ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

GraphTreeBoost: Soft Decision Tree-Based Graph Learning With Spectral Aggregation

László Fetter and András Gézsi

Budapest University of Technology and Economics
Department of Artificial Intelligence and Systems Engineering
1111 Budapest - Hungary

Abstract. We propose GraphTreeBoost, a gradient-boosted framework for graph-structured data that couples soft decision trees with spectral feature aggregation. Each split operates on features filtered by truncated Chebyshev or Chebyshev–Bessel (heat kernel) expansions of the normalized adjacency, enabling efficient graph-aware learning without eigendecomposition. Training combines analytic second-order leaf updates with AdamW for routing and filter parameters, yielding stable optimization and interpretable thresholds with per-node gain scores. All spectral operations are implemented in the feature space using sparse graph primitives, scaling linearly with the number of edges and remaining practical on CPU hardware. Experiments on six benchmarks show consistent accuracy gains over feature-only baselines, demonstrating that GraphTreeBoost unites the transparency of decision trees with scalable spectral graph learning.

1 Introduction

Graph machine learning spans many applications, motivating interpretable and efficient models. These include biological interaction networks, knowledge graphs, social and information networks, engineered systems, and financial transaction networks.

Graph convolutional networks (GCNs) [1] and other graph neural networks (GNNs) [2, 3] dominate this domain, but they face known issues: over-smoothing in deeper layers, weak performance on heterophilous graphs, and poor interpretability. Early diffusion- and spectral-based GNNs further rely on costly eigendecomposition or dense multiplications [4, 5], while later methods mitigated this but still introduced computational overhead.

Decision-tree ensembles such as Random Forests and Gradient Boosting [6, 7] remain attractive alternatives. They are interpretable, handle nonlinear dependencies without feature engineering, and scale efficiently. Their strong tabular-data performance motivates extending these advantages to graphs [8, 9, 10, 11].

We introduce **GraphTreeBoost**, a gradient-boosted framework where soft decision trees operate on spectrally aggregated features. Filters parameterized via a Chebyshev–Bessel heat kernel expansion enable efficient learning in feature space, avoiding eigen-decomposition entirely. Training uses analytic second-order leaf updates with AdamW for filter and routing parameters. The approach scales linearly with edge count and retains interpretability through calibrated thresholds and per-node gains, while extending naturally to vertex- and edge-level prediction via normalized 1-Hodge (edge) operators.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

2 Related work on tree-based methods for graph learning

Gradient boosting [6, 7] and decision tree ensemble methods provide interpretable, efficient learning and have recently been extended to graphs. Ivanov et al. [8] combined gradient boosted decision trees with GNNs, using trees for heterogeneous features and convolutions for structural learning, while Deng et al. [9] used GNN-derived molecular embeddings with XGBoost for molecular property prediction. Müller et al. [10] introduced DT+GNN, merging decision trees with graph message passing for explainable predictions, and Bechler-Speicher et al. [11] developed TREE-G, showing decision trees can rival GNNs on graph tasks.

On the spectral side, Chebyshev polynomial approximations [2] enabled efficient localized graph filters without eigendecomposition, later extended [4] with heat kernel formulations [3]. GraphTreeBoost unifies these ideas, combining gradient-boosted soft decision trees with Chebyshev-based spectral aggregation, offering interpretable, scalable graph learning without dense spectral operations.

3 GraphTreeBoost: Soft decision tree-based graph learning

GraphTreeBoost is a gradient-boosted ensemble of soft decision trees whose split functions are driven by graph-aware spectral aggregation of input features [3, 2]. Each internal node computes a differentiable routing probability from an aggregated version of a single input feature, and each leaf stores an analytically updated score [6, 7]. This way, GraphTreeBoost captures robust spectral signals and is resistant to mid-to-low frequency spectral bias. The spectral aggregation is implemented in the vertex (or edge) domain via a truncated Chebyshev polynomial of the normalized adjacency, avoiding eigendecomposition while retaining an interpretable frequency response [3, 4].

3.1 Mathematical formulation

We consider a graph with n nodes and e edges, each node described by k standardized input features. Let $X = [\mathbf{x}_1, \ldots, \mathbf{x}_k] \in \mathbb{R}^{n\times k}$ denote the feature matrix, and $\widehat{A} = D^{-1/2}AD^{-1/2}$ the normalized adjacency.

A degree-d Chebyshev filter with coefficients $\{b_m\}_{m=0}^{d}$ acts on feature $\mathbf{x}_j$ as

$$\phi(\mathbf{x}_j) = \sum_{m=0}^{d} b_m \, T_m(\widehat{A})\, \mathbf{x}_j.$$

$$T_0(\widehat{A}) = I, \qquad T_1(\widehat{A}) = \widehat{A}, \qquad T_{m+1}(\widehat{A}) = 2\widehat{A}T_m(\widehat{A}) - T_{m-1}(\widehat{A}).$$

For $d = 3$ the filter admits the closed form

$$\phi(\mathbf{x}_j) = \big(4b_3\widehat{A}^3 + 2b_2\widehat{A}^2 + (b_1 - 3b_3)\widehat{A} + (b_0 - b_2)I\big)\mathbf{x}_j.$$

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

At a node with incoming responsibilities $\mathbf{w}_{\text{node}} \in \mathbb{R}^n_{\geq 0}$ the (right) routing weights are

$$\mathbf{w}_{\text{right}} = \mathbf{w}_{\text{node}} \odot \sigma\big(\tau \cdot \big(\phi(\mathbf{x}_j) - \theta\big)\big)\,,$$

where σ is the sigmoid function, τ and θ are learned parameters, and $\odot$ denotes elementwise multiplication. The left routing weights are

$$\mathbf{w}_{\text{left}} = \mathbf{w}_{\text{node}} - \mathbf{w}_{\text{right}}.$$

Let f_l denote the leaf score of leaf ℓ and let ν be the learning rate. In multiclass settings, f_l becomes a vector with one component per class. A tree with leaves $\{l = 1, \ldots, T\}$ produces the additive update

$$\boldsymbol{\eta}^{(t)} = \boldsymbol{\eta}^{(t-1)} + \nu \sum_{l=1}^{T} f_l \, \mathbf{w}_l$$

to the linear predictor. Let ψ denote the inverse link function (e.g., sigmoid or softmax) and $\mathcal{D}$ the decision function mapping responses to final predictions. The predictions after t boosting rounds are

$$\hat{\mathbf{y}}^{(t)} = \mathcal{D}\big(\psi(\boldsymbol{\eta}^{(t)})\big),$$

Leaf values are regularized as

$$\Omega(f_l) = \tfrac{1}{2}\lambda_{\text{reg}} \|f_l\|_2^2.$$

The per-tree objective minimizes the mean loss plus regularization and split penalty:

$$\min_{\{b_m\},\,\text{tree}} \sum_{l=1}^{T} \Big(\mathbf{w}_l^\top \ell\big(\mathbf{y}, \psi(\boldsymbol{\eta}^{(t-1)} + f_l)\big) + \Omega(f_l)\Big) + \gamma T.$$

We either learn $\{b_m\}$ directly under a simplex constraint $0 \leq b_m \leq 1$ and $\sum_{m=0}^{d} b_m = 1$ enforced via softmax reparameterization, or parameterize them via the Chebyshev–Bessel expansion of the heat kernel using a single scalar $\beta > 0$:

$$b_0 = e^{-\beta} I_0(\beta),$$
$$b_m = 2e^{-\beta} I_m(\beta) \quad (m \geq 1)$$

where $\{I_m\}$ denote the modified Bessel functions of the first kind. These coefficients arise from the Chebyshev–Bessel expansion of the heat kernel on the normalized Laplacian $\mathcal{L} = I - \widehat{A}$, which satisfies the identity [3]:

$$e^{-t\mathcal{L}} = e^{-t}\left(I_0(t)\, T_0(\widehat{A}) + 2\sum_{m=1}^{\infty} I_m(t)\, T_m(\widehat{A})\right).$$

Extension to edge learning. We note that the formulation can be extended to edge-level learning by replacing the normalized adjacency $\widehat{A}$ with its edge-space analogue $\widehat{A}_1 = I - \mathcal{L}_1$, where $\mathcal{L}_1 = \widehat{B}^\top \widehat{B}$ is the normalized edge Laplacian constructed using the normalized incidence matrix $\widehat{B}$, suitable for simple graphs without higher order cells. For edge features $F = [\mathbf{f}_1, \ldots, \mathbf{f}_k] \in \mathbb{R}^{e \times k}$, spectral aggregation is performed as $\phi(\mathbf{f}_j) = \sum_{m=0}^{d} b_m \, T_m(\widehat{A}_1)\, \mathbf{f}_j$.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

3.2 Optimization procedure

Outer loop (per-tree). The filter parameters $\{b_m\}$ (or β) are updated by AdamW on the mean training loss. The simplex is enforced by softmax reparameterization of an unconstrained vector, while β and the gate steepness τ are enforced positive via softplus with a small numerical offset.
Inner loop (per-node). For a candidate split (j, θ, τ) we use the second-order scheme of XGBoost [7], adapted to soft decision trees with per-example gradient $\mathbf{g}^{(t-1)}$ and Hessian diagonal $\mathbf{h}^{(t-1)}$. The node, child and gain scores are

$$\ell_{\text{node}} = -\frac{1}{2}\,\frac{\left(\mathbf{w}_{\text{node}}^{\top}\mathbf{g}^{(t-1)}\right)^2}{\mathbf{w}_{\text{node}}^{\top}\mathbf{h}^{(t-1)} + \lambda_{\text{reg}} + 10^{-8}},$$

$$\ell_{\text{split}} = \ell_{\text{left}} + \ell_{\text{right}},$$

$$G_{\text{node}} = \max_{j}\Big(\ell_{\text{node}} - [\ell_{\text{split}}]_j\Big),$$

and the split is committed if $G_{\text{node}} > \gamma$. Leaf scores admit a closed form:

$$f_{\text{leaf}} = -\frac{\mathbf{w}_{\text{leaf}}^{\top}\mathbf{g}^{(t-1)}}{\mathbf{w}_{\text{leaf}}^{\top}\mathbf{h}^{(t-1)} + \lambda_{\text{reg}} + 10^{-8}}.$$

Within a node, (θ, τ) are refined by AdamW on the sum of second-order losses of its children, while j is chosen by maximizing G_{node} over features.

3.3 Efficient computation

All spectral operations are realized in feature space with sparse graph primitives, avoiding eigendecomposition.
Recursive evaluation. We cache successive propagations of the normalized adjacency to form

$$\mathbf{z}_0 = \mathbf{x}, \quad \mathbf{z}_{m+1} = \widehat{A}\mathbf{z}_m, \qquad \phi(\mathbf{x}) = \sum_{m=0}^{d} c_m\,\mathbf{z}_m,$$

where $\{c_m\}$ are coefficients derived from $\{b_m\}$. The cached $\{\mathbf{z}_m\}$ are reused across nodes and boosting rounds to avoid redundant sparse multiplications.
Complexity. With e edges and sparse storage, one filtering pass costs $\mathcal{O}(de)$ per feature. A tree with T leaves has $2T-1$ nodes; over R boosting rounds and k features the overall cost is $\mathcal{O}(RTkde)$ for single-output tasks, or $\mathcal{O}(CRTkde)$ for C classes. Memory is $\mathcal{O}(dkn)$ for cached aggregated features.
Numerical and implementation details. We standardize features (per-feature z-score) using statistics computed only from the training vertices and those reachable within d hops, where validation and test vertices (and any edges incident to them) are explicitly masked out to avoid information leakage. The resulting (μ_j, σ_j) are retained for invertible threshold reporting. We use preallocated buffers for $\{\mathbf{z}_m\}$ and split evaluation, vectorized feature scans, AdamW with small stability constants, centered class-prior logit initialization for multiclass objectives, softplus for (β, τ), and softmax (or one-parameter mapping via β) for $\{b_m\}$ to maintain constraints and stability.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

4 Experimental evaluation

Setup. All experiments were conducted on a CPU-only workstation using six benchmark datasets (Cora, Citeseer, Texas, Cornell, Actor, Chameleon). We compared three model variants: (i) *No aggregation* (feature-only), (ii) *GTB-Cheby*, and (iii) *GTB-Heat*. We followed the original splits for all datasets. The Chebyshev truncation degree was fixed at $d = 7$. The GCN results in Table 1 are taken directly from the evaluations reported in [12].

Model structure. In *GTB-Cheby*, Chebyshev coefficients $\{b_m\}_{m=0}^{7}$ were learned with a simplex constraint enforced via softmax. In *GTB-Heat*, coefficients were generated from a single parameter β using the Chebyshev–Bessel expansion. All spectral operations were computed efficiently in the feature domain using cached Chebyshev feature propagations computed through sparse matrix-vector recurrences of $\widehat{A}$.

Training protocol. All configurations were trained ten times with different random initializations. Each tree was limited to a maximum of 10 leaf nodes. The maximum number of boosting rounds was set to 100, with early stopping based on validation performance over 10 consecutive rounds. The inner (per-node) optimization used AdamW with a learning rate of 0.1, weight decay of 0.1, and 50 epochs, whereas the outer (per-tree) optimization used AdamW with a learning rate of 0.05, no weight decay, and 20 epochs. Early stopping was applied to both optimizers with a patience of five rounds each. Leaf values were updated analytically using a second-order formula, and the objective was multi-class softmax cross-entropy.

Model	Cora	Citeseer	Texas	Cornell	Actor	Chameleon
GCN [1, 12]	$87.12_{\pm 1.38}$	$76.50_{\pm 1.61}$	$63.81_{\pm 5.27}$	$59.35_{\pm 4.19}$	$30.31_{\pm 0.98}$	$67.96_{\pm 1.82}$
No aggr.	$52.79_{\pm 0.96}$	$53.56_{\pm 1.59}$	$71.08_{\pm 2.56}$	$57.30_{\pm 3.78}$	$33.53_{\pm 1.06}$	$38.55_{\pm 2.13}$
GTB-Cheby	$74.09_{\pm 2.73}$	$56.77_{\pm 1.22}$	$72.43_{\pm 2.79}$	$66.75_{\pm 4.42}$	$31.05_{\pm 1.10}$	$71.75_{\pm 2.13}$
GTB-Heat	$71.27_{\pm 2.46}$	$57.18_{\pm 1.03}$	$73.24_{\pm 3.24}$	$72.97_{\pm 8.55}$	$33.34_{\pm 0.47}$	$50.83_{\pm 1.87}$

Table 1: Test accuracy (%) of GraphTreeBoost variants across benchmark datasets, reported as mean ± sample standard deviation over ten runs.

Results. Spectral aggregation improved accuracy on most datasets compared to the feature-only baseline, yielding observable gains on Cora, Citeseer, Texas, Cornell, and Chameleon. *GTB-Cheby* offered strong and stable improvements and achieved the best performance within our framework on Chameleon, while also narrowing the gap to GCN on the homophilous datasets. *GTB-Heat* complemented this by delivering the strongest results on Texas and competitive accuracy on Cornell, and additionally surpassing GCN on Texas, Cornell, and Actor. Although Actor remained challenging for both spectral variants relative to the feature-only model, they consistently strengthened performance on the remaining benchmarks. Overall, GraphTreeBoost with spectral filtering provides a robust and interpretable alternative with competitive accuracy across datasets.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

5 Conclusion

We introduced GraphTreeBoost, a gradient-boosted ensemble of soft decision trees enhanced with spectral aggregation at a truncated polynomial degree. With a stable training scheme using nested AdamW updates and analytic leaf formulas, the method yields accurate and interpretable graph models. Both Chebyshev and heat-kernel variants improved accuracy across datasets. The framework preserves decision-tree transparency, exposes feature thresholds and per-node gains, and scales efficiently with sparse graphs. Overall, GraphTreeBoost offers a practical and interpretable alternative to deep message-passing networks.

References

[1] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[2] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 3844–3852, Red Hook, NY, USA, 2016. Curran Associates Inc.

[3] David K. Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

[4] Bingbing Xu, Huawei Shen, Qi Cao, Keting Cen, and Xueqi Cheng. Graph convolutional networks using heat kernel for semi-supervised learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, pages 1928–1934. AAAI Press, 2019.

[5] Heng Chang, Yu Rong, Tingyang Xu, Wenbing Huang, Somayeh Sojoudi, Junzhou Huang, and Wenwu Zhu. Spectral graph attention network with fast eigen-approximation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, pages 2905–2909, New York, NY, USA, 2021. Association for Computing Machinery.

[6] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189 – 1232, 2001.

[7] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. Association for Computing Machinery.

[8] Sergei Ivanov and Liudmila Prokhorenkova. Boost then convolve: Gradient boosting meets graph neural networks, 2021.

[9] Daiguo Deng, Xiaowei Chen, Ruochi Zhang, Zengrong Lei, Xiaojian Wang, and Fengfeng Zhou. Xgraphboost: Extracting graph neural network-based features for a better prediction of molecular properties. *Journal of Chemical Information and Modeling*, 61(6):2697–2705, 2021. PMID: 34009965.

[10] Peter Müller, Lukas Faber, Karolis Martinkus, and Roger Wattenhofer. Dt+gnn: A fully explainable graph neural network using decision trees, 2022.

[11] Maya Bechler-Speicher, Amir Globerson, and Ran Gilad-Bachrach. Tree-g: Decision trees contesting graph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(10):11032–11042, Mar. 2024.

[12] Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks?, 2023.