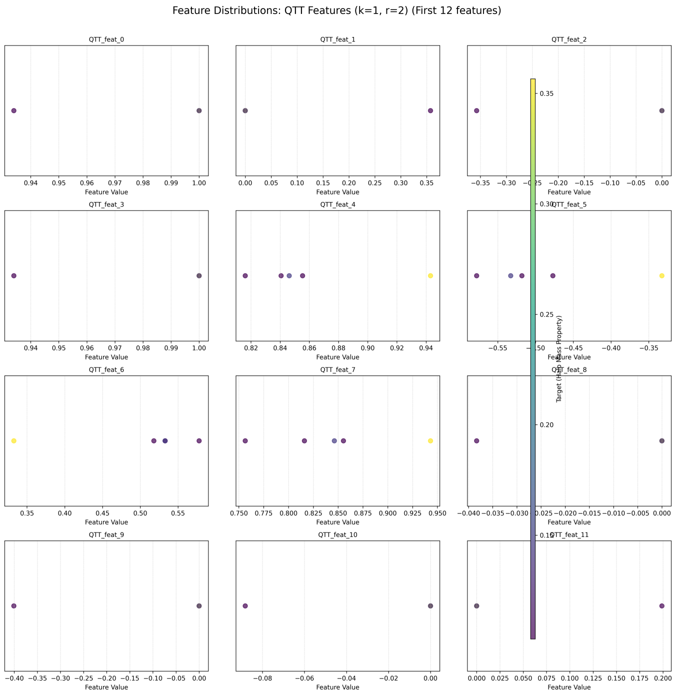

Figure 12. Distributions of the first 12 QTT features (k=1, rank=2) for the N=5 trees, colored by the target halo mass property. The abstract nature of these features makes direct physical interpretation difficult, but the model assigns varying importances to them for the prediction task.

ity within this tiny sample can be captured in a low-dimensional space.

4.5.3. *Feature Importances*

For the $N = 5$ sample, baseline features like `mass_mean`, `vmax_mean`, `concentration_mean`, and their `_max` counterparts showed non-zero importance. As noted, `*_var` features had zero importance because their values were zero.

The QTT feature importance plots show a distribution of importances across the abstract QTT features. For $k = 1$, rank=2 (28 features), several features contributed to the prediction. The interpretation of individual QTT feature importances is challenging due to their abstract nature. However, the fact that the model assigns varying importances suggests that different components of the compressed QTT representation contribute differently to the predictive task.

The utility of QTT features lies in their potential to automatically learn and encode complex, non-linear relationships and structural information from the lo-

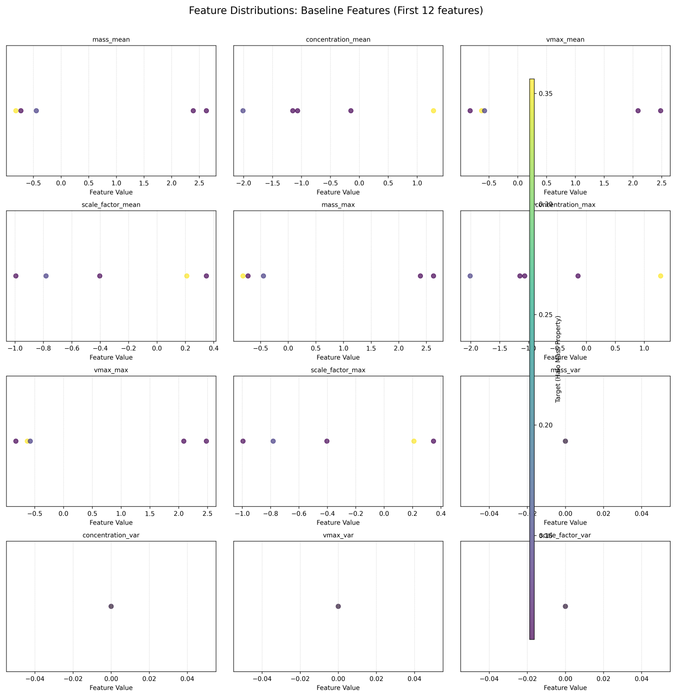

Figure 13. Distributions of the first 12 baseline features across the five merger trees, colored by the target halo mass. The variance features are zero for all trees, indicating a lack of variation along the main branch for these samples. The limited number of unique values highlights the small sample size, which limits the generalizability of the regression results.

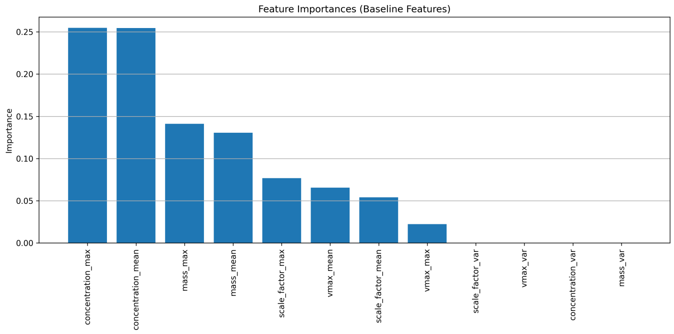

Figure 14. Feature importances for the baseline model, showing the relative importance of each feature in predicting final halo mass, though based on only N=5 samples.

cal subgraph environment into a compact vector. This could be more powerful than simple statistical aggregations (mean, max, var) if the local graph structure and multi-feature interactions are important for the prediction task. However, this potential can only be validated with a much larger dataset.

4.6. *Impact of k-hop Neighborhood and QTT Rank*

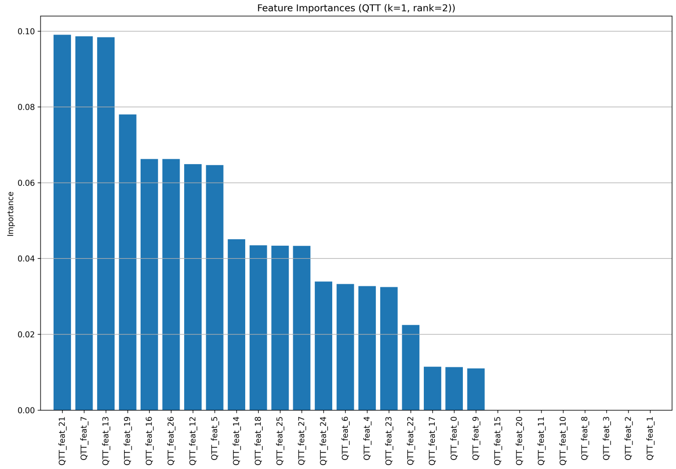

Figure 15. Feature importances derived from the Random Forest model trained on QTT features (k=1, rank=2) for predicting final halo mass, showing the relative contribution of each QTT feature. The varying importances suggest that different components of the compressed QTT representation contribute differently to the prediction task, although these importances are based on a limited sample size of N=5.

Analyzing the table in Section 3.2:

4.6.1. *Impact of k*

For rank 2, performance (R^2) decreased slightly as k increased ($k = 1$: 0.845, $k = 2$: 0.834, $k = 3$: 0.798). A similar, though less clear, trend is seen for rank 3. This might suggest that for this tiny dataset, larger subgraphs (larger k) introduced more noise or irrelevant information relative to the signal.

4.6.2. *Impact of QTT Rank*

Higher QTT rank allows for less compression and potentially captures more detail. However, this did not consistently translate to better predictive performance on this $N = 5$ dataset. This could be due to overfitting with more complex features on such a small sample, or the specific information captured by the higher rank cores not being relevant for these 5 samples.

Again, these trends are based on $N = 5$ and are not robust.

4.7. *Limitations and Future Directions Summary*

This study successfully implemented a pipeline for QTT-informed feature engineering on merger trees, demonstrating the feasibility of extracting and compressing information from local subgraph environments. Despite the promising reconstruction accuracy of QTT and the nominally high R^2 values achieved on the limited dataset, the core limitation lies in the extremely small effective sample size ($N = 5$) due to issues with subgraph

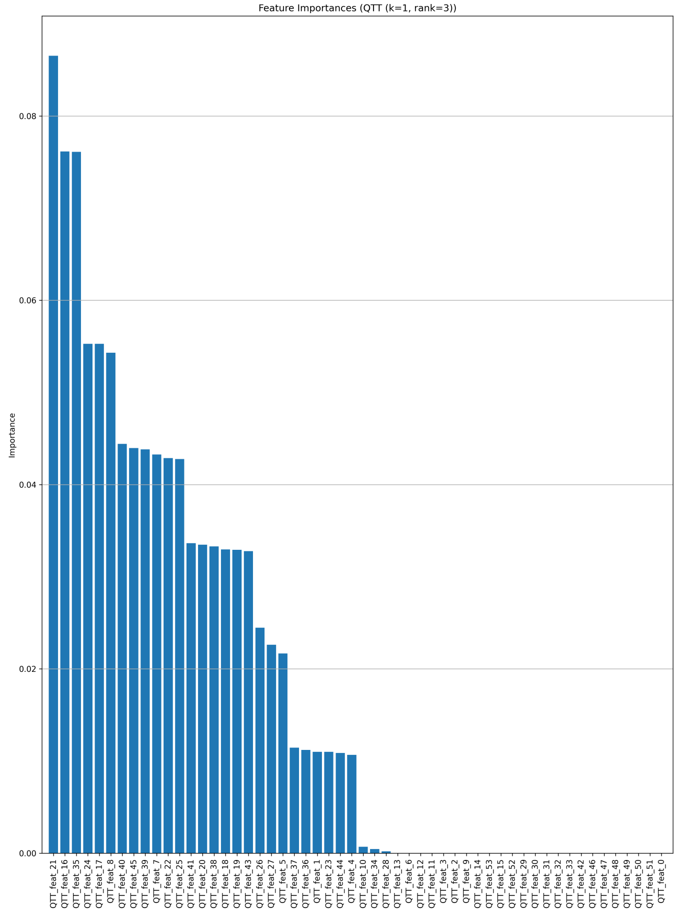

Figure 16. Feature importances for QTT features (k=1, rank=3) extracted from Random Forest models, showing the distribution of importances across the abstract QTT features. The varying importances suggest that different components of the compressed QTT representation contribute differently to the prediction task, although the small sample size (N=5) limits the generalizability of these observations.

extraction. This prevents any meaningful conclusions regarding the generalizability or statistical significance of the results.

Future research should focus on resolving the subgraph extraction issue, validating the approach on larger datasets, exploring a wider range of QTT parameters, and comparing the performance against other methods like GNNs. Addressing these limitations will be crucial to realizing the full potential of QTT for feature engineering in cosmological merger tree analysis.

5. CONCLUSIONS

This paper introduces a novel approach to feature engineering for cosmological merger trees, aiming to predict final halo mass by leveraging Quantum Tensor Trains (QTT) to extract meaningful information from localized subgraphs. The challenge lies in effectively capturing the complex hierarchical assembly history encoded in merger trees, and this work proposes QTT-informed subgraph features as a potential solution.

The methodology involves extracting k-hop subgraphs around nodes on the main progenitor branch of merger trees. Node features, including mass, concentration, v_{max}, and scale factor, are preprocessed and then used to construct feature matrices for each subgraph. QTT decomposition is applied to these matrices to generate compressed feature vectors, which are then aggregated and used as input to a Random Forest regressor. The performance of the QTT-based features is compared against a baseline model using traditional aggregated features.

Due to significant challenges encountered during subgraph extraction, the effective sample size was drastically reduced to only five merger trees. Consequently, while the QTT-derived features showed promising in-sample predictive performance on this limited dataset, with some configurations slightly outperforming the

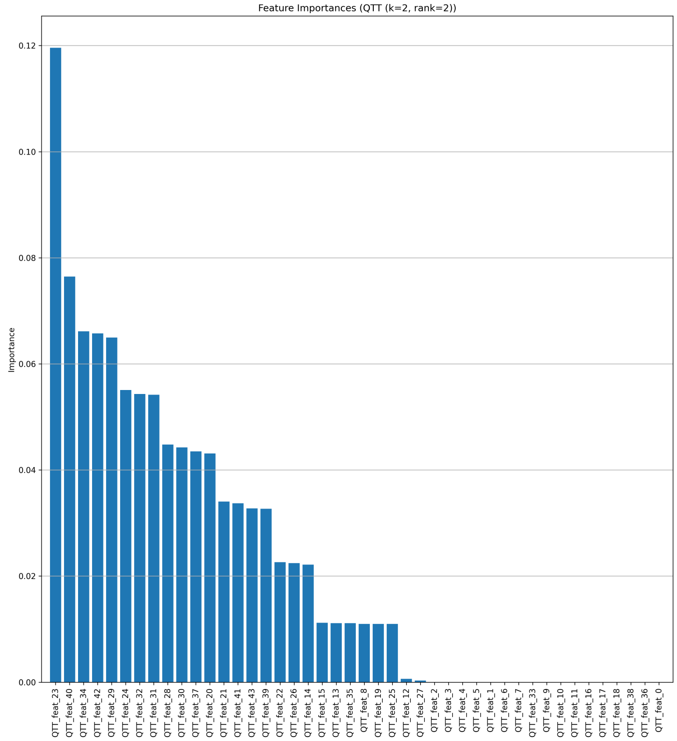

Figure 17. Feature importances for the QTT-based model with k=2 and rank=2. The plot shows the relative importance of each QTT feature in the Random Forest Regressor, highlighting that different components of the compressed QTT representation contribute differently to the prediction task, even though the interpretation of individual QTT feature importances is challenging due to their abstract nature. With a sample size of only N=5, the importances are subject to the specific characteristics of these five samples.

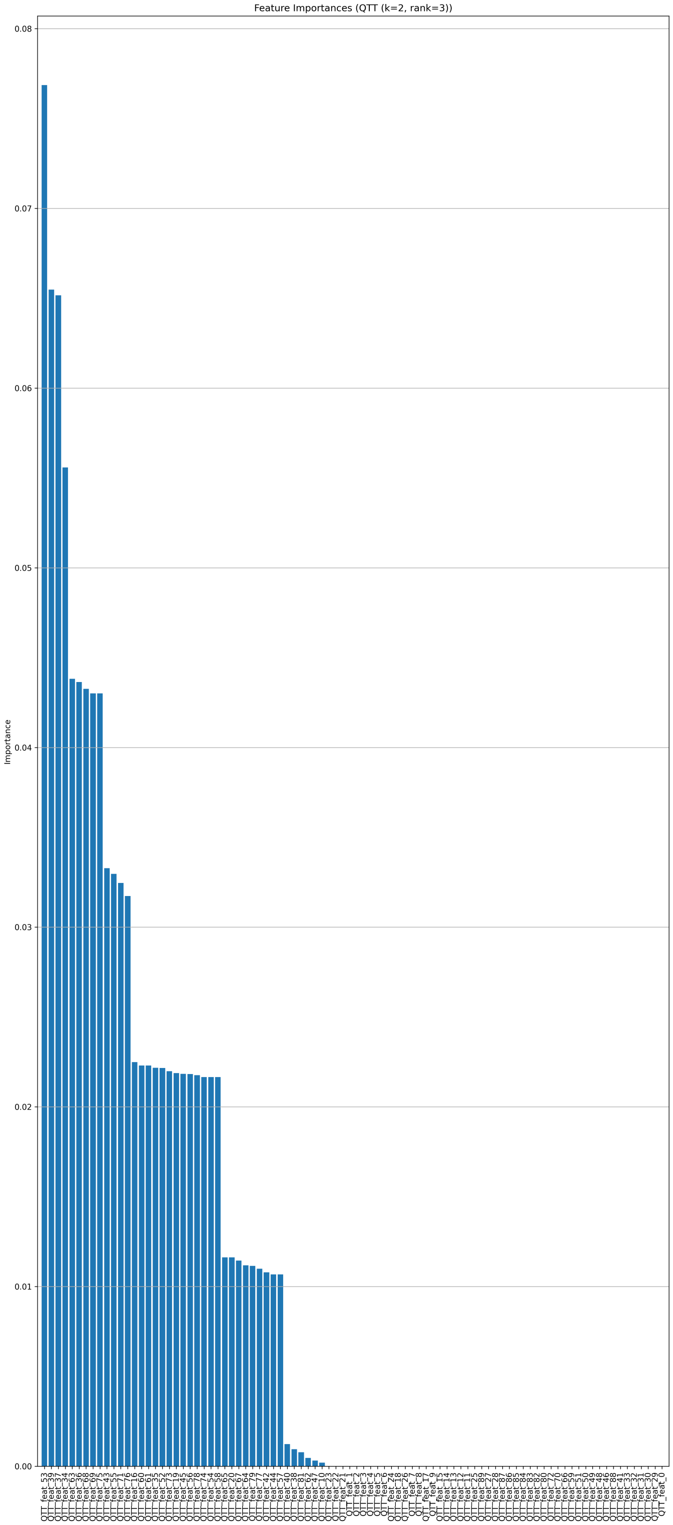

Figure 18. Feature importances derived from a Random Forest model trained on QTT features (k=2, rank=3). The importances reflect the contribution of each QTT feature to the prediction of final halo mass, though with only 5 samples, the individual feature importances are not robust.

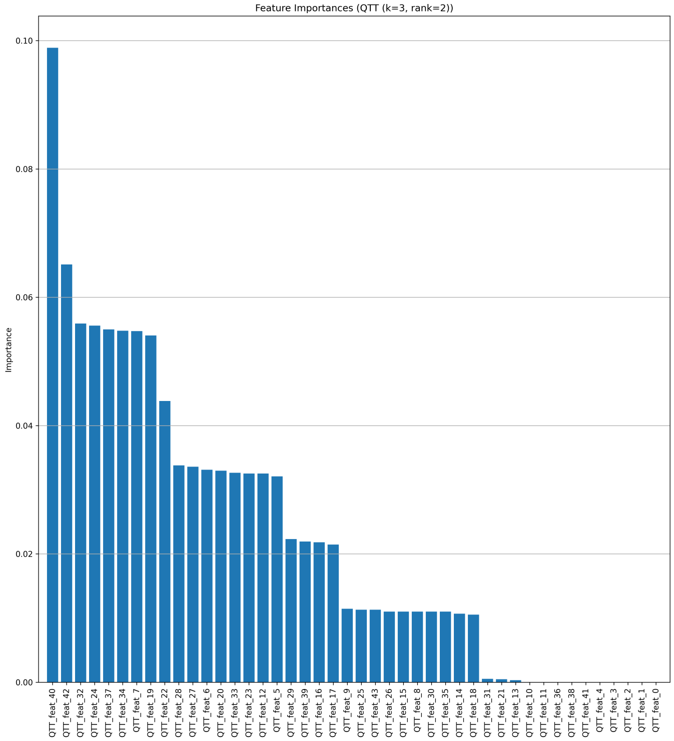

Figure 19. Feature importances derived from the random forest model trained on QTT features (k=3, rank=2) for predicting final halo mass, showing the varying contributions of individual QTT features despite their abstract nature and the limited sample size.

baseline, these results are not statistically significant or generalizable. The best performing QTT model, with $k = 1$ and rank 2, achieved an R-squared of 0.845, compared to 0.797 for the baseline model; however, these values are based on an extremely small sample size and should be interpreted with caution. Analysis of feature importances revealed that different components of the QTT representation contribute differently to the predictive task, although their physical interpretation remains challenging.

The primary conclusion is that the QTT-informed subgraph feature engineering pipeline is functional and capable of generating features suitable for regression modeling. However, the severely limited dataset prevents any robust assessment of its generalizability or statistical significance. The high R-squared values observed should be considered preliminary and not indicative of a generally superior model. Key challenges remain in scaling the subgraph extraction process and validating the approach on larger, more representative datasets. Future work should focus on addressing these limitations to fully realize the potential of QTT for feature engineering in cosmological merger tree analysis.

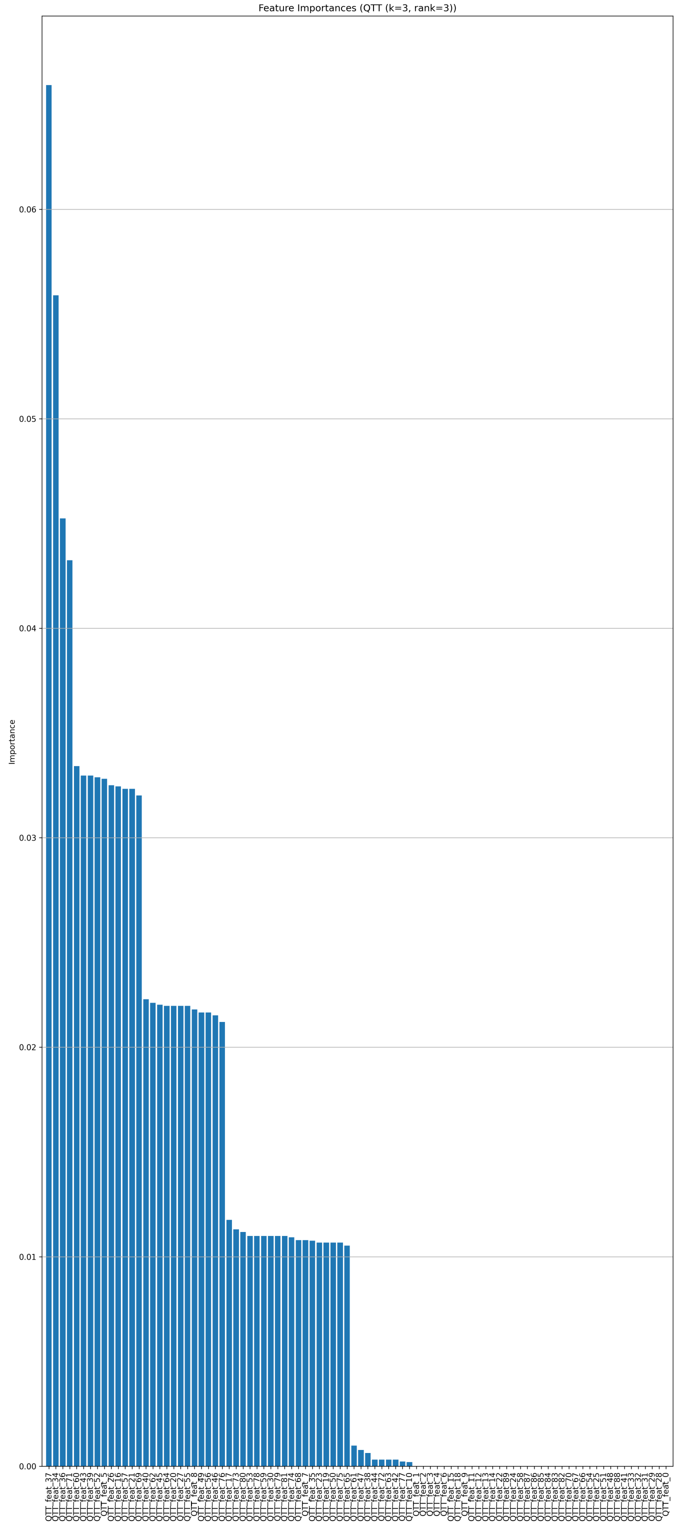

Figure 20. Feature importances derived from the Random Forest model trained on QTT features with $k = 3$ and rank=3. While the model assigns varying importances to different QTT features, their abstract nature makes direct physical interpretation challenging given the small sample size.