PAVE: Parallel Adaptive View Excitation for Scalable Spectral Learning on Tabular Data

Matthew Fried, Mohammad Alshibli
SUNY Farmingdale, USA

***Abstract*—We introduce Parallel Adaptive View Excitation (PAVE), a framework for parallel selection and training across sample-view and feature-view graph Laplacians. Each view independently decomposes the data manifold, exposing complementary spectral structure for downstream learning. Unlike unified-graph pipelines that merge all relations into a single Laplacian with inherent serial bottlenecks, PAVE exposes coarse-grained task parallelism by evaluating sample-view and feature-view manifolds concurrently and selecting the view with higher validation accuracy, while retaining the alternate view for interpretability. Across 16 benchmark datasets, PAVE matches or exceeds strong gradient-boosted baselines on most tasks, with performance degradations typically small when present, while maintaining comparable or improved F1 and AUC. On modern multi-core systems, PAVE achieves 1.5–1.9$\times$ training speedup via view-level parallelism, with approximately 60% strong-scaling efficiency up to 32 workers and near-ideal efficiency under weak scaling relative to a serial execution. These results show that dual-view spectral excitation provides a lightweight, interpretable, and scalable alternative to conventional feature selection and manifold learning.**

***Index Terms*—parallel computation, spectral analysis, manifold decomposition**

I. Introduction

Traditional machine learning pipelines and ensemble models typically rely on sequential evaluation, either training on a single data view or combining multiple representations through fixed fusion strategies. These approaches implicitly assume that one representation dominates or that heterogeneous views can be meaningfully merged into a single model. Spectral methods provide a principled mechanism for uncovering the manifold structure of data [1], but their effectiveness depends critically on whether the sample-level and feature-level manifolds are compatible. In practice, unified spectral constructions often introduce computational bottlenecks and obscure whether multiple views are complementary or redundant. To address these limitations, we introduce *Parallel Adaptive View Excitation* (PAVE), a general framework that decomposes learning across independent sample-view and feature-view manifolds. Within PAVE, Laplacian-based learners are trained concurrently under lightweight coordination, allowing the system to explore multiple manifold structures without serial overhead. Rather than forcing early fusion, PAVE performs adaptive view selection based on validation performance, retaining alternative views for interpretability and diagnostic insight.

PAVE represents data separately in the sample and feature manifolds, constructing spectral embeddings that are applied to downstream learners. To control computational cost, PAVE subsamples up to 2,000 samples or features to form a representative Laplacian, and interpolates embeddings for remaining points using fast k-nearest-neighbor projection. This reduces graph construction from naive $O(n^2)$ complexity to $O(n)$ while preserving essential manifold geometry. We further employ the Nyström approximation, a standard technique in scalable spectral clustering and kernel learning, to improve efficiency.

Crucially, the computational cost of graph construction is offset by PAVE's distributed architecture, an effective embarrassingly parallel model. Sample-view and feature-view pipelines execute asynchronously on independent workers, each operating on a reduced representation. Adaptive excitation reconciles these pipelines with minimal coordination, making PAVE well suited to modern multi-core, distributed, and federated environments. For evaluation, we compare against strong gradient-boosted baselines on raw features, as well as fixed-view spectral variants. Empirically, spectral features alone can improve performance but are inconsistent across datasets, while forced view fusion often degrades accuracy. In contrast, PAVE consistently matches or exceeds the best-performing view, including the baseline, while eliminating manual tuning and exposing natural parallelism. We note that he reported speedups are relative to serial execution of PAVE's dual-view pipeline, not relative to highly optimized gradient boost libraries such as XGBoost. Overall, PAVE demonstrates how dual-view spectral learning can be exploited efficiently without fragile global graph constructions or serial bottlenecks.

PAVE is a structural organization layer over tabular representations wherein the method constructs alternative geometric organizations of the same data (sample-space and feature-space), evaluates them in parallel, and selects the representation that yields the most reliable downstream performance. In this sense, PAVE operates upstream of the learner: it determines which structural view of the data is most appropriate before standard predictors are applied.

To support reproducibility, the code and implementation details for this work are publicly available at https://github.com/MatthewFried/PAVE.

II. Related Work

Graph Laplacian construction is known to help in dimensionality reduction [2] and spectral clustering [3], with further conceptual developments analyzing kernel bandwidth

and the manifold boundary [4]. Various metrics have been applied [5] to understand the distribution of the data within both theory (such as the multiplicity of eigenvalues [6]) and application (such as studying the topological properties of graph Laplacians as applied to the brain).

Spectral clustering methods exploit the eigenvalues and eigenvectors of the Laplacian to partition data effectively, with algorithms optimized for parallel execution being particularly beneficial for handling large-scale datasets [7], with some approaches regarding graph Laplacians as a "Fourier transform" [8], and with newer approaches attempting to use spectral based methods on directed graphs [9].

While tabular data can be meaningfully converted to a graph by relaxing certain conditions [10], even survey papers discussing such techniques [11] for tabular data rarely study dual-view Laplacian pipelines with adaptive selection and explicit task-level parallelism, which is the focus here. Despite this, we still lean on standard graph methods such as applying mutual information and other statistics related to understanding sparsity [12], but highlight that our method is fast, lightweight, consistent, and highly parallelizable.

We note that unlike graph neural networks or end-to-end deep architectures, PAVE does not introduce a new parametric model over graphs. Instead, it organizes tabular data into competing structural representations and delegates prediction to standard learners. The contribution is therefore in representation selection and structural alignment, not in defining a new graph-based hypothesis class.

III. Technical Background

The graph Laplacian provides a matrix representation of a graph $G = (V, E)$ that encodes its structural and spectral properties. For an undirected weighted graph with adjacency matrix $A \in \mathbb{R}^{n\times n}$ and degree matrix D, where $D_{ii} = \sum_j A_{ij}$, the *normalized graph Laplacian* is defined as:

$$L_{\text{sym}} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}.$$

This normalization ensures numerical stability and allows spectral methods to account for varying vertex degrees. The matrix L_{sym} is symmetric and positive semi-definite, with eigen-decomposition

$$L_{\text{sym}}\mathbf{u}_k = \lambda_k \mathbf{u}_k, \quad 0 = \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n,$$

where the eigenvectors $\mathbf{u}_k$ capture the smoothest variations across the graph. These eigenvectors form a low-dimensional manifold embedding that preserves local connectivity [1] [13].

IV. Methodology

PAVE (Parallel Adaptive View Excitation) constructs two complementary spectral embeddings in parallel and selects the most effective representation for the target classification task. The algorithm operates in three phases: parallel view construction, adaptive selection, and final prediction. Each view constructs a graph (sparse kNN for samples, sparse top-k correlation for features), computes a Laplacian eigenmap, augments features, and trains a base learner. The views run concurrently , using a shared train/validation split, where the final model is chosen by validation accuracy.

We refer to this process as *view excitation*: constructing a structural representation for each candidate view, augmenting the original features accordingly, and selecting the view whose induced structure leads to the strongest validation performance.

In the first phase, that of parallel construction, given training data $\mathbf{X} \in \mathbb{R}^{n\times p}$ with labels $\mathbf{y} \in \{0,1\}^n$, we split training and validation sets: $\mathbf{X}_{\text{train}}, \mathbf{X}_{\text{val}}$. Two workers execute simultaneously:

Worker 1 (Sample View): Constructs a sample-sample similarity graph $\mathbf{W}^{(s)} \in \mathbb{R}^{n\times n}$ via sparse k-NN:

$$\mathbf{W}^{(s)} = (1-\lambda)\mathbf{W}^{(s)}_{\text{unsup}} + \lambda \mathbf{W}^{(s)}_{\text{sup}} \tag{1}$$

where the unsupervised part is $\exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2) \cdot \mathbb{I}(j \in \mathcal{N}_k(i))$ for k-nearest neighbors $\mathcal{N}_k(i)$, with adaptive bandwidth $\gamma = 1/(2\sigma^2_{\text{median}})$ computed from local k-th neighbor distances. The supervised component is $\mathbb{I}(y_i = y_j)$. We compute the normalized graph Laplacian $\mathbf{L}^{(s)} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}^{(s)}\mathbf{D}^{-1/2}$ and extract the first $d = 5$ non-trivial eigenvectors to form spectral features $\mathbf{Z}^{(s)} \in \mathbb{R}^{n\times d}$. For $n > 2000$, we use Nyström approximation with $m = 2000$ landmark points and interpolate embeddings for remaining samples via weighted k-NN.

Worker 2 (Feature View): Constructs a feature-feature similarity graph $\mathbf{W}^{(f)} \in \mathbb{R}^{p\times p}$:

$$\mathbf{W}^{(f)} = (1-\lambda)\mathbf{W}^{(f)}_{\text{unsup}} + \lambda \mathbf{W}^{(f)}_{\text{sup}} \tag{2}$$

where the unsupervised part is $|\text{corr}(\mathbf{x}_{\cdot a}, \mathbf{x}_{\cdot b})|$ (keeping only top-$k = 10$ correlations per feature for sparsity), and the supervised part is $\text{MI}(x_a, \mathbf{y}) \cdot \text{MI}(x_b, \mathbf{y})$ using mutual information normalized by the 90th percentile. Similarly, we compute the Laplacian, $\mathbf{L}^{(f)}$, and extract eigenvectors to form a basis $\mathbf{B}^{(f)} \in \mathbb{R}^{p\times d}$. Feature embeddings are obtained via projection: $\mathbf{Z}^{(f)} = \mathbf{X}\mathbf{B}^{(f)}$.

Both workers train gradient boost classifiers on augmented feature spaces: $\mathbf{X}^{(v)}_{\text{aug}} = [\mathbf{X} \,|\, \mathbf{Z}^{(v)}]$ for $v \in \{\text{sample}, \text{feature}\}$. A baseline model trained on $\mathbf{X}$ alone provides a fallback. All three models are validated on $\mathbf{X}_{\text{val}}$.

In words, we construct the sample or feature graph, compute the spectral view by extracting $d = 5$ eigenvectors (as a default), giving each sample $d = 5$ new manifold coordinates [as samples] or transform the data by extracting the eigenvectors from the projection basis [as features]. Each eigenvector is a coordinate system aligned with the manifold structure, where low eigenvalues represent smoothness, capturing clustering, density, and curvature. We further note that a lower number of eigenvectors may miss structure and a higher number of them may overfit the data. We then shrink the data by 0.4 or 0.15 (for small datasets) so as not to overwhelm the original features, concatenate with the original features, and train on a base learner. In our study, we used Gradient Boost with 100 estimators, a max depth of 4, a learning rate of 0.5, and 80%

subsampling per tree as our strong baseline that handles high-dimensional data and is relatively noncomplex.

In the second phase, adaptive selection, on each train/validation split, PAVE evaluates three candidate models: (1) baseline gradient enhancement on raw features, (2) sample-view enhancement with Laplacian graph eigenvectors from a sample similarity graph, and (3) feature-view enhancement with eigenvectors from a feature correlation graph (disabled if $p \leq 10$ as feature-graph construction becomes ill-conditioned in very low-dimensional settings). The model with highest validation accuracy is selected, with a small tolerance ($\epsilon = 0.001$) to prefer spectral views on near-ties. For datasets with $n \geq 400$, we do not do grid search in order to avoid overfitting.

In the third phase, prediction, for test sample $\mathbf{x}_{\text{test}}$, if $v^* =$ sample, we interpolate its embedding via weighted averaging over its 5 nearest training neighbors. If $v^* =$ feature, we project: $\mathbf{z}_{\text{test}}^{(f)} = \mathbf{x}_{\text{test}}^T \mathbf{B}^{(f)}$. The augmented test vector $[\mathbf{x}_{\text{test}} \,|\, \mathbf{z}_{\text{test}}^{(v^*)}]$ is classified by the selected model.

This approach works well as a graph weighted by similarity, extracting spectral information via the graph Laplacian. It captures the non-linear manifold structure and is effective to address clusters, curved manifolds, and local neighborhoods. In contrast to PCA which connects all points equally, the graph Laplacian connects similar neighborhoods.

All experiments use a fixed spectral configuration of $d = 5$ eigenvectors, top-$k = 10$ feature correlations, and at most $m = 2000$ Nyström landmarks. These values were selected based on pilot experiments to provide a stable and computationally lightweight representation across datasets. We intentionally keep these parameters fixed throughout to ensure a consistent comparison protocol and reproducibility of results.

We note that all graph constructions, embeddings, and spectral bases are learned exclusively on the training split. Validation and test samples are embedded using train-only objects (e.g., fixed neighborhood structures and projection operators) without recomputing graphs or affinities. Any supervised quantities used in view construction (e.g., label agreement or mutual information terms) are computed using training labels only, and are never exposed to validation or test labels.

Our sample view construction is $O(nk \log n + dk^2)$ for sparse k-NN graph and $O(d^3)$ for eigen-decomposition on $d \times d$ subspace (via Nyström). Feature view is $O(p^2 + pd^2)$ for correlation matrix and projection. Total parallel time is $T_{\|} = \max(T_{\text{sample}}, T_{\text{feature}}) + T_{\text{select}}$, compared to serial $T_{\text{serial}} = T_{\text{sample}} + T_{\text{feature}} + T_{\text{select}}$, yielding theoretical speedup of $S = T_{\text{serial}}/T_{\|} \approx 1.3$–$1.5\times$ with efficiency $\eta = S/2 \approx 65$–75% for two workers.

V. Numerical Analysis

We tested our data on an AWS C7i.24xlarge with 48 vCPU and 96 GB. All methods use identical preprocessing and fixed train/validation/test splits. Results are averaged over multiple random seeds. View selection is performed on the validation split only, and final test accuracy is evaluated once per split using the selected view refit on the training data.

Algorithm 1 PAVE: Parallel Adaptive View Excitation (workflow)

Require: Tabular data (X, y), base learner $\mathcal{A}$ (GBM), eigenvector count k, spectral feature scale α, supervision mix λ_{sup}, near-tie tolerance ϵ.

1: Split data into training and validation sets.
2: **Launch two independent tasks in parallel (sample-view, feature-view):**
3: **Sample-view pipeline:** build a sparse kNN graph over training *samples* (optionally mixed with label agreement via λ_{sup}); compute k Laplacian eigenvectors; treat them as k new sample coordinates; interpolate validation coordinates by nearest-neighbor weighting.
4: **Feature-view pipeline:** build a sparse graph over *features* (top correlations, optionally weighted by supervised relevance via λ_{sup}); compute k Laplacian eigenvectors; use them as a projection basis to produce k new features for train/val.
5: Train three predictors with the same learner $\mathcal{A}$:
6: **Base:** train on raw X only.
7: **Sample:** train on $[X \mid \alpha \cdot Z_{\text{sample}}]$ (raw features plus scaled sample-view spectral coordinates).
8: **Feature:** train on $[X \mid \alpha \cdot Z_{\text{feature}}]$ (raw features plus scaled feature-view spectral features; skip if very low-dimensional).
9: Evaluate validation accuracy of the three predictors.
10: Select the predictor with best validation accuracy; if the base is best but a spectral view is within ϵ, prefer the spectral view.
11: Refit the selected predictor on all training data using the same view; retain the non-selected view outputs for interpretation.
12: **Output:** Selected view and trained predictor.

A. Datasets Results

We show that standard datasets from UCI, OpenML, and PMLB, with varying sizes, produced consistent results along with parallelization speedups. 13 of the 16 datasets tested had a positive delta from PAVE, with the speedup noted in Table I. We see from the table that out of the 5 CV tests, most methods did better with the sample view, although outliers abound. While most datasets only received modest gains, spectral methods pushed the haberman dataset to excellent testing accuracy, as can be seen in Table II. We note that of the three datasets where PAVE underperformed the baseline, two showed only minimal degradation. The largest gain is observed on hill_valley, a dataset known to be highly sensitive to representation geometry. The baseline struggles due to nonlinear class structure, whereas PAVE consistently selects a graph-based view that better aligns with the underlying manifold. Results are stable across repeated runs with 3 different random seeds, indicating that the improvement reflects structural alignment rather than variance or split effects.

Our main argument is that the novel organization of spectral

TABLE I
COMPLETE RESULTS ACROSS 16 BENCHMARK DATASETS.

Dataset	Source	n	p	Baseline	PAVE	Δ	View	Speedup
haberman	PMLB	306	3	69.26%	69.60%	+0.34%	S	1.48×
hill_valley	OpenML	1212	100	54.40%	90.12%	+35.72%	F	1.19×
wine	sklearn	178	13	96.1%	97.8%	+1.70%	S	1.24×
ionosphere	OpenML	351	33	91.27%	91.27%	+0.00%	F	1.29×
saheart	PMLB	462	9	67.55%	69.71%	+2.16%	F	1.50×
bupa	PMLB	345	5	56.2%	56.8%	+0.60%	S	1.47×
breast_cancer	sklearn	569	30	93.68%	94.91%	+1.23%	F	1.24×
spectf	OpenML	349	44	89.98%	90.27%	+0.29%	F	1.21×
hepatitis	OpenML	155	19	84.52%	85.16%	+0.65%	F	1.23×
eeg-eye-state	OpenML	14980	14	91.2%	91.6%	+0.40%	F	1.47×
synthetic_dim	Synth	1000	200	84.2%	84.5%	+0.30%	F	1.11×
spambase	OpenML	4601	57	93.85%	94.25%	+0.39%	F	1.27×
digits_binary	UCI	1797	61	95.5%	98.89%	+3.39%	S	1.18×
phoneme	OpenML	5404	5	86.84%	86.74%	−0.10%	B	0.96×
vehicle	PMLB	846	18	79.67%	79.43%	−0.24%	B	1.19×
sonar	OpenML	208	60	83.81%	82.38%	−1.43%	F	1.23×

View: S = Sample, F = Feature, B = Baseline. View indicates the option selected most frequently across 3 random seeds; reported values are averages across seeds.

TABLE II
COMPREHENSIVE EVALUATION OF PAVE VS. BASELINE ACROSS MATCHED DATASETS FOR THE MOST SIGNIFICANT GAINS AND LOSSES. VALUES SHOW MEAN $\pm$ 95% CI; Δ INDICATES PAVE $-$ BASELINE. SIGNIFICANCE IS JUDGED BY $p < 0.05$ OR LARGE EFFECT SIZE ($|d| \geq 0.8$).

Dataset	ΔAcc (%)	95% CI	ΔF1	95% CI	ΔAUC	95% CI	p-value	Relevance
hill_valley	+35.72	±5.88	+0.360	±0.049	+0.377	±0.045	0.0004	Significant (p)
digits_binary	+3.39	±0.42	+0.034	±0.004	−0.004	±0.002	0.0001	Significant (p)
breast_cancer	+1.23	±0.92	+0.010	±0.007	+0.001	±0.001	0.0800	Large d=1.17
spambase	+0.37	±0.33	+0.005	±0.004	−0.002	±0.002	0.1193	Large d=0.99
ozone	−0.28	±0.36	−0.037	±0.054	+0.001	±0.007	0.2455	Medium d=0.68

pipelines of using the sample view or the feature view with spectral methods (i.e. a graph Laplacian) exhibits consistent performance under parallel execution, enhancing state of the art methods with little downside in practice.

B. Parallelization Results

We observed strong scaling with stable times from 6.6-9.8 across 2-32 workers. As Table III shows, we even attained a near-linear speedup for 32 workers, with none of the variance ranges impacting accuracy.

As can be seen in Table IV, efficiency was sustained between 92-100%, with negligible communication overhead, showing that our pipeline preserved model quality. Our weak scaling configuration eliminates data transfer by moving data generation onto the workers, which explains the near-ideal efficiency up to 32 workers. Furthermore, as shown in Table V our tests on five challenging synthetic datasets showed consistent (but small) gains. We thus argue that in highly parallelized environments, our model is an effective addition to state of the art techniques.

Our design aligns with modern HPC paradigms by minimizing synchronization and data movement. Each view executes embarrassingly in parallel with constant $O(1)$ communication per worker, enabling HPC-compliant spectral feature extraction without specialized hardware.

VI. CONCLUSION AND FUTURE WORK

This work introduces a parallel framework for organizing tabular data into competing structural representations and selecting between them. The emphasis is on identifying which geometric view of the data is most informative prior to learning.

In future work we hope to explore parallelizable ways to include adaptive eigenvalue selection based on validation loss or spectral gaps, creating a self-tuning spectral depth analysis. We also hope to explore multi-stage excitation, where refinement and training may work as a consensus mechanism between manifolds or models. With respect to limitations, while PAVE demonstrates consistent scaling and robustness, it is currently limited by fixed Laplacian hyperparameters and assumes moderate graph sparsity. We address this via adaptive eigenvalue selection and multi-stage excitation, but hope to expand to greater robustness going forward.

In summary, we have shown that PAVE adds value to state of the art methods by effectively exploiting spectral information in parallel. PAVE is a novel organization of spectral pipelines using a multi-view approach with stable strong scaling and efficient weak scaling. While it may seem straightforward, PAVE works on *general* tabular data, as shown from the 16 standard data sets. Advanced techniques such as pairwise and triplet search strategies, spectral clustering, and related

TABLE III
STRONG SCALING WITH REPEATABILITY (MEAN ± STD OVER 3 RUNS).

Workers	Time (s)	Speedup	Efficiency (%)	Accuracy (%)	Imbalance	Serial (%)
2	6.60(10)	1.77(3)	88.3(13)	92.27(116)	1.31(2)	13.3(17)
4	9.34(3)	2.50(1)	62.4(3)	93.87(31)	1.85(1)	20.1(2)
8	9.42(5)	4.95(3)	61.9(4)	95.17(38)	1.87(2)	8.8(1)
16	9.74(25)	9.58(25)	59.9(15)	95.17(38)	1.93(5)	4.5(3)
32	9.84(30)	18.96(58)	59.2(18)	95.27(31)	1.93(6)	2.2(2)

TABLE IV
WEAK SCALING WITH CONSTANT WORK PER WORKER (2,500 SAMPLES/WORKER).

Workers	n	n/worker	Time (s)	Efficiency (%)	Verdict
2	5000	2500	3.82	100.0	Excellent
4	10 000	2500	3.83	99.8	Excellent
8	20 000	2500	3.86	99.0	Excellent
16	40 000	2500	4.13	92.6	Excellent
32	80 000	2500	3.97	96.4	Excellent

TABLE V
COMPARISON OF PAVE AND XGBOOST ACROSS FIVE SYNTHETIC DATASETS.

Dataset	**PAVE Acc (%)**	**XGB Acc (%)**	**ΔAcc (%)**	**PAVE Time (s)**	**XGB Time (s)**	**Time Ratio***
synthetic_1	92.30	91.60	0.70	10.20	0.56	18.2
synthetic_2	92.20	92.15	0.05	42.99	1.16	37.1
synthetic_3	92.30	91.80	0.50	46.56	0.70	66.5
synthetic_4	91.70	91.10	0.60	20.24	0.93	21.8
synthetic_5	93.90	94.10	−0.20	50.50	0.94	53.7
Average	92.48	92.15	0.33	34.30	0.86	39.5

* Time Ratio = `PAVE Time` ÷ `XGB Time`; values > 1 indicate PAVE is slower.

TABLE VI
SCALABILITY VALIDATION: RUNTIME AND ACCURACY COMPARISON ON SYNTHETIC DATASETS OF INCREASING SIZE.

n	**Dense (s)**	**Sparse (s)**	**Speedup**	**Acc. Δ**
500	0.10	0.08	1.25×	−0.02%
2,000	1.20	0.40	3.0×	+0.05%
5,000	8.00	0.80	10.0×	+0.12%
10,000	30.00	2.50	12.0×	+0.08%

categorization methods yielded inconsistent outcomes, with performance degradation primarily attributed to overfitting and exponential increases in search complexity.

Overall, PAVE demonstrates that spectral adaptivity and parallel computation can be mutually reinforcing, opening a path towards more scalable and interpretable learning frameworks.

ACKNOWLEDGMENT

This work benefited from the use of AI-assisted tools for limited support in code generation and formatting. All core ideas, methods, experiments, and analyses were developed and validated by the author.

REFERENCES

[1] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[2] A. Berlinet and C. Thomas-Agnan, *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Springer New York, NY, 2004.

[3] U. von Luxburg, M. Belkin, and O. Bousquet, "Consistency of spectral clustering," *Annals of Statistics*, vol. 36, no. 2, pp. 555–586, 2008.

[4] X. Zhou and M. Belkin, "Behavior of graph laplacians on manifolds with boundary," *arXiv preprint arXiv:1105.3931*, 2011.

[5] B. Xie, M. Wang, and D. Tao, "Toward the optimization of normalized graph laplacian," *IEEE Transactions on Neural Networks*, vol. 22, no. 4, pp. 660–666, 2011.

[6] A. Banerjee and J. Jost, "On the spectrum of the normalized graph laplacian," *Linear Algebra and Its Applications*, vol. 428, pp. 3015–3022, 2008.

[7] O. E. Livne and A. Brandt, "Lean algebraic multigrid (lamg): fast graph laplacian linear solver," *SIAM Journal on Scientific Computing*, vol. 34, pp. B499–B522, 2012.

[8] X. Zhu and M. Rabbat, "Approximating signals supported on graphs," 2012, pp. 3921–3924.

[9] Y. Ma, J. Hao, Y. Yang, H. Li, J. Jin, and G. Chen, "Spectral-based graph convolutional network for directed graphs," *arXiv preprint arXiv:1907.08990*, 2019.

[10] M. Rabbani and A. Samad, "Between-sample relationship in learning tabular data using graph and attention networks," *arXiv preprint*

arXiv:2306.06772, 2023. [Online]. Available: https://arxiv.org/abs/2306.06772

[11] J. Li, Y.-H. H. Tsai, P.-Y. Chen, and B.-W. Liao, "Graph neural networks for tabular data learning: A survey," *arXiv preprint arXiv:2401.02143*, 2024. [Online]. Available: https://arxiv.org/abs/2401.02143

[12] K. Zhou, Z. Liu, R. Chen, L. Li, S.-H. Choi, and X. Hu, "Table2graph: Transforming tabular data to unified weighted graph," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, L. D. Raedt, Ed. International Joint Conferences on Artificial Intelligence Organization, 7 2022, pp. 2420–2426, main Track. [Online]. Available: https://doi.org/10.24963/ijcai.2022/336

[13] U. von Luxburg, "A tutorial on spectral clustering," *arXiv preprint arXiv:0711.0189*, vol. abs/0711.0189, 2007. [Online]. Available: http://arxiv.org/abs/0711.0189

[14] L. Zelnik-manor and P. Perona, "Self-tuning spectral clustering," in *Advances in Neural Information Processing Systems*, L. Saul, Y. Weiss, and L. Bottou, Eds., vol. 17. MIT Press, 2004. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2004/file/40173ea48d9567f1f393b20c855bb40b-Paper.pdf

[15] J. W. Peoples and J. Harlim, "Spectral convergence of symmetrized graph laplacian on manifolds with boundary," *Foundations of Data Science*, vol. 8, no. 0, p. 119–167, 2026. [Online]. Available: http://dx.doi.org/10.3934/fods.2025005

[16] M. Belkin, Q. Que, Y. Wang, and X. Zhou, "Graph laplacians on singular manifolds: toward understanding complex spaces: graph laplacians on manifolds with singularities and boundaries," *arXiv preprint arXiv:1211.6727*, 2012.

[17] X. Gao, H. Lü, and Y. Hao, "The laplacian and signless laplacian spectrum of semi-cayley graphs over abelian groups," *Journal of Applied Mathematics and Computing*, vol. 51, pp. 383–395, 2015.

[18] J. Yan, J. Chen, Y. Wu, D. Z. Chen, and J. Wu, "T2g-former: organizing tabular features into relation graphs promotes heterogeneous feature interaction," 2023. [Online]. Available: https://doi.org/10.1609/aaai.v37i9.26272

[19] C. Cucumides and F. Geerts, "augraph: Task-aware graph construction for relational and tabular learning," in *Proceedings of the VLDB Workshops (TaDA 2025)*. VLDB, 2025. [Online]. Available: https://www.vldb.org/2025/Workshops/VLDB-Workshops-2025/TaDA/TaDA25_5.pdf

[20] G. Bazhenov, O. Platonov, and L. Prokhorenkova, "Tabgraphs: A benchmark and strong baselines for learning on graphs with tabular node features," 2025. [Online]. Available: https://openreview.net/forum?id=a6XE2GJHjk

[21] C. Williams and M. Seeger, "Using the nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems*, T. Leen, T. Dietterich, and V. Tresp, Eds., vol. 13. MIT Press, 2000.

[22] C. Xu, D. Tao, and C. Xu, "A survey on multi-view learning," 2013. [Online]. Available: https://arxiv.org/abs/1304.5634

[23] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, ser. COLT' 98. New York, NY, USA: Association for Computing Machinery, 1998, p. 92–100. [Online]. Available: https://doi.org/10.1145/279943.279962

[24] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.290.5500.2319

[25] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting deep learning models for tabular data," ser. NIPS '21. Red Hook, NY, USA: Curran Associates Inc., 2021.