GSSNN: Graph Smoothing Splines Neural Networks

Shichao Zhu[1,3], Lewei Zhou[2,4], Shirui Pan[5], Chuan Zhou[2,3], Guiying Yan[2,4] and Bin Wang[6]

[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

[2]Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China

[3]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

[4] University of Chinese Academy of Sciences, Beijing, China

[5]Faculty of Information Technology, Monash University, Melbourne, Australia [6]Xiaomi AI Lab, Xiaomi Inc., Beijing, China

AAAI-20; 7-12 February 2020, New York, USA

Outline

- Introduction
  - Preliminaries
  - Motivation
- Approach
  - GSSNN: Graph Smoothing Splines Neural Networks
  - Overall Model
- Experiments
  - Settings
  - Results and Analysis
- Conclusion

Outline

- Introduction
  - Preliminaries
  - Motivation
- Approach
  - GSSNN: Graph Smoothing Splines Neural Networks
  - Overall Model
- Experiments
  - Settings
  - Results and Analysis
- Conclusion

Preliminaries

- Graph-level Representation Learning
  - Definition: Given a set of graphs $\mathcal{G} = \{G_i\}_i^t$, learn a mapping function: $\mathcal{G} \rightarrow \mathbb{R}^n$ that project each graph G_i into low dimensional vectors in space $\mathbb{R}^n$.

  - Existing Methods
    - Kernel-based methods
    - GNN-based methods

Preliminaries

- Existing Methods
  - Kernel-based methods
    - Intuition: decompose graph into sub-components → build graph embedding in feature-based manner → apply ML algorithms to perform graph classification
    - Works: Weisfeiler-Lehman subtree kernel (WL) [1], graphlet count kernel (GK) [2], Random Walk (RW) [3]
  - GNN-based methods

Preliminaries

- Existing Methods
  - Kernel-based methods
  - GNN-based methods
    1. Graph Summarization: collect the embedding for all nodes to generate graph representation
       - Works: GCAPS-CNN [4], CapsGNN [5], GIN [6]
    2. Graph Pooling: reduce the size of nodes to coarsen the graph progressively through learning topology-based node assignments
       - Global pooling methods: Set2Set [7], SortPool [8]
       - Hierarchical pooling methods: DiffPool [9], SAGPool [10]

Preliminaries

- Existing Methods
  - Kernel-based methods
  - GNN-based methods → Only exploit local information via convolution or neighbor aggregation

1. Graph Summarization: collect the embedding for all nodes to generate graph representation

   Cannot distinguish the importance of different nodes

2. Graph Pooling: reduce the size of nodes to coarsen the graph progressively through learning topology-based node assignments

   Loss some important information for nodes

Non-smoothing node features

- GNN *aggregation* operation

  ① Applying a feature fitting function $g(X) = XW$

  ② Propagating the new representation $A \cdot g(X)$

  ③ Fitting it into a nonlinear activation function

Result in degenerated node embedding due to the non-smooth feature fitting function $g(X)$

- Node-level representation → Graph-level representation

Non-smoothing node features + Noise features

Suboptimal graph embedding

Motivation

- Limitations of existing methods
  - Non-smoothing node features → suboptimal embedding
  - Ignore global topological knowledge
  - Lack of interpretability

Motivation

- Limitations of existing methods
  - Non-smoothing node features → suboptimal embedding
  - Ignore global topological knowledge
  - Lack of interpretability

  - *How do we overcome these limitations uniformly?*

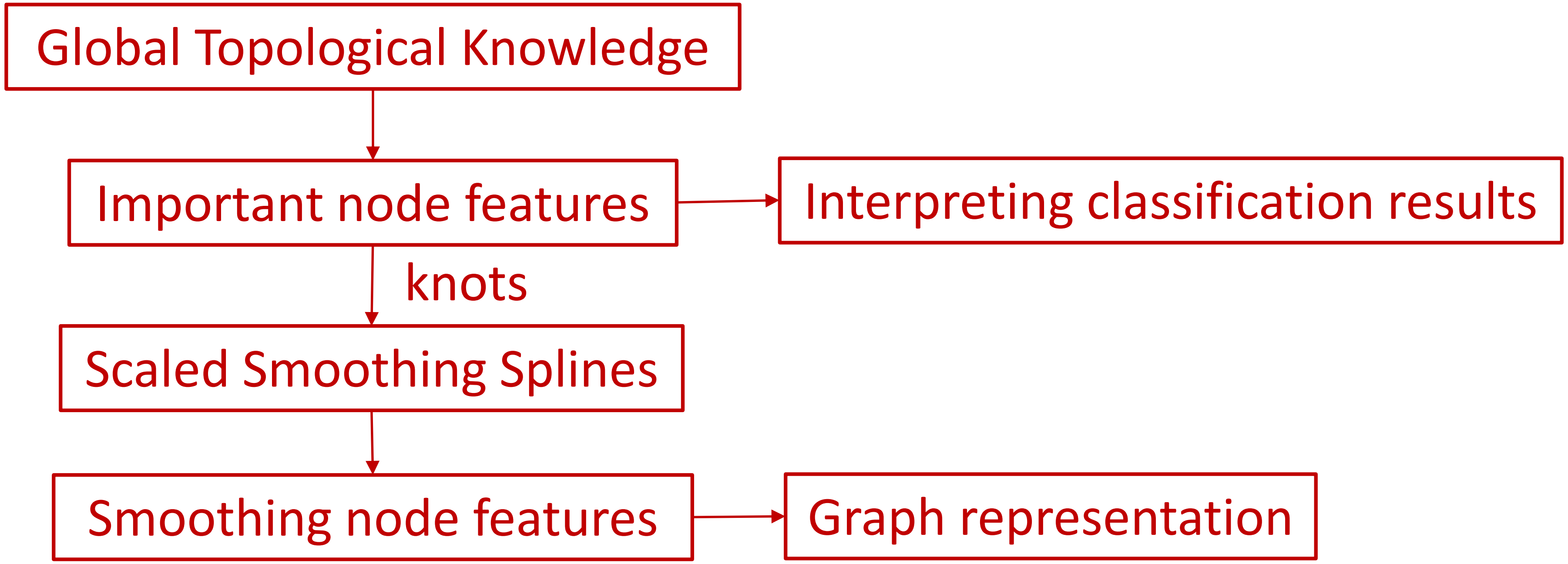
- GSSNN: Graph Smoothing Splines Neural Networks

Outline

- Introduction
  - Preliminaries
  - Motivation
- Approach
  - GSSNN: Graph Smoothing Splines Neural Networks
  - Overall Model
- Experiments
  - Settings
  - Results and Analysis
- Conclusion

GSSNN

- Roadmap
  - Non-smoothing node features → Scaled Smoothing Splines
  - Ignore global topological knowledge → Node Importance Scoring



GSSNN-Scaled Smoothing Splines (S^3)

- Smoothing Splines
  - Regression skill, aim to solve the following problem

$$\mathrm{RSS}(f,\lambda)=\sum_{i=1}^{N}\{y_i-\boxed{f(x_i)}\}^2+\lambda\int_a^b\{f''(t)\}^2\,dt, \quad (1)$$

- Generalize smoothing splines to multi-dimensional values

Consider the first layer's feature fitting function: $g_k(X_i)=X_i^T W_k^0$, where $g_k(X_i)$ denote the ith node's kth feature.
To make g_k smooth and insensitive to noisy data, we hope to minimize the following penalized residual sum of squares:

$$\mathrm{RSS}(g_k,\lambda)=\sum_{i=1}^{N}\left\{y_i-\boxed{g_k}(x_i^1,x_i^2,\dots,x_i^d)\right\}^2+\lambda\int_B\sum_{j=1}^{d}\left(\frac{\partial^2 g_k}{\partial x^{j\,2}}\right)^2 dx$$

GSSNN-Scaled Smoothing Splines (S^3)

- Generalize smoothing splines to multi-dimensional values

Theorem 1. If $g_k(x^1, x^2, \dots, x^d)$ that minimizes the RSS equation with two continuous derivatives has the form $g_k(x^1, x^2, \dots, x^d) = \sum_{j=1}^{d} u_j(x^j)$, then RSS equation has an explicit, finite-dimensional, unique minimizer:

$$g_k\left(x^1, x^2, \dots, x^d\right) = \sum_{j=1}^{d} \sum_{i=1}^{N} \boxed{\alpha_i^j\left(x^j\right)} \theta_{ij}$$

Important nodes features

Where θ_{ij} is the learnable parameter, $\alpha_i^j(x^j)$ can be represented by the natural cubic spline with N knots ξ_k, and x_i^j is the value of the jth feature of node v_i, $l_{j,k}$ are the node indexes that make $x_{l_{j,1}}^j < \cdots < x_{l_{j,N}}^j$.

$$\alpha_1^j(x^j) = 1,$$
$$\alpha_2^j(x^j) = x^j,$$
$$\alpha_{k+2}^j(x^j) = \boxed{d_k^j}(x^j) - d_{N-1}^j(x^j),$$

$$d_k^j(x^j) = \frac{\left(x^j - \xi_k^j\right)_+^3 - \left(x^j - \xi_N^j\right)_+^3}{\xi_N^j - \xi_k^j},$$

$$\xi_k^j = x_{l_{j,k}}^j \in \mathbb{R} \text{ and } a_j < x_{l_{j,1}}^j < \dots < x_{l_{j,N}}^j < b_j$$

GSSNN-Smoothing Feature Enhancement

- Generalize smoothing splines to graph neural networks

According to Theorem 1, we design a natural cubic splines function on $(x^1, x^2, \ldots, x^d)$ *to make* $g_k(x)$ *minimize the RSS equation.*

$$F_s(X) = \sigma([\beta(X_1^T), \beta(X_2^T), \ldots, \beta(X_N^T)]^T W_s + b_s)$$

$$\beta(x^1, x^2, \ldots, x^d) = (\gamma(x^1), \gamma(x^2), \ldots, \gamma(x^d))$$

$$\gamma(x^j) = (\alpha_1^j(x^j), \alpha_2^j(x^j), \ldots, \alpha_K^j(x^j)),$$

Where the K *is the number of knots for one feature dimension, and* W_s *and* b_s *are learnable parameters for scaling the expanded nodes dimension.*
Apply scaled smoothing splines after single layer of GNN as follows.

$$f_1(X, \hat{A}) = \sigma\left(\hat{A} F_s(X) W^{(0)}\right)$$

GSSNN - Node Importance Scoring

- Incorporate Global Topological Knowledge

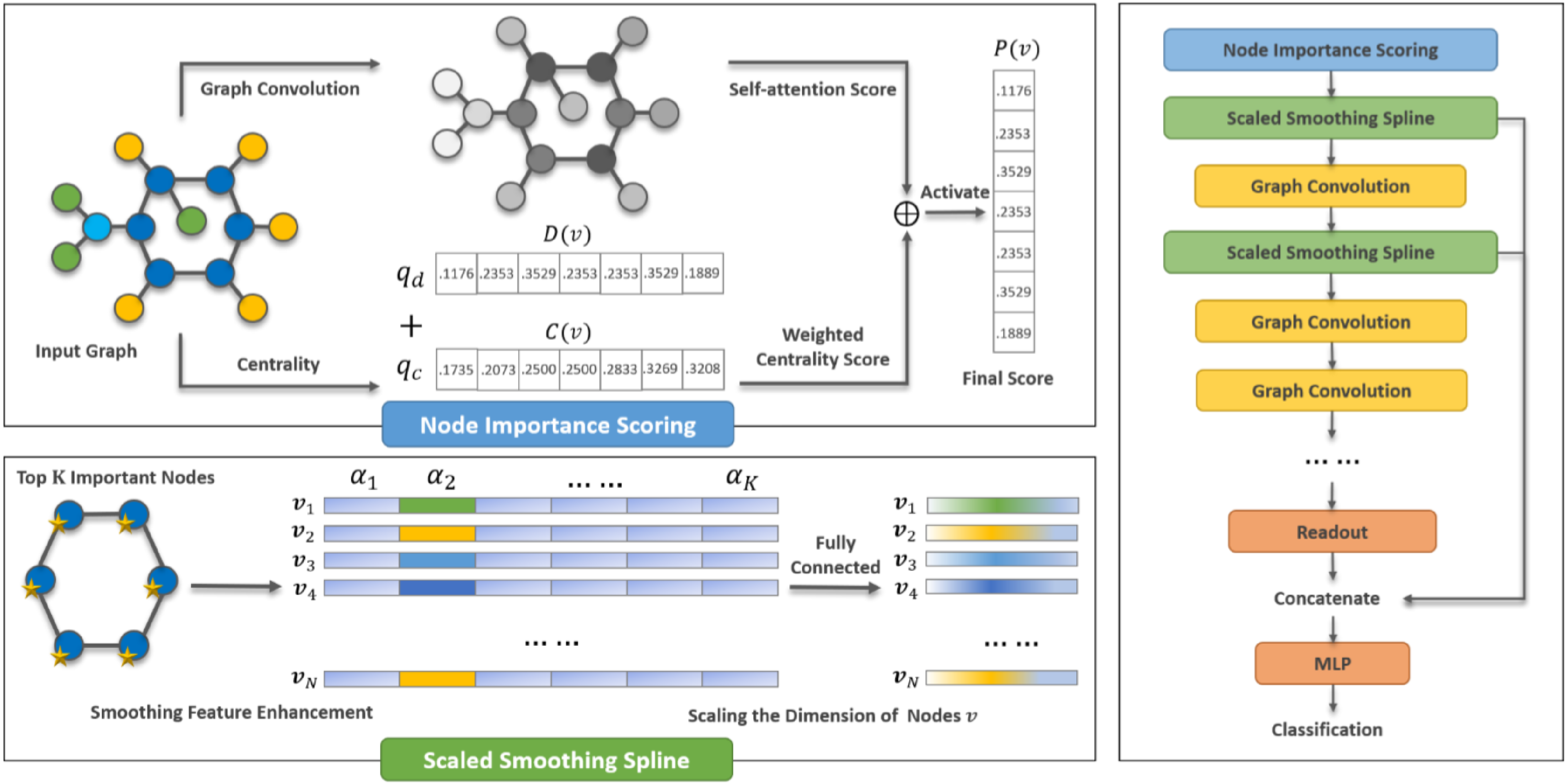
- *Self-attention Scoring* $S = \sigma\left(\hat{A}\sigma\left(\hat{A}XW^{(0)}\right)W^{(1)}\right)$ *[local]*
  - $W^{(0)} \in \mathbb{R}^{d\times d}, W^{(1)} \in \mathbb{R}^{d\times 1}$ *are learnable parameters.*

- *Centrality Scoring*
  - *Degree centrality* $D(v)$ *[local]*
  - *Closeness centrality* $C(v)$ *[global]*

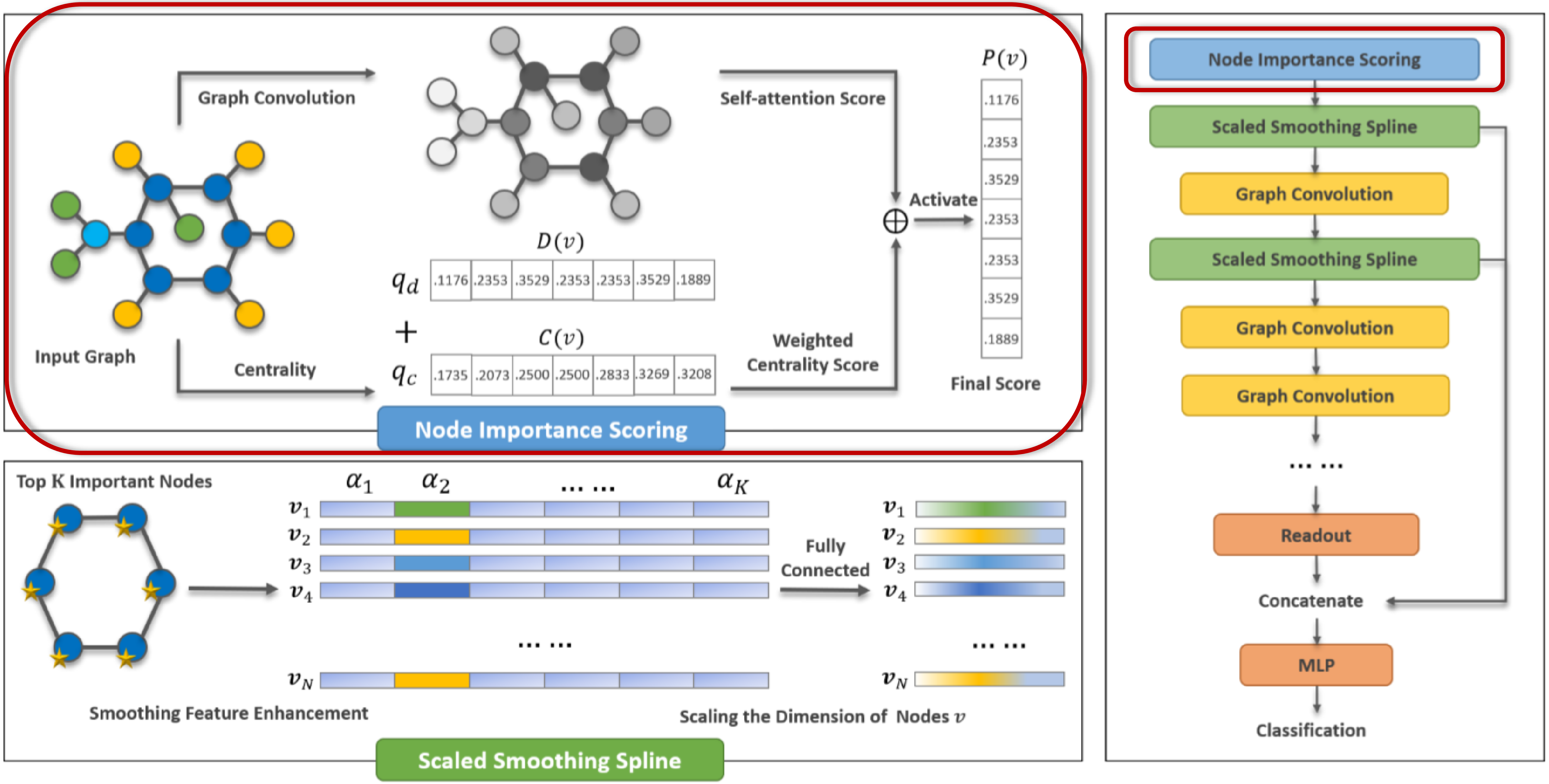
- *Final importance scores*
  - *Weighted sum of above scores:* $P(v) = \sigma(q_s S_v + q_d D(v) + q_c C(v))$

According to importance scores, select the top-K important nodes features as knots in scaled smoothing splines.

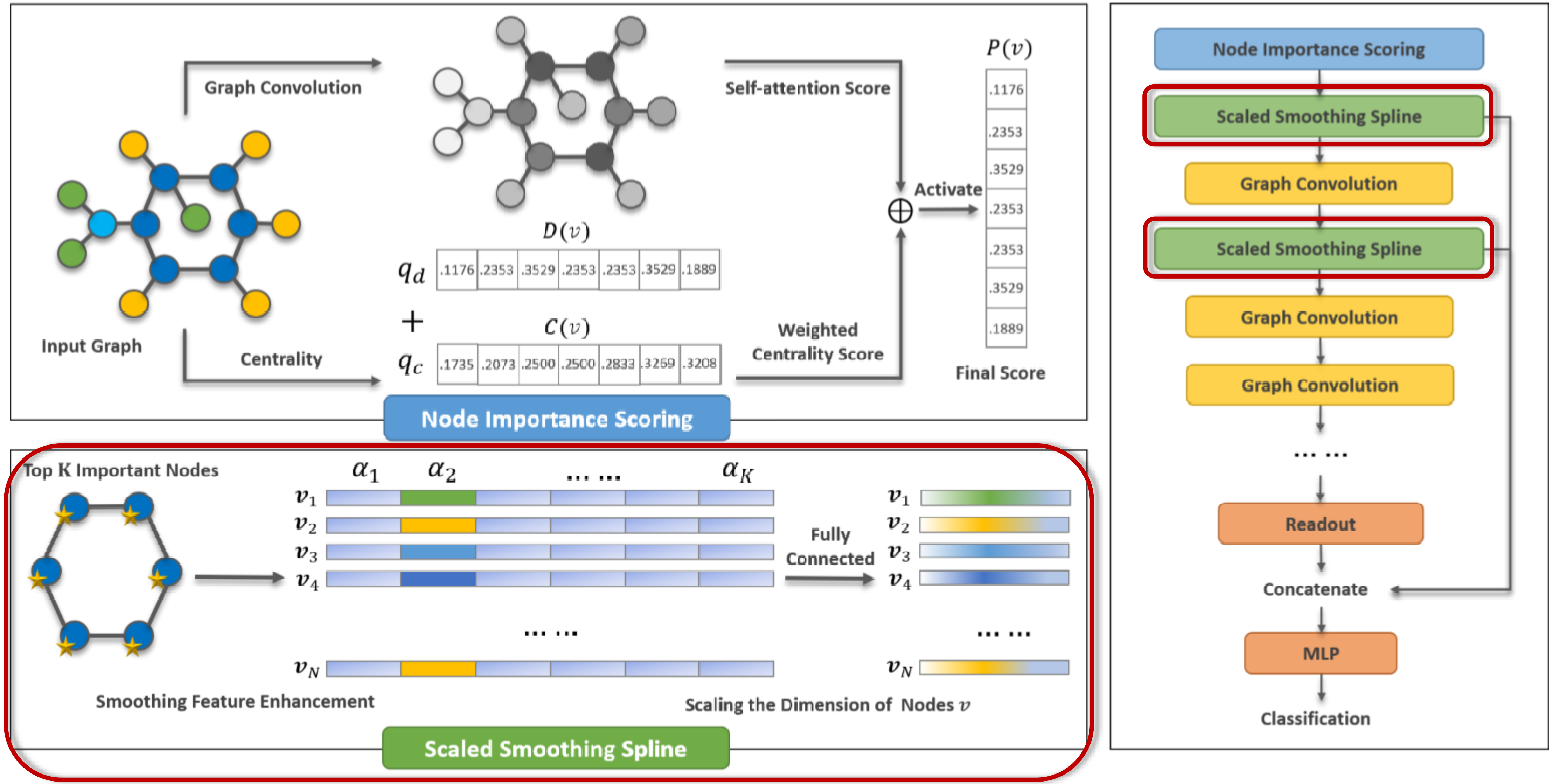
GSSNN-Architecture



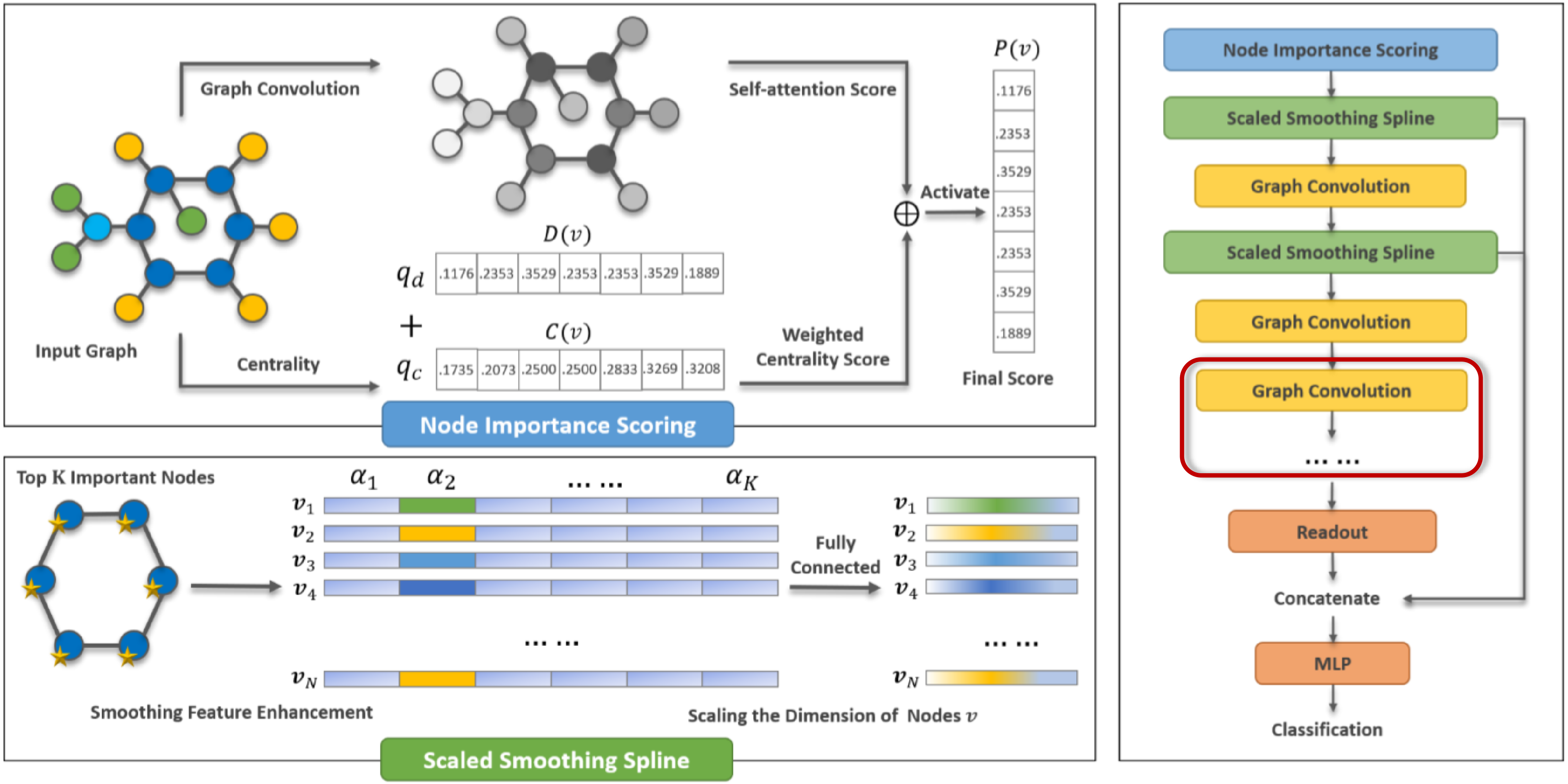
GSSNN-Architecture



GSSNN-Architecture



GSSNN-Architecture



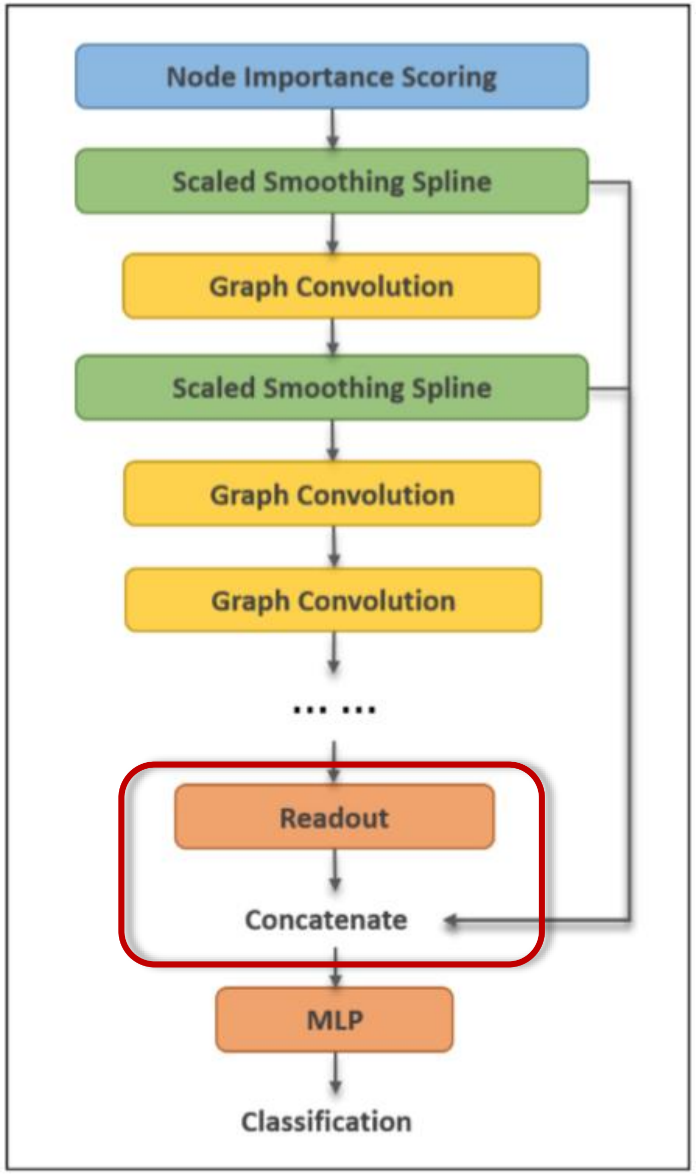
GSSNN-Architecture

- Readout Layer

$$h_G = \mathrm{CONCAT}\left\{\mathrm{SUM}\left\{h_v | v \in G\right\}, p\left(\xi_i | i = 1, ..., K\right)\right\}$$

- Model Training
  - Feed the graph embedding to MLP
  - Minimizing the cross-entropy loss over labeled training examples:

$$\mathcal{L} = -\sum_{l \in \mathcal{Y}_L} \sum_{f=1}^{F} Y_{lf} \ln X_{lf}$$



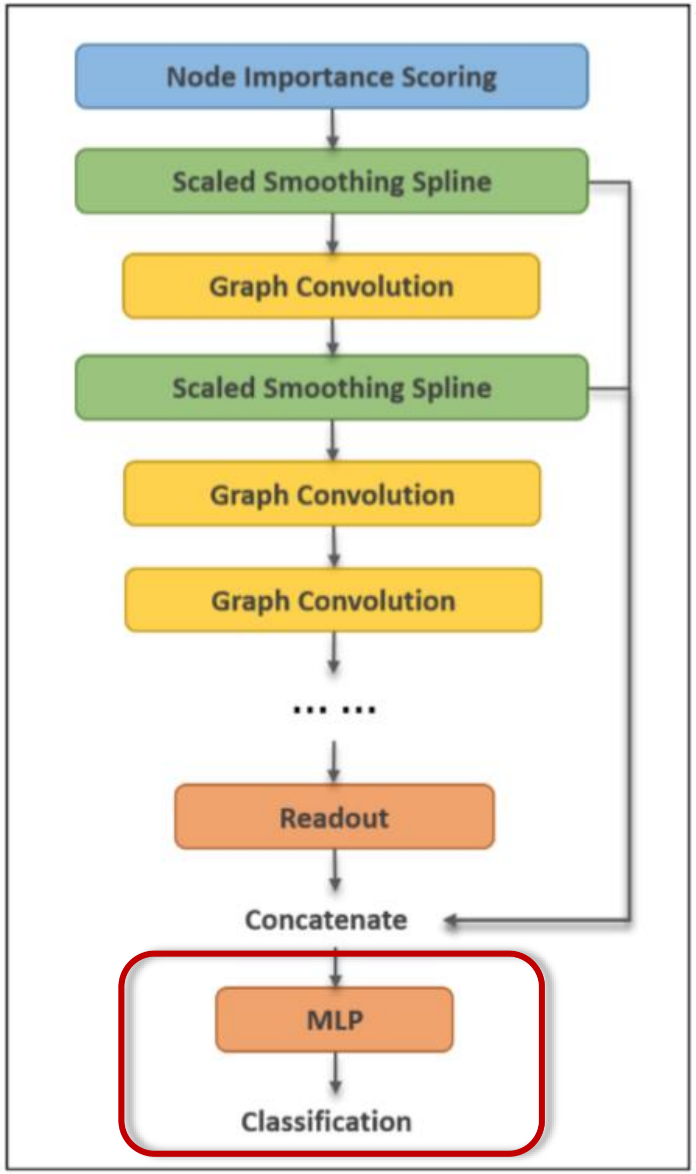
GSSNN-Architecture

- Readout Layer

$$h_G = \mathrm{CONCAT}\left\{\mathrm{SUM}\left\{h_v | v \in G\right\}, p\left(\xi_i | i = 1, ..., K\right)\right\}$$

- Model Training
  - Feed the graph embedding to MLP
  - Minimizing the cross-entropy loss over labeled training examples:

$$\mathcal{L} = -\sum_{l \in \mathcal{Y}_L} \sum_{f=1}^{F} Y_{lf} \ln X_{lf}$$



Outline

- Introduction
  - Preliminaries
  - Motivation
- Approach
  - GSSNN: Graph Smoothing Splines Neural Networks
  - Overall Model
- Experiments
  - Settings
  - Results and Analysis
- Conclusion

Experiment Settings

- Datasets
  - Biological datasets
  - Social datasets

Dataset	Source	Graphs	Classes	Avg.N	Avg.E
MUTAG	Bio	188	2	17.93	19.79
PROTEINS	Bio	1113	2	39.06	72.81
D&D	Bio	1178	2	284.31	715.65
NCI1	Bio	4110	2	29.87	32.30
IMDB-B	Social	1000	2	19.77	193.06
IMDB-M	Social	1500	3	13	131.87
COLLAB	Social	5000	3	74.49	4914.99

- Baselines
  - Kernel-based methods: WL, GK, DGK
  - GNN-based methods:
    - GCAPS-CNN, GapsGNN, GIN
    - SortPool, DiffPool, SAGPool

Results

- Graph Classification

Table 4: Graph classification results of biological and social datasets in accuracy.

	Method	MUTAG	NCI1	PROTEINS	DD	COLLAB	IMDB-B	IMDB-M
Kernel	WL	82.05±0.36	**82.19±0.18**	74.68±0.49	79.78±0.36	79.02±1.77	73.40±4.63	49.33±4.75
	GK	81.58±2.11	62.49±0.27	71.67±0.55	78.45±0.26	72.84±0.28	65.87±0.98	43.89±0.38
	DGK	87.44±2.72	80.31±0.46	75.68±0.54	73.50±1.01	73.09±0.25	66.96±0.56	44.55±0.52
GNN	GCAPS-CNN	89.62±5.38	81.35±2.37	75.70±3.86	78.82±3.17	77.32±1.98	72.02±4.10	49.31±5.30
	GapsGNN	87.78±6.68	78.25±2.22	75.68±3.22	75.88±3.41	79.67±1.24	74.68±3.10	52.17±4.25
	GIN	93.50±6.49	80.85±2.34	76.81±3.78	77.76±2.27	80.50±1.43	78.60±3.37	54.33±4.49
	SortPool	86.62±4.72	70.36±4.36	76.72±3.77	75.27±2.60	78.70±1.52	74.40±5.29	53.07±5.20
	DiffPool	89.79±8.15	78.29±3.33	77.02±3.23	70.95±2.41	79.70±1.84	78.08±4.24	53.13±4.70
	SAGPool	90.42±7.78	77.62±2.37	76.55±3.50	76.91±2.12	79.88±1.02	78.10±4.20	53.80±4.08
	GSSNN	**96.77±4.68**	80.75±4.07	**79.73±3.31**	**80.26±2.50**	**81.60±1.26**	**80.10±3.25**	**59.00±3.80**
		3.27	1.44	2.92	0.48	1.10	1.50	4.67

- GSSNN achieves the best performance on six datasets.

Results

- Global Information

Table 5: Graph classification accuracy with different scoring strategies.

Method	S_v	$S_v + D(v)$	$S_v + D(v) + C(v)$
MUTAG	88.89	94.44	96.77
DD	74.62	77.78	80.26
IMDB-B	77.30	79.30	80.10

- S_v : Self-attention scores [local]
- $S_v + D(v)$: self-attention score plus degree scores [local]
- $S_v + D(v) + C(v)$: self-attention score plus degree scores and closeness scores [global]

Results

- Scaled Smoothing Splines (S^3) as a Plugin

Table 6: Graph classification results of existing GNNs plugged with S^3 in accuracy.

Method	MUTAG	PROTEINS	DD	IMDB-M
GCN	93.50	76.81	77.76	54.33
GCN+S^3	96.77	79.73	80.26	59.00
GAT	95.33	77.48	77.78	55.33
GAT+S^3	96.89	80.18	81.20	56.67

- The effectiveness of scaled smoothing splines (S^3)

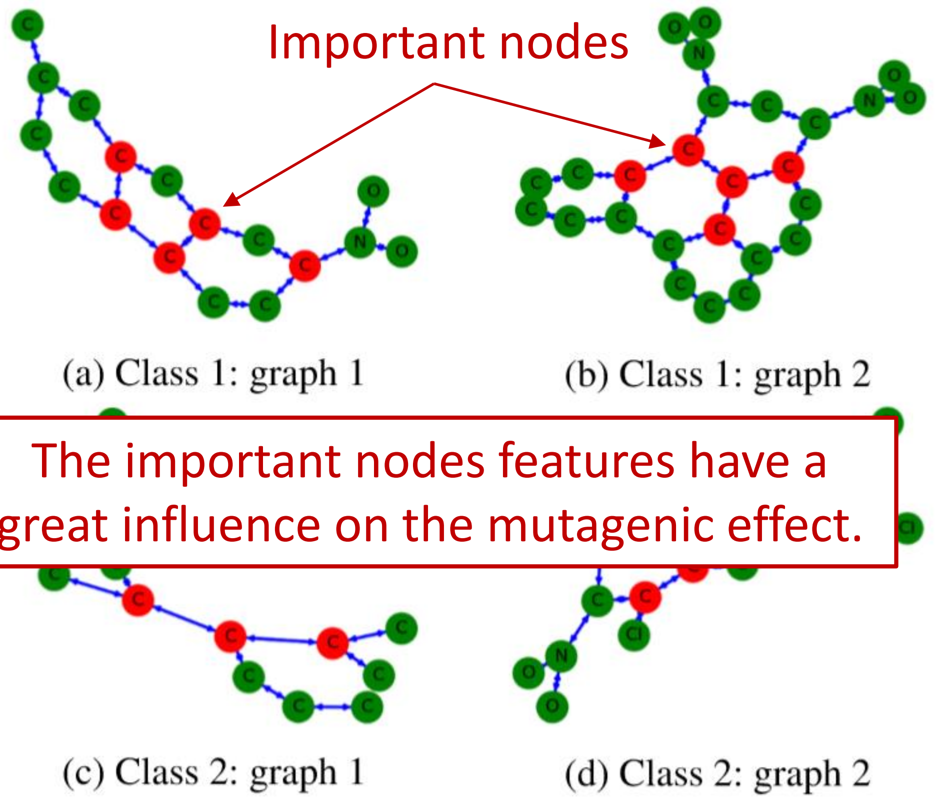
Results

- Interpretability
  - Important nodes are mainly focused on heavy atoms with large degree, which determine the structure and properties of the compound to a large extent.

The important nodes or substructures

affect

Graph classification results



The important nodes features have a great influence on the mutagenic effect.

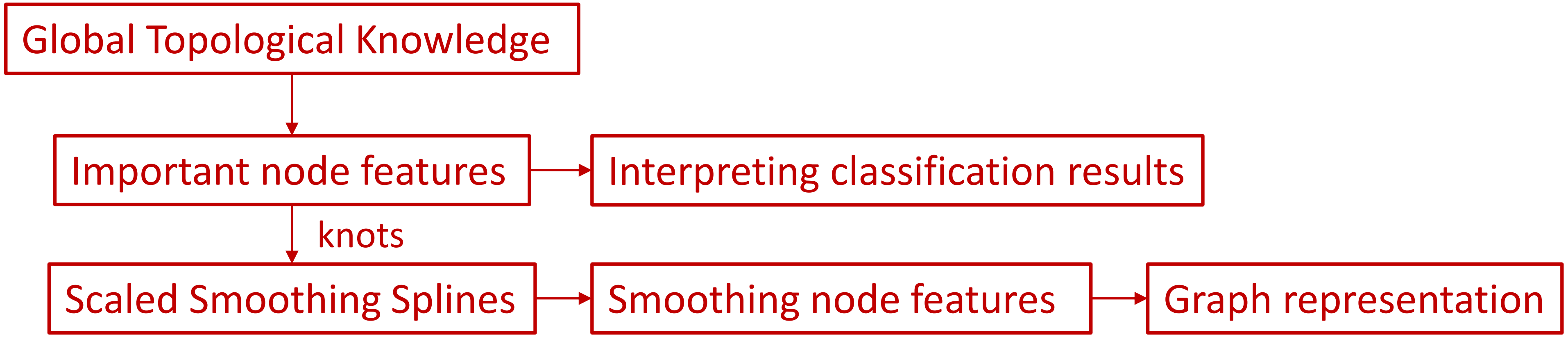
Visualization of important nodes in MUTAG dataset

Outline

- Introduction
  - Preliminaries
  - Motivation
- Approach
  - GSSNN: Graph Smoothing Splines Neural Networks
  - Overall Model
- Experiments
  - Settings
  - Results and Analysis
- Conclusion

Conclusion

- GSSNN
  - End-to-end model for graph-level representation learning: smoothing node features + global topological knowledge → high-quality and more robust graph features
  - Scaled smoothing splines: easily fit into existing GNNs
  - Interpretability



Thanks!

Q&A

zhushichao@iie.ac.cn

References

- [1] Shervashidze, N.; Schweitzer, P.; Leeuwen, E. J. v.; Mehlhorn, K.; and Borgwardt, K. M. 2011. Weisfeilerlehman graph kernels. Journal of Machine Learning Research 12(Sep):2539–2561.
- [2] Shervashidze,N.;Vishwanathan,S.;Petri,T.;Mehlhorn,K.; andBorgwardt,K. 2009. Efficient graphlet kernels for large graph comparison. In Artificial Intelligence and Statistics, 488–495.
- [3] Vishwanathan,S.V.N.;Schraudolph,N.N.;Kondor,R.;and Borgwardt,K.M. 2010. Graph kernels. Journal of Machine Learning Research 11(Apr):1201–1242.
- [4] Verma, S., and Zhang, Z.-L. 2019. Graph capsule convolutional neural networks. In Proceedings of the 7th International Conference on Learning Representations.
- [5] Xinyi, Z., and Chen, L. 2019. Capsule graph neural network. In Proceedings of the 7th International Conference on Learning Representations.
- [6] Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How powerful are graph neural networks? In Proceedings of the 7th International Conference on Learning Representations.

References

- [7] Vinyals, O.; Bengio, S.; and Kudlur, M. 2015. Order matters: Sequence to sequence for sets.
- [8] Zhang, M.; Cui, Z.; Neumann, M.; and Chen, Y. 2018. An end-to-end deep learning architecture for graph classification. In Proceedings of the 32th AAAI Conference on Artificial Intelligence.
- [9] Ying, Z.; You, J.; Morris, C.; Ren, X.; Hamilton, W.; and Leskovec, J. 2018. Hierarchical graph representation learning with differentiable pooling. In Advances in Neural Information Processing Systems, 4800–4810.
- [10] Lee,J.;Lee,I.;andKang,J. 2019. Self-attentiongraphpooling. InProceedings of the 36th International Conferenceon Machine Learning.