Meta-Graph Prototypical Diffusion for Tabular Classification

Boshko Koloski[1,2] **Nada Lavrač**[1] **Blaž Škrlj**[1]
[1] Jožef Stefan Institute, Ljubljana, Slovenia
[2] Jožef Stefan International Postgraduate School, Ljubljana, Slovenia
{boshko.koloski, nada.lavrac, blaz.skrlj}@ijs.si

Abstract

High-dimensional, low-sample-size (HDLSS) classification is a persistent challenge in microarray genomics and related domains, where limited samples, noisy features, and unreliable high-dimensional geometry confound standard methods. We propose MGPD, a meta-learning approach that learns compact embeddings, constructs and fuses multiple complementary graph representations with learned weights, and combines prototype-based global reasoning with local neighbor evidence via APPNP diffusion on strictly inductive soft-kNN graphs. Our hybrid readout incorporates balanced priors to handle class imbalance. On six microarray benchmarks, MGPD attains balanced accuracy on par with state-of-the-art general-purpose methods (RealMLP, TabPFN v2, TabICL) while achieving superior average AUPRC, demonstrating that compact, inductive graph-based architectures can compete with heavily pretrained tabular models on HDLSS tasks.

1 Introduction

HDLSS regimes, preeminent in microarray gene expression, strain both estimation and representation learning. When $n \ll p$, variance dominates, distances concentrate, and spurious nearest neighbors proliferate [16]. Traditional approaches offer mixed success in this setting. While gradient-boosted decision trees (GBDTs) such as XGBoost [7] and CatBoost [6] remain formidable on tabular data and recent work has clarified why tree ensembles often excel on typical tabular distributions [10] - they too struggle with extreme dimensionality. Modern neural baselines such as RealMLP [11], alongside pretrained and in-context approaches like TabPFN [12] and TabICL [15], have raised the bar for small-sample regimes through heavy pretraining, though they were designed for more moderate feature spaces. We ask whether a *small, strictly inductive* graph learner tailored to HDLSS can match these general methods on microarrays. We introduce **MGPD** (Meta–Graph Prototypical Diffusion with Multi–Graph Fusion), which learns what *similarity* means by fusing multiple standardized affinity functions [9], and how far to *propagate* by APPNP diffusion [1], before combining global prototypes [14] and local neighbor evidence with balanced priors [5]. The result is a simple, end–to–end pipeline designed for few–shot training and *strictly inductive* test–time inference.

Contributions. (1) A few–shot, end–to–end *inductive* meta–graph classifier that learns a convex fusion over standardized kernels and diffuses with APPNP. (2) A prototype/neighbor readout calibrated by balanced priors to stabilize minority classes. (3) A head–to–head evaluation on six microarray datasets against RealMLP [11], TabPFN v2 [12], TabICL, XGBoost [7], and CatBoost [6], with classical and Bayesian statistical comparisons.

The paper is structured as follows. We describe the proposed method in Section 2, followed by the experimental setup in Section 3 and results in Section 4. Limitations are discussed in Appendix D, and related work is presented in Appendix C.

AI for Tabular Data workshop at EurIPS 2025

2 Method

At a high level, MGPD–MF consists of three stages: (i) a small MLP maps high-dimensional inputs to compact embeddings; (ii) a bank of heterogeneous similarity graphs is constructed on these embeddings and fused into a single inductive soft-kNN graph; (iii) APPNP diffusion produces context-aware representations that are read out by a prototype/neighbor classifier calibrated with class priors. We detail each stage below.

Formal definition. Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ with $\mathbf{x}_i \in \mathbb{R}^p$ and $y_i \in \{1, \dots, C\}$. MGPD begins by learning a compact embedding $\phi : \mathbb{R}^p \to \mathbb{R}^d$ ($d \ll p$) via a small MLP with batch normalization and dropout, giving $\mathbf{z}_i = \phi(\mathbf{x}_i)$ and $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_n]^\top \in \mathbb{R}^{n \times d}$. To capture complementary inductive biases, we compute a *bank* of similarity matrices $\{\mathbf{S}^{(m)}\}_{m=1}^M$ on $\mathbf{Z}$: cosine; correlation–like via row centering and unit normalization; RBF kernels with $\gamma \in \{0.25, 0.5, 1.0\}$ applied to dimension–normalized squared distances; a shrinkage–Mahalanobis score using covariance fit on supports; and a learned bilinear form

$$S_{ij}^{\text{bilin}} = \frac{1}{\tau\sqrt{d}}\, \mathbf{z}_i^\top \mathbf{W} \mathbf{z}_j, \qquad \tau > 0. \tag{1}$$

Because these scores live on different scales, we standardize each by its off–diagonal mean and standard deviation,

$$\bar{\mathbf{S}}^{(m)} = \frac{\mathbf{S}^{(m)} - \mu_m}{\sigma_m + \varepsilon}, \qquad \mu_m = \text{mean}\{S_{ij}^{(m)} : i \neq j\}, \quad \sigma_m = \text{std}\{S_{ij}^{(m)} : i \neq j\}, \tag{2}$$

and learn fusion logits $\theta \in \mathbb{R}^M$ whose softmax gives weights $w = \text{softmax}(\theta)$ for a fused similarity

$$\mathbf{S}_{\text{fused}} = \sum_{m=1}^{M} w_m\, \bar{\mathbf{S}}^{(m)}. \tag{3}$$

From each row of $\mathbf{S}_{\text{fused}}$ we select the top–k entries (excluding self) and apply a row–softmax to obtain a *row–stochastic* soft–kNN adjacency $\mathbf{A}$. During training episodes we *mask* the query–to–query block of $\mathbf{A}$ to respect strict inductivity. We then diffuse with APPNP [1]: with $\mathbf{H}^{(0)} = \mathbf{Z}$ and teleport $\alpha \in (0, 1]$,

$$\mathbf{H}^{(t+1)} = (1 - \alpha)\, \mathbf{A}\, \mathbf{H}^{(t)} + \alpha\, \mathbf{H}^{(0)}, \qquad t = 0, \dots, K-1, \quad \mathbf{H} = \mathbf{H}^{(K)}. \tag{4}$$

Class prototypes are means over diffused supports, $\mathbf{p}_c = \frac{1}{|\mathcal{S}_c|} \sum_{i \in \mathcal{S}_c} \boldsymbol{h}_i$. For a query $\boldsymbol{h}_q$, we compute a prototype similarity $s_c^{\text{proto}} = \frac{\boldsymbol{h}_q^\top \mathbf{p}_c}{\|\boldsymbol{h}_q\|\, \|\mathbf{p}_c\|}$ and a neighbor evidence term $s_c^{\text{nbr}} = \sum_{i \in \mathcal{S}_c} A_{q,i}$. A learned gate $\beta = \sigma(\beta_{\text{raw}}) \in (0, 1)$ mixes them, and we add $\log \pi_c$ (empirical priors) to obtain logits/probabilities

$$\ell_c = \beta\, s_c^{\text{proto}} + (1 - \beta)\, s_c^{\text{nbr}} + \log \pi_c, \qquad p(y = c \mid \boldsymbol{h}_q) = \frac{e^{\ell_c}}{\sum_{c'} e^{\ell_{c'}}}. \tag{5}$$

Additional details on the model components and loss functions can be found in Appendices A and B.

Training objective and episodes. Training proceeds via few–shot episodes: per class we sample supports and queries, fit the Mahalanobis covariance on supports, compute/standardize/fuse kernels, build $\mathbf{A}$ with the query–mask, diffuse via (4), and evaluate (5) on queries. We minimize a composite objective

$$\mathcal{L} = \lambda_{\text{CE}}\mathcal{L}_{\text{CE}} + \lambda_{\text{SupCon}}\mathcal{L}_{\text{SupCon}} + \lambda_{\text{edge}}\mathcal{L}_{\text{edge}} + \lambda_{\text{lap}}\mathcal{L}_{\text{lap}} + \lambda_{\text{mix}}\mathcal{R}_{\text{mix}},$$

with precise definitions and roles detailed in Appendix A. At test time, each query connects only to training nodes, reusing cached train–train diffusion (strictly inductive inference). The model surfaces kernel weights and the prototype/neighbor decomposition for light–touch inspection.

3 Experimental setup

Datasets. Six scikit–feature[1] microarray datasets: `ALLAML`, `Prostate_GE`, `SMK_CAN_187`, `TOX_171`, `colon`, `leukemia`. We perform 3 times repeated 5-fold CV per dataset. More details are provided

[1] `https://jundongl.github.io/scikit-feature/`

in Appendix E. Unless otherwise stated, we report *balanced accuracy* (Bal. Acc), area under the precision–recall curve (AUPRC), and area under the ROC curve (AUROC).

Compared models. *MGPD* (ours) versus **RealMLP** [11], **TabPFN v2** [12], **TabICL**, and **XGBoost/CatBoost** [7, 6]. Implementations for XGBoost/CatBoost/RealMLP follow the codebase [11]; TabICL and TabPFN v2 use authors' public implementations.

Hyperparameters (MGPD). Embedding dimension d=32, MLP depth 3 (hidden 128, dropout 0.3), k=5, APPNP K=3, α=0.1, SupCon temperature 0.07; loss weights: CE 1.0, SupCon 0.6, edge 0.12, Laplacian 0.05, mix–entropy 0.01; RBF $\gamma \in \{0.25, 0.5, 1.0\}$; Mahalanobis shrinkage 0.1; 50 episodes/epoch, 200 epochs; per–class supports in $[2, 5]$.

4 Results and Discussion

On balanced accuracy (Table 1), TabPFN v2 attains the best overall mean, with MGPD a close second (average gap ≈0.005) and the strongest result on `colon`. On AUPRC (Table 2), MGPD achieves the best mean and best mean rank, consistent with our design that aggregates local neighbor evidence with global prototypes and calibrates logits by class priors. Following a conservative protocol that *forced* a nonparametric omnibus test, the Friedman test [8] across six populations (six datasets, 15 paired scores each) yielded p=0.111; a Nemenyi post–hoc with CD=3.078 found no mean–rank gaps exceeding the critical distance. A complementary hierarchical Bayesian comparison [2] (balanced accuracy; ROPE= 0.01) between MGPD and TabPFN v2 estimated $p(\text{MGPD} < \text{TabPFN})$=0.2308, $p(|\Delta| \leq \text{ROPE})$=0.2010, and $p(\text{MGPD} > \text{TabPFN})$=0.5682, indicating practical parity with a slight posterior advantage for MGPD.

Table 1: Balanced accuracy (mean ± std over 15 runs). Best per–row in bold.

Dataset	catboost_td	**MGPD**	realmlp_td	tabicl	tabpfnv2	xgb_td
ALLAML	0.9007 ± 0.0889	0.9556 ± 0.0540	**0.9733** ± 0.0458	0.9356 ± 0.0996	0.9567 ± 0.0495	0.9226 ± 0.1004
Prostate_GE	0.9018 ± 0.0728	0.8964 ± 0.0751	0.8961 ± 0.0778	0.9055 ± 0.0544	**0.9445** ± 0.0446	0.9348 ± 0.0536
SMK_CAN_187	0.6734 ± 0.0814	0.7068 ± 0.0745	0.6914 ± 0.0602	0.6886 ± 0.0632	**0.7217** ± 0.0831	0.6926 ± 0.0716
TOX_171	0.7404 ± 0.0772	0.9581 ± 0.0375	0.9259 ± 0.0275	0.8904 ± 0.0616	**0.9593** ± 0.0367	0.8311 ± 0.0893
colon	0.7258 ± 0.1118	**0.8458** ± 0.0741	0.7783 ± 0.1244	0.8192 ± 0.0751	0.8025 ± 0.0865	0.7708 ± 0.1191
leukemia	0.9522 ± 0.0565	0.9407 ± 0.0847	**0.9733** ± 0.0594	0.9067 ± 0.0863	0.9489 ± 0.0829	0.9496 ± 0.0655
Average	0.8157	0.8839	0.8731	0.8577	**0.8889**	0.8503
Mean rank	5.00	3.00	3.17	4.00	**2.00**	3.83

Table 2: AUPRC (mean ± std over 15 runs). Best per–row in bold.

Dataset	catboost_td	**MGPD**	realmlp_td	tabicl	tabpfnv2	xgb_td
ALLAML	0.4818 ± 0.0230	**0.5009** ± 0.0531	0.4732 ± 0.0127	0.4776 ± 0.0205	0.4710 ± 0.0103	0.4790 ± 0.0329
Prostate_GE	0.3340 ± 0.0122	**0.3489** ± 0.0204	0.3322 ± 0.0123	0.3329 ± 0.0138	0.3270 ± 0.0099	0.3319 ± 0.0138
SMK_CAN_187	0.3893 ± 0.0355	0.3850 ± 0.0390	**0.4017** ± 0.0540	0.3889 ± 0.0384	0.3557 ± 0.0296	0.3886 ± 0.0474
TOX_171	0.8597 ± 0.0650	0.9718 ± 0.0287	0.9793 ± 0.0338	0.9687 ± 0.0222	**0.9893** ± 0.0138	0.9124 ± 0.0631
colon	0.7769 ± 0.1319	0.8209 ± 0.1169	0.7631 ± 0.1467	0.8311 ± 0.1539	**0.8449** ± 0.1242	0.7824 ± 0.1542
leukemia	0.9726 ± 0.0542	0.9803 ± 0.0413	0.9836 ± 0.0350	0.9806 ± 0.0366	**0.9929** ± 0.0202	0.9917 ± 0.0179
Average	0.6357	**0.6680**	0.6555	0.6633	0.6635	0.6477
Mean rank	3.83	**3.00**	3.50	3.33	3.50	3.83

4.1 Ablation study

To assess the contributions of MGPD's components, we conduct an ablation study on `ALLAML`.

Loss components. Removing **supervised contrastive loss** yields the largest drop in balanced accuracy ($\Delta = -0.0248$), while *slightly increasing* AUPRC ($+0.0062$). **APPNP diffusion** is most critical for AUPRC (-0.0235) and also harms Bal. Acc (-0.0215). **Laplacian regularization** modestly supports Bal. Acc (-0.0148) with negligible AUPRC change ($+0.0026$). **Edge supervision** and **mix entropy** have small effects (Bal. Acc drops 0.0096 and 0.0074; AUPRC drops 0.0065 and 0.0133), suggesting potential simplification. See Table 5.

Prototype vs. neighbor readout. Using **neighbors only** markedly improves AUPRC to 0.6078 ($+0.1122$ over full) but reduces Bal. Acc to 0.9452 (-0.0074) and AUROC to 0.9445 (-0.0269). **Prototypes only** similarly lower AUPRC (0.4810) and Bal. Acc (0.9452) but yield higher AUROC

(0.9837). The learned gate β in the *full model* blends both signals and achieves the best Bal. Acc (0.9526). See Table 3.

Kernel fusion. Among single–kernel additions to the learned bilinear form, **RBF** with $\gamma = 1.0$ gives the highest Bal. Acc (0.9667). **Bilinear only** is competitive on AUPRC (0.4919). The *fused* model (all seven kernels) attains the best overall AUPRC (0.4956) while maintaining strong Bal. Acc (0.9526), trading a small amount of Bal. Acc for improved precision-recall. See Table 4.

Training regime. On ALLAML, graph–only (frozen embeddings) and end–to–end training perform equivalently (Bal. Acc 0.9526, AUPRC 0.4956), indicating that most signal is captured by the graph module; graph–only inference may suffice when efficiency is prioritized.

Bal. Acc is primarily supported by supervised contrastive learning and APPNP, while neighbor evidence is especially valuable for minority–class precision-recall. Kernel fusion offers robustness: RBF improves class separation, and full fusion yields the best AUPRC with competitive Bal. Acc.

Table 3: Prediction strategy ablation on ALLAML (3 rep $\times$ 5 folds). Prototype–only uses $\beta = 1$; neighbor–only uses $\beta = 0$; full model learns β.

Readout strategy	Bal. Acc	Δ Bal. Acc	AUPRC	Δ AUPRC
Full model (learned β)	**0.9526**	–	0.4956	–
Prototype–only ($\beta = 1$)	0.9452	−0.0074	0.4810	−0.0146
Neighbor–only ($\beta = 0$)	0.9452	−0.0074	**0.6078**	+0.1122

Table 4: Kernel combination ablation on ALLAML (3 rep $\times$ 5 folds). All configurations use the learned bilinear kernel plus the specified additional kernel(s). The full model fuses all seven kernels.

Kernel configuration	Bal. Acc	AUPRC	AUROC
Bilinear only	0.9563	**0.4919**	0.9844
Bilinear + Cosine	0.9374	0.4898	0.9543
Bilinear + Correlation	0.9389	0.4851	0.9733
Bilinear + RBF ($\gamma = 0.25$)	0.9630	0.4858	0.9777
Bilinear + RBF ($\gamma = 0.5$)	0.9489	0.4881	0.9700
Bilinear + RBF ($\gamma = 1.0$)	**0.9667**	0.4776	0.9853
Bilinear + Mahalanobis	0.9552	0.4878	0.9733
Full model (all kernels fused)	0.9526	0.4956	0.9714

Table 5: Loss component ablation on ALLAML (3 rep $\times$ 5 folds). Each row shows performance when the specified component is *removed* and the model is retrained. Negative drops indicate improvement.

Component removed	Bal. Acc	Δ Bal. Acc	AUPRC	Δ AUPRC
(Full model)	0.9526	–	0.4956	–
Supervised contrastive	0.9278	0.0248	0.5018	−0.0062
APPNP diffusion	0.9311	0.0215	0.4721	0.0235
Laplacian regularization	0.9378	0.0148	0.4982	−0.0026
Edge supervision	0.9430	0.0096	0.4891	0.0065
Mix entropy	0.9452	0.0074	0.4823	0.0133

5 Conclusion

We presented **MGPD**, a small, strictly inductive meta-graph learner for HDLSS classification. By fusing heterogeneous similarities after off-diagonal standardization, diffusing with APPNP, and balancing prototype with neighbor evidence under class priors, MGPD matches TabPFN v2 on balanced accuracy and leads on AUPRC across six microarray datasets, despite clear disparity in parameter count and pre-training. Under conservative nonparametric testing, we observe no significant differences; Bayesian analysis suggests practical parity. Ablations confirm that performance hinges on supervised contrastive learning and APPNP diffusion, while the hybrid prototype-neighbor readout achieves the best balanced accuracy by blending global and local evidence, further supporting the necessity of the two distinct information streams.

Acknowledgments

We acknowledge the financial support of the Slovenian Research and Innovation Agency (ARIS) through the core research programme P2-0103 (Knowledge Technologies). The work of B.K. was supported by the Young Researcher Grant PR-12394.

References

[1] Klicpera, J., Bojchevski, A., & Günnemann, S. (2019). Predict then propagate: Graph neural networks meet personalized PageRank. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

[2] Benavoli, A., Corani, G., Demšar, J., & Zaffalon, M. (2017). Time for a change: A tutorial for comparing multiple classifiers through Bayesian analysis. *Journal of Machine Learning Research*, 18(77), 1-36.

[3] Koloski, B., Škrlj, B., Stepišnik, T., Pollak, S., & Lavrač, N. (2024). HorNets: Learning from discrete and continuous signals with routing neural networks. *Machine Learning*, 113, 4965-5002. `https://doi.org/10.1007/s10994-024-06673-1`

[4] Jiang, X., Margeloiu, A., Simidjievski, N., & Jamnik, M. (2024). ProtoGate: Prototype-based neural networks with global-to-local feature selection for tabular biomedical data. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, vol. 235, 21844-21878.

[5] Ren, J., Yu, C., Ma, X., Zhao, H., Yi, S., et al. (2020). Balanced meta-softmax for long-tailed visual recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 4175-4186.

[6] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 6638-6648.

[7] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 785-794.

[8] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7, 1-30.

[9] Gönen, M., & Alpaydın, E. (2011). Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12, 2211-2268.

[10] Grinsztajn, L., Oyallon, E., & Varoquaux, G. (2022). Why do tree-based models still outperform deep learning on typical tabular data? In *Advances in Neural Information Processing Systems (NeurIPS)*, 35, 507-520.

[11] Gorishniy, Y., Rubachev, I., & Babenko, A. (2022). Revisiting deep learning models for tabular data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 35, 18932-18943.

[12] Hollmann, N., Müller, S., Eggensperger, K., & Hutter, F. (2023). TabPFN: A transformer that solves small tabular classification problems in a second. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, 13252-13282.

[13] Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., & Krishnan, D. (2020). Supervised contrastive learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 18661-18673.

[14] Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 30, 4077-4087.

[15] Qu, J., Holzmüller, D., Varoquaux, G., & Le Morvan, M. (2025). TabICL: A tabular foundation model for in-context learning on large data. arXiv preprint arXiv:2502.05564.

[16] Andrei Margeloiu, Nikola Simidjievski, Pietro Liò, and Mateja Jamnik, "GC_{ond}Net: A Novel Method for Improving Neural Networks on Small High-Dimensional Tabular Data," in *NeurIPS 2023 Second Table Representation Learning Workshop*, 2023. Available at: https://openreview.net/forum?id=jSebr3OJA2.

[17] Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2018). Feature selection: A data perspective. *ACM Computing Surveys*, 50(6), 94.

[18] Li, J. (2018). scikit-feature: Feature selection repository in Python. Retrieved from https://jundongl.github.io/scikit-feature/

A Appendix A: Graph components and operators

Embedding. $\phi(\mathbf{x}) = \mathbf{W}_L\sigma(\mathrm{BN}(\cdots\sigma(\mathrm{BN}(\mathbf{W}_1\mathbf{x}))\cdots)) \in \mathbb{R}^d$; collect $\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_n]^\top$.

Cosine. With $\tilde{\mathbf{z}}_i = \mathbf{z}_i/\|\mathbf{z}_i\|$, $S_{ij}^{\cos} = \tilde{\mathbf{z}}_i^\top\tilde{\mathbf{z}}_j$.

Correlation–like. Row–centering and unit–norm: $\hat{\mathbf{z}}_i = (\mathbf{z}_i - \bar{\mathbf{z}}_i)/\|\mathbf{z}_i - \bar{\mathbf{z}}_i\|$, $S_{ij}^{\mathrm{corr}} = \hat{\mathbf{z}}_i^\top\hat{\mathbf{z}}_j$.

RBF (multi–scale). With $D_{ij}^2 = \frac{\|\mathbf{z}_i - \mathbf{z}_j\|^2}{d}$, $S_{ij}^{\mathrm{rbf}_\gamma} = \exp(-\gamma D_{ij}^2)$, $\gamma \in \{0.25, 0.5, 1.0\}$.

Shrinkage Mahalanobis. Center $\mathbf{Z}_c$, $\hat{\mathbf{\Sigma}} = \frac{1}{n-1}\mathbf{Z}_c^\top\mathbf{Z}_c$. Shrink $\mathbf{\Sigma} = (1-\lambda)\hat{\mathbf{\Sigma}} + \lambda\frac{\mathrm{tr}(\hat{\mathbf{\Sigma}})}{d}\mathbf{I} + \epsilon\mathbf{I}$, then $S_{ij}^{\mathrm{maha}} = -\frac{1}{2}(\mathbf{z}_i - \mathbf{z}_j)^\top\mathbf{\Sigma}^{-1}(\mathbf{z}_i - \mathbf{z}_j)$.

Learned bilinear. $S_{ij}^{\mathrm{bilin}} = \frac{1}{\tau\sqrt{d}}\mathbf{z}_i^\top\mathbf{W}\mathbf{z}_j$ with learnable $\mathbf{W}$ and temperature τ.

Off–diagonal standardization and fusion. Apply (2) per score matrix; fuse by (3) with $w = \mathrm{softmax}(\theta)$.

Soft–kNN (row–stochastic). For row i, let $\mathcal{N}_k(i)$ be top–k indices of $\mathbf{S}_{\text{fused},i:}$ (excluding i). Define

$$A_{ij} = \begin{cases} \dfrac{\exp(S_{\text{fused},ij})}{\sum_{j'\in\mathcal{N}_k(i)}\exp(S_{\text{fused},ij'})}, & j \in \mathcal{N}_k(i), \\ 0, & \text{otherwise.} \end{cases}$$

During training episodes, set $\mathbf{A}[\mathcal{Q}, \mathcal{Q}] = \mathbf{0}$.

APPNP diffusion. $\mathbf{H}^{(0)} = \mathbf{Z}$, $\mathbf{H}^{(t+1)} = (1-\alpha)\mathbf{A}\mathbf{H}^{(t)} + \alpha\mathbf{H}^{(0)}$; $\mathbf{H} = \mathbf{H}^{(K)}$ [1].

Appendix B: Losses - definitions and roles

Below, *Inputs* list the tensors each term consumes, and *Role* explains its function within MGPD. The composite objective is

$$\mathcal{L} = \lambda_{\mathrm{CE}}\mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{SupCon}}\mathcal{L}_{\mathrm{SupCon}} + \lambda_{\mathrm{edge}}\mathcal{L}_{\mathrm{edge}} + \lambda_{\mathrm{lap}}\mathcal{L}_{\mathrm{lap}} + \lambda_{\mathrm{mix}}\mathcal{R}_{\mathrm{mix}}.$$

Balanced cross–entropy ($\mathcal{L}_{\mathrm{CE}}$). *Definition.* For query set $\mathcal{Q}$, logits ℓ_c in (5), and probabilities $p_c = \frac{e^{\ell_c}}{\sum_{c'} e^{\ell_{c'}}}$,

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{|\mathcal{Q}|}\sum_{q\in\mathcal{Q}}\log p_{y_q}.$$

Equivalently, add $\log\pi_c$ to pre–softmax logits ("balanced softmax") [5]. *Inputs.* Query logits from prototype/neighbor fusion (β), class priors π. *Role.* Calibrates decision boundaries under class imbalance; reduces bias toward frequent classes.

Supervised contrastive loss ($\mathcal{L}_{\text{SupCon}}$) [13]. *Definition.* Let normalized embeddings $\tilde{\boldsymbol{h}}_i = \boldsymbol{h}_i/\|\boldsymbol{h}_i\|$, temperature T, and positives $P(i) = \{j \neq i : y_j = y_i\}$. Then

$$\mathcal{L}_{\text{SupCon}} = -\frac{1}{\sum_i |P(i)|} \sum_i \sum_{p \in P(i)} \log \frac{\exp(\tilde{\boldsymbol{h}}_i^\top \tilde{\boldsymbol{h}}_p/T)}{\sum_{a \neq i} \exp(\tilde{\boldsymbol{h}}_i^\top \tilde{\boldsymbol{h}}_a/T)}.$$

Inputs. Diffused representations $\mathbf{H}$ (supports and queries), labels. *Role.* Tightens class clusters and enlarges inter–class margins in the representation used by graph construction and prototypes.

Edge supervision ($\mathcal{L}_{\text{edge}}$). *Definition.* On supports $\mathcal{S}$, with fused scores $Z_{ij} = S_{\text{fused},ij}$ and targets $Y_{ij} = \mathbb{1}[y_i = y_j]$ for $i \neq j$, use BCE with positive weight ω:

$$\mathcal{L}_{\text{edge}} = \frac{1}{|\{(i,j) \in \mathcal{S}^2 : i \neq j\}|} \sum_{i \neq j} [-\omega Y_{ij} \log \sigma(Z_{ij}) - (1 - Y_{ij}) \log(1 - \sigma(Z_{ij}))] \,.$$

Inputs. Fused score matrix on supports; labels. *Role.* Encourages same–class pairs to score higher than different–class pairs, guiding the kernel fusion toward class–aware neighborhoods.

Laplacian regularization ($\mathcal{L}_{\text{lap}}$). *Definition.* With the symmetrized Laplacian $\mathbf{L} = \mathbf{D} - \frac{1}{2}(\mathbf{A} + \mathbf{A}^\top)$ and pre–diffusion embeddings $\mathbf{Z}$,

$$\mathcal{L}_{\text{lap}} = \frac{1}{n} \text{tr}(\mathbf{Z}^\top \mathbf{L} \mathbf{Z}).$$

Inputs. Row–stochastic adjacency $\mathbf{A}$; embeddings $\mathbf{Z}$. *Role.* Promotes local smoothness of representations along the learned graph, reducing variance in HDLSS.

Fusion entropy ($\mathcal{R}_{\text{mix}}$). *Definition.* For mixture weights $w = \text{softmax}(\theta)$ over kernels,

$$\mathcal{R}_{\text{mix}} = -\sum_{m=1}^{M} w_m \log w_m.$$

Inputs. Fusion weights w. *Role.* Discourages collapse onto a single kernel; encourages the model to leverage multiple, complementary similarity notions [9].

Composite fusion. *Inputs.* The five terms above; hyperparameters $\lambda_\bullet$. *Role.* The composite blends label–driven discrimination (CE, SupCon), graph shaping (edge, Laplacian), and kernel diversity (entropy) to produce robust, class–aware neighborhoods and stable diffusion in few–shot HDLSS settings.

Appendix C: Related work

GBDTs such as XGBoost [7] and CatBoost [6] remain strong on tabular data due to additive modeling over decision rules, effective regularization (shrinkage, subsampling), and native handling of heterogeneity. Recent analyses explain why tree-based models still outperform deep networks on common tabular regimes [10]. Neural baselines like RealMLP [11] have narrowed this gap through careful architectures and training recipes.

Pretrained and in-context approaches offer a different paradigm. TabPFN amortizes Bayesian inference over synthetic tasks with a transformer [12], while TabICL [15] employs a two-stage architecture: column-then-row attention builds fixed-dimensional row embeddings, which a transformer then processes for efficient in-context classification. Both rely on extensive pretraining over moderate-dimensional feature spaces.

MGPD complements these lines by targeting HDLSS directly. It combines multi-kernel fusion [9], APPNP diffusion [1], and prototype-based few-shot reasoning [14] in a compact, strictly inductive graph learner specialized for extreme dimensionality.

Table 6: Dataset statistics (from the SCIKIT-FEATURE repository [17]). n = number of samples, p = number of features, C = number of classes.

Dataset	n	p	C
ALLAML	72	7129	2
Prostate_GE	102	5966	2
SMK_CAN_187	187	19993	2
TOX_171	171	5748	4
colon	62	2000	2
leukemia	72	7070	2

Appendix D: Limitations and Future Work

While MGPD achieves strong performance on microarray benchmarks, several limitations highlight directions for future research. Our evaluation focuses on general-purpose tabular models (RealMLP, TabPFN, TabICL, XGBoost, CatBoost) rather than HDLSS-specific methods. Comparing MGPD with recent specialized architectures such as HorNets [3] and ProtoGate [4] would clarify its relative strengths and complementarities, particularly regarding higher-order feature modeling and gating mechanisms. The multi-graph fusion process introduces computational overhead: constructing M similarity matrices scales as $O(Mn^2d)$, and the bilinear form adds $O(d^2)$ parameters. Although APPNP diffusion mitigates oversmoothing, scalability may be limited for large n or d; future work could explore sparse or low-rank approximations. MGPD also involves several hyperparameters—embedding dimension, neighborhood size, diffusion depth, teleport probability, and loss weights—whose sensitivity remains underexplored. Automated or meta-learned tuning strategies may enhance robustness across domains. Experiments are currently restricted to microarray data; extending evaluation to other high-dimensional domains (e.g., text embeddings, finance, neuroimaging) would test generalization and adaptability of graph construction strategies. Finally, the current framework targets classification tasks. Extending MGPD to regression, multi-label, or survival analysis could broaden its applicability in genomics and other scientific settings.

Appendix E: Datasets

We evaluate MGPD–MF on six benchmark microarray datasets from the SCIKIT-FEATURE repository [17]: ALLAML, Prostate_GE, SMK_CAN_187, TOX_171, colon, and leukemia. All of them are high-dimensional, low-sample-size (HDLSS) gene expression datasets with binary or small multi-class labels. Table 6 summarizes the number of samples n, features p, and classes C.

These datasets lie firmly in the HDLSS regime with n in the [62, 187] range and p between $2{,}000$ and $19{,}993$ features. Following common practice for microarray benchmarks, we treat each dataset as a separate supervised classification task and do not apply any task-specific feature selection prior to training. Performance is reported using repeated cross-validation (3 repetitions of 5-fold CV, yielding 15 scores per dataset) to reduce variance in the small-sample setting.