ICASSP 2022 - Singapore

Graph Convolutional Neural Networks With Autoencoder-Based Compression and Multi Layer Graph Learning

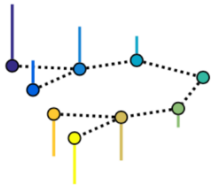
Lorenzo Giusti, Claudio Battiloro, Paolo Di Lorenzo, Sergio Barbarossa

SAPIENZA
UNIVERSITÀ DI ROMA

SAPIENZA
UNIVERSITÀ DI ROMA

Outline

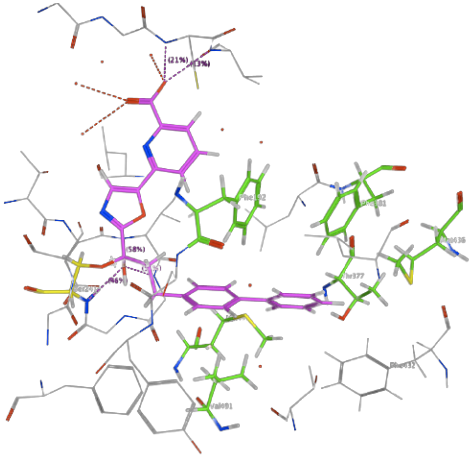
- Introduction and Research purposes
- Mathematical Background
- Autoencoder-Aided Graph Convolutional Network
- Joint Training and Multi-Layer Representation Learning
- Experimental Results
- Conclusion and Future Developments

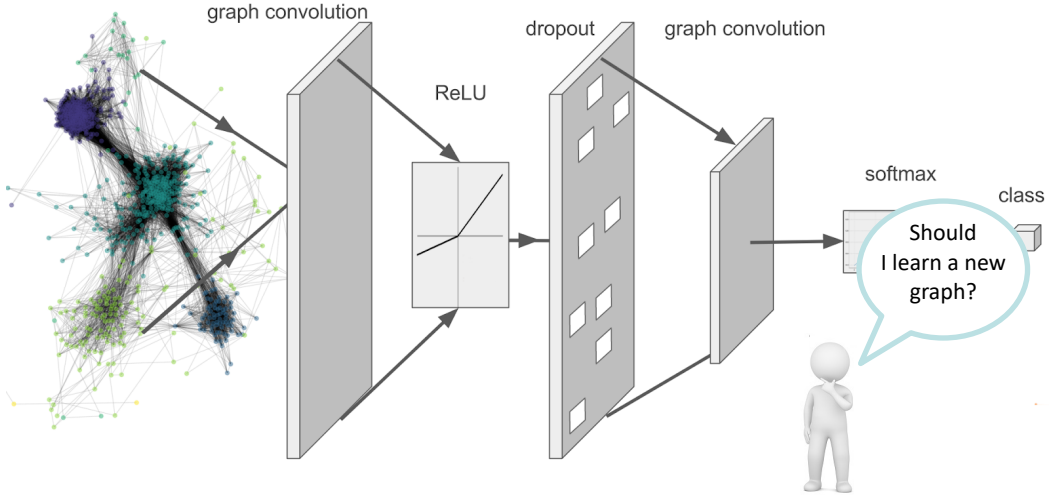


SAPIENZA
UNIVERSITÀ DI ROMA

Graph Deep Learning: Dealing with lots of "edges"

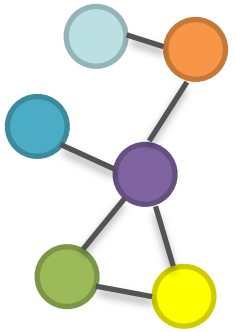
- Non-trivial applications have a graph structure which accounts millions or even billions of nodes
- Dealing with huge graphs can be computationally unfeasible if the common geometric deep learning techniques
- The more you go in depth within the network the more the initial graph structure cannot encode well the pairwise relationships



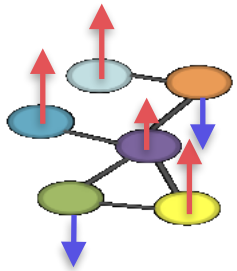


Mathematical Background

- Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a weighted undirected graph
  - $\mathcal{V} = \{1, \ldots, N\}$ is the set of vertices
  - $\mathcal{E} = \{a_{ij}\}_{i,j \in \mathcal{V}}$ is the set of weighted edges



- Let $\mathbf{A} = \{a_{i,j}\},\ i, j = 1, \ldots, N$ be the adjacency matrix of $\mathcal{G}$
- Let $\mathbf{L} = \text{diag}(\mathbf{1}^T \mathbf{A}) - \mathbf{A}$ be the Laplacian matrix of $\mathcal{G}$
  - $\text{diag}(\mathbf{x})$ is a matrix having $\mathbf{x}$ as main diagonal, and zeros elsewhere
- A graph signal x is a mapping $x : \mathcal{V} \rightarrow \mathbb{R}$
- Graph data are collections of graph signals $\mathbf{X} = \{\mathbf{x}^f\}_{f=1}^{F} \in \mathbb{R}^{N \times F}$



SAPIENZA
UNIVERSITÀ DI ROMA

Mathematical Background

- Linear Shift Invariant Graph Filter:

$$\mathbf{y} = \sum_{k=0}^{K-1} h_k \mathbf{S}^k \mathbf{x} \quad [1]$$

[1] D. Shuman et al., “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains,” *IEEE Signal Processing Magazine*, vol. 30, 10 2012.

Mathematical Background

- Linear Shift Invariant Graph Filter:

$$\mathbf{y} = \sum_{k=0}^{K-1} h_k \mathbf{S}^k \mathbf{x}$$ [1]

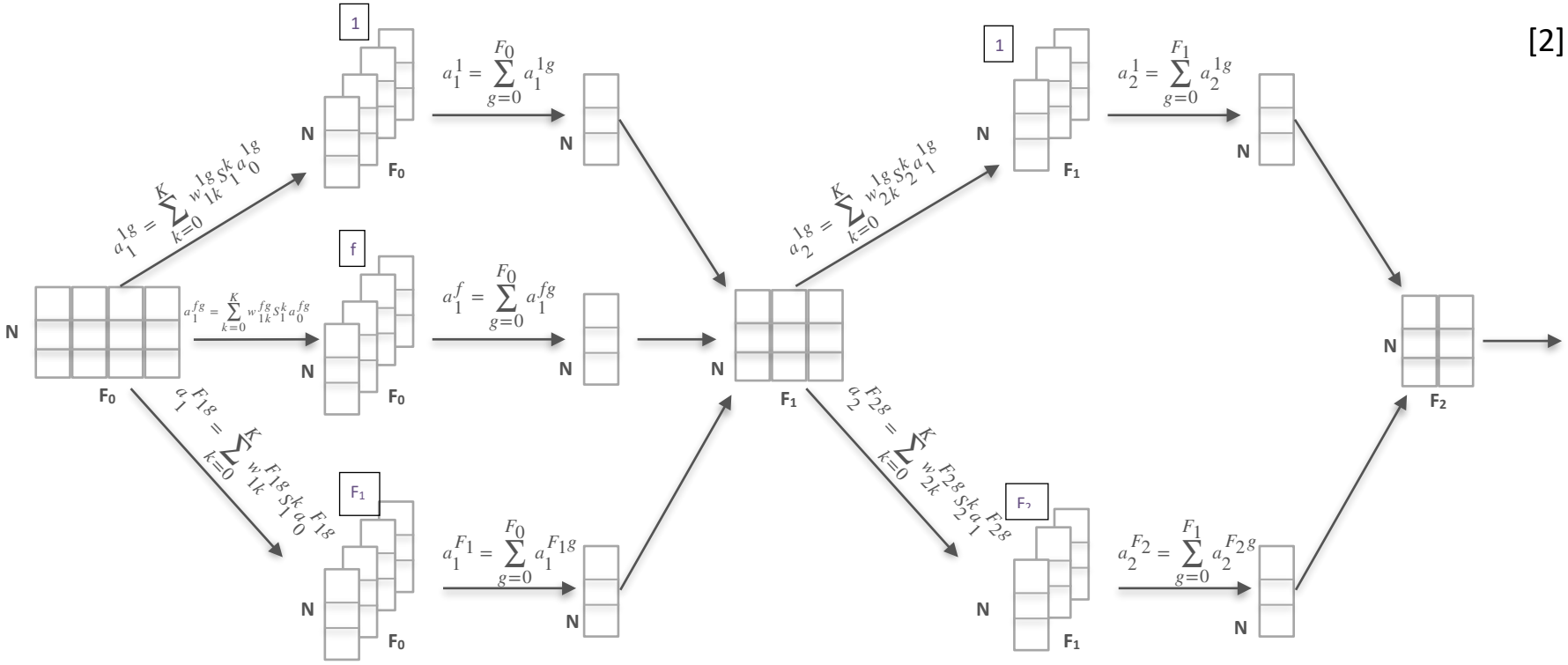
- The $\ell - th$ layer of a state of the art GCN can be summarised as:

$$\tilde{\mathbf{z}}_l^g = \sigma_l\left(\sum_{f=1}^{F_{l-1}} \sum_{k=0}^{K_l-1} \mathbf{h}_{lk}^{fg} \mathbf{S}^k \tilde{\mathbf{z}}_{l-1}^f \right), \quad g = 1,\ldots,F_l .$$

[1] D. Shuman et al., "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, 10 2012.

SAPIENZA
UNIVERSITÀ DI ROMA

State of the Art Architecture



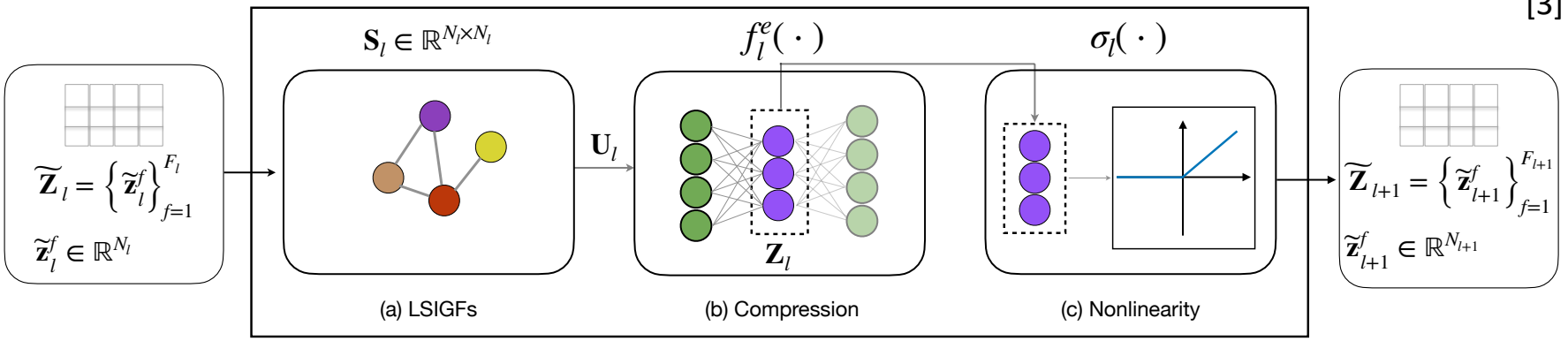
- This architecture uses the same graph for all layers!

[2] Fernando Gama et al., Graphs, Convolutions, and Neural Networks, 2021, arXiv:2003.03777.

SAPIENZA
UNIVERSITÀ DI ROMA

Autoencoder-Aided Graph Convolutional Network

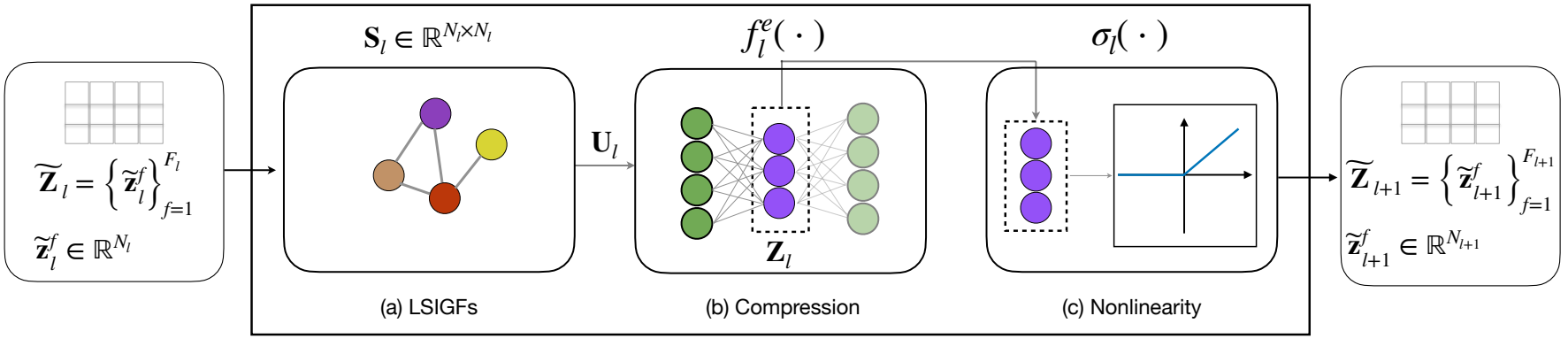
[3]



- Each layer is composed of three main stages:
  - Filtering
  - Compression
  - Non-Linearity
- The idea is to have a tunable compression operation induced by the encoding function of the autoencoders

[3] Lorenzo Giusti et al. Graph Convolutional Networks With Autoencoder-Based Compression and Multi-Layer Graph Learning

Autoencoder-Aided Graph Convolutional Network



- The forward rule for the $l-th$ layer is defined as:

$$\widetilde{\mathbf{z}}_l^g = \sigma_l\left(f_l^e\left(\mathbf{u}_l^g\right)\right), \quad g = 1,\ldots,F_l$$

- Where:

$$\mathbf{u}_l^g = \sum_{f=1}^{F_{l-1}} \sum_{k=0}^{K_l-1} h_{lk}^{fg} \mathbf{S}_l^k \widetilde{\mathbf{z}}_{l-1}^f, \quad g = 1,\ldots,F_l$$

- Up to now, there is the need to learn a new graph to be used in the next layer

Joint Training of AA-GNN With Multi-Layer Graph Learning

- The architecture requires a joint training of graph filter weights, autoencoder parameters, and per-layer graph representation.
- Assuming that the shift operator $\mathbf{S}_l$ is a function of the adjacency $\mathbf{A}_l$ and letting $W = \{w_l\}_{l=1}^{L}$ be the set of all autoencoder's parameters. Then, the joint training problem reads as:

$$\min_{\{\mathbf{A}_l\}_{l=1}^{L},\mathbf{H},\mathbf{W}} \quad \mathscr{L}(\{\mathbf{A}_l\}_{l=1}^{L}, \mathbf{H}, \mathbf{W}; \{\mathbf{x}_i, \mathbf{y}_i\}_{i\in\mathscr{T}})$$

$$+\eta \sum_{l=1}^{L} \sum_{g=1}^{F_l} ||f_l^d \circ f_l^e(\mathbf{w}_l; \mathbf{u}_l^g) - \mathbf{u}_l^g||_2^2$$

$$+\beta \sum_{l=1}^{L} \mathrm{Tr}\{\widetilde{\mathbf{Z}}_l^T \mathbf{L}_l \widetilde{\mathbf{Z}}_l\} - \gamma \sum_{l=1}^{L} \mathbf{1}^T \log(\mathbf{A}_l \mathbf{1}) + \lambda \sum_{l=1}^{L} ||\mathbf{A}_l||_F^2$$

subject to

$$[\mathbf{A}_l]_{i,i} = 0, \qquad [\mathbf{A}_l]_{i,j} = [\mathbf{A}_l]_{j,i} \geq 0, \qquad \forall\, i, j, l$$

$$\mathrm{Tr}\{\mathbf{L}_l\} = d_l, \qquad L_l = diag(\mathbf{1}^T A_l) - A_l, \quad \forall\, l$$

Joint Training of AA-GNN With Multi-Layer Graph Learning

- The architecture requires a joint training of graph filter weights, autoencoder parameters, and per-layer graph representation.
- Assuming that the shift operator $\mathbf{S}_l$ is a function of the adjacency $\mathbf{A}_l$ and letting $W = \{w_l\}_{l=1}^{L}$ be the set of all autoencoder's parameters. Then, the joint training problem reads as:

$$\min_{\{\mathbf{A}_l\}_{l=1}^{L},\mathbf{H},\mathbf{W}} \mathscr{L}(\{\mathbf{A}_l\}_{l=1}^{L}, \mathbf{H}, \mathbf{W}; \{\mathbf{x}_i, \mathbf{y}_i\}_{i\in\mathscr{T}})$$

$$+\eta \sum_{l=1}^{L}\sum_{g=1}^{F_l} ||f_l^d \circ f_l^e(\mathbf{w}_l; \mathbf{u}_l^g) - \mathbf{u}_l^g||_2^2$$

$$+\beta \sum_{l=1}^{L} \mathrm{Tr}\{\widetilde{\mathbf{Z}}_l^T \mathbf{L}_l \widetilde{\mathbf{Z}}_l\} - \gamma \sum_{l=1}^{L} \mathbf{1}^T \log(\mathbf{A}_l\mathbf{1}) + \lambda \sum_{l=1}^{L} ||\mathbf{A}_l||_F^2$$

subject to

$$[\mathbf{A}_l]_{i,i} = 0, \qquad [\mathbf{A}_l]_{i,j} = [\mathbf{A}_l]_{j,i} \geq 0, \qquad \forall\, i, j, l$$

$$\mathrm{Tr}\{\mathbf{L}_l\} = d_l, \qquad L_l = diag(\mathbf{1}^T A_l) - A_l, \quad \forall\, l$$

Joint Training of AA-GNN With Multi-Layer Graph Learning

- The architecture requires a joint training of graph filter weights, autoencoder parameters, and per-layer graph representation.
- Assuming that the shift operator $\mathbf{S}_l$ is a function of the adjacency $\mathbf{A}_l$ and letting $W = \{w_l\}_{l=1}^{L}$ be the set of all autoencoder's parameters. Then, the joint training problem reads as:

$$\min_{\{\mathbf{A}_l\}_{l=1}^{L},\mathbf{H},\mathbf{W}} \quad \mathscr{L}(\{\mathbf{A}_l\}_{l=1}^{L}, \mathbf{H}, \mathbf{W}; \{\mathbf{x}_i, \mathbf{y}_i\}_{i\in\mathscr{T}})$$

$$+\eta \sum_{l=1}^{L}\sum_{g=1}^{F_l} ||f_l^d \circ f_l^e(\mathbf{w}_l; \mathbf{u}_l^g) - \mathbf{u}_l^g||_2^2$$

$$+\beta \sum_{l=1}^{L} \mathrm{Tr}\{\widetilde{\mathbf{Z}}_l^T \mathbf{L}_l \widetilde{\mathbf{Z}}_l\} - \gamma \sum_{l=1}^{L} \mathbf{1}^T \log(\mathbf{A}_l\mathbf{1}) + \lambda \sum_{l=1}^{L} ||\mathbf{A}_l||_F^2$$

subject to

$$[\mathbf{A}_l]_{i,i} = 0, \qquad [\mathbf{A}_l]_{i,j} = [\mathbf{A}_l]_{j,i} \geq 0, \qquad \forall\, i, j, l$$

$$\mathrm{Tr}\{\mathbf{L}_l\} = d_l, \qquad L_l = diag(\mathbf{1}^T A_l) - A_l, \quad \forall\, l$$

Joint Training of AA-GNN With Multi-Layer Graph Learning

- The architecture requires a joint training of graph filter weights, autoencoder parameters, and per-layer graph representation.
- Assuming that the shift operator $\mathbf{S}_l$ is a function of the adjacency $\mathbf{A}_l$ and letting $W = \{w_l\}_{l=1}^{L}$ be the set of all autoencoder's parameters. Then, the joint training problem reads as:

$$\min_{\{\mathbf{A}_l\}_{l=1}^{L},\mathbf{H},\mathbf{W}} \quad \mathscr{L}(\{\mathbf{A}_l\}_{l=1}^{L}, \mathbf{H}, \mathbf{W}; \{\mathbf{x}_i, \mathbf{y}_i\}_{i\in\mathscr{T}})$$

$$+\eta \sum_{l=1}^{L}\sum_{g=1}^{F_l} ||f_l^d \circ f_l^e(\mathbf{w}_l; \mathbf{u}_l^g) - \mathbf{u}_l^g||_2^2$$

$$+\beta \sum_{l=1}^{L} \mathrm{Tr}\{\widetilde{\mathbf{Z}}_l^T \mathbf{L}_l \widetilde{\mathbf{Z}}_l\} - \gamma \sum_{l=1}^{L} \mathbf{1}^T \log(\mathbf{A}_l\mathbf{1}) + \lambda \sum_{l=1}^{L} ||\mathbf{A}_l||_F^2$$

subject to

$$[\mathbf{A}_l]_{i,i} = 0, \qquad [\mathbf{A}_l]_{i,j} = [\mathbf{A}_l]_{j,i} \geq 0, \qquad \forall\, i, j, l$$

$$\mathrm{Tr}\{\mathbf{L}_l\} = d_l, \qquad L_l = diag(\mathbf{1}^T A_l) - A_l, \quad \forall\, l$$

Joint Training of AA-GNN With Multi-Layer Graph Learning

- The architecture requires a joint training of graph filter weights, autoencoder parameters, and per-layer graph representation.
- Assuming that the shift operator $\mathbf{S}_l$ is a function of the adjacency $\mathbf{A}_l$ and letting $W = \{w_l\}_{l=1}^{L}$ be the set of all autoencoder's parameters. Then, the joint training problem reads as:

$$\min_{\{\mathbf{A}_l\}_{l=1}^{L},\mathbf{H},\mathbf{W}} \quad \mathscr{L}(\{\mathbf{A}_l\}_{l=1}^{L}, \mathbf{H}, \mathbf{W}; \{\mathbf{x}_i, \mathbf{y}_i\}_{i\in\mathscr{T}})$$

$$+\eta \sum_{l=1}^{L}\sum_{g=1}^{F_l} ||f_l^d \circ f_l^e(\mathbf{w}_l; \mathbf{u}_l^g) - \mathbf{u}_l^g||_2^2$$

$$+\beta \sum_{l=1}^{L} \mathrm{Tr}\{\widetilde{\mathbf{Z}}_l^T \mathbf{L}_l \widetilde{\mathbf{Z}}_l\} - \gamma \sum_{l=1}^{L} \mathbf{1}^T \log(\mathbf{A}_l\mathbf{1}) + \lambda \sum_{l=1}^{L} ||\mathbf{A}_l||_F^2$$

subject to

$$[\mathbf{A}_l]_{i,i} = 0, \qquad [\mathbf{A}_l]_{i,j} = [\mathbf{A}_l]_{j,i} \geq 0, \qquad \forall\, i, j, l$$

$$\mathrm{Tr}\{\mathbf{L}_l\} = d_l, \qquad L_l = diag(\mathbf{1}^T A_l) - A_l, \quad \forall\, l$$

Joint Training of AA-GNN With Multi-Layer Graph Learning

- The architecture requires a joint training of graph filter weights, autoencoder parameters, and per-layer graph representation.
- Assuming that the shift operator $\mathbf{S}_l$ is a function of the adjacency $\mathbf{A}_l$ and letting $W = \{w_l\}_{l=1}^{L}$ be the set of all autoencoder's parameters. Then, the joint training problem reads as:

$$\min_{\{\mathbf{A}_l\}_{l=1}^{L},\mathbf{H},\mathbf{W}} \quad \mathscr{L}(\{\mathbf{A}_l\}_{l=1}^{L}, \mathbf{H}, \mathbf{W}; \{\mathbf{x}_i, \mathbf{y}_i\}_{i\in\mathscr{T}})$$

$$+\eta \sum_{l=1}^{L}\sum_{g=1}^{F_l} ||f_l^d \circ f_l^e(\mathbf{w}_l; \mathbf{u}_l^g) - \mathbf{u}_l^g||_2^2$$

$$+\beta \sum_{l=1}^{L} \mathrm{Tr}\{\widetilde{\mathbf{Z}}_l^T \mathbf{L}_l \widetilde{\mathbf{Z}}_l\} - \gamma \sum_{l=1}^{L} \mathbf{1}^T \log(\mathbf{A}_l\mathbf{1}) + \lambda \sum_{l=1}^{L} ||\mathbf{A}_l||_F^2$$

subject to

$$[\mathbf{A}_l]_{i,i} = 0, \qquad [\mathbf{A}_l]_{i,j} = [\mathbf{A}_l]_{j,i} \geq 0, \qquad \forall\, i, j, l$$

$$\mathrm{Tr}\{\mathbf{L}_l\} = d_l, \qquad L_l = diag(\mathbf{1}^T A_l) - A_l, \quad \forall\, l$$

Joint Training of AA-GNN With Multi-Layer Graph Learning

- The architecture requires a joint training of graph filter weights, autoencoder parameters, and per-layer graph representation.
- Assuming that the shift operator $\mathbf{S}_l$ is a function of the adjacency $\mathbf{A}_l$ and letting $W = \{w_l\}_{l=1}^{L}$ be the set of all autoencoder's parameters. Then, the joint training problem reads as:

$$\min_{\{\mathbf{A}_l\}_{l=1}^{L},\mathbf{H},\mathbf{W}} \quad \mathscr{L}(\{\mathbf{A}_l\}_{l=1}^{L}, \mathbf{H}, \mathbf{W}; \{\mathbf{x}_i, \mathbf{y}_i\}_{i\in\mathscr{T}})$$

$$+\eta \sum_{l=1}^{L}\sum_{g=1}^{F_l} ||f_l^d \circ f_l^e(\mathbf{w}_l; \mathbf{u}_l^g) - \mathbf{u}_l^g||_2^2$$

$$+\beta \sum_{l=1}^{L} \mathrm{Tr}\{\widetilde{\mathbf{Z}}_l^T \mathbf{L}_l \widetilde{\mathbf{Z}}_l\} - \gamma \sum_{l=1}^{L} \mathbf{1}^T \log(\mathbf{A}_l\mathbf{1}) + \lambda \sum_{l=1}^{L} ||\mathbf{A}_l||_F^2$$

subject to

$$[\mathbf{A}_l]_{i,i} = 0, \qquad [\mathbf{A}_l]_{i,j} = [\mathbf{A}_l]_{j,i} \geq 0, \qquad \forall\, i, j, l$$

$$\mathrm{Tr}\{\mathbf{L}_l\} = d_l, \qquad L_l = diag(\mathbf{1}^T A_l) - A_l, \quad \forall\, l$$

Joint Training of AA-GNN With Multi-Layer Graph Learning: Half Vectorization

- Since the adjacency matrices are symmetric, the number of variables of the optimization problem can be greatly reduced
- Let $\boldsymbol{\alpha}_l = \text{vech}(\mathbf{A}_l) \in \mathbb{R}^{\frac{N(N+1)}{2}}$ be the half-vectorization of $\mathbf{A}_l$, obtained by vectorizing only the lower triangular part of $\mathbf{A}_l$.
- The following relations hold:

$$\text{vec}(\mathbf{A}_l) = \mathbf{M}_d \boldsymbol{\alpha}_l \quad \Longleftrightarrow \quad \mathbf{A}_l = \text{vec}^{-1}(\mathbf{M}_d \boldsymbol{\alpha}_l)$$

Joint Training of AA-GNN With Multi-Layer Graph Learning: Training Algorithm

- A stochastic gradient based optimizer is chosen
- An optimizer-dependent back-propagation step is performed at each iteration on the current estimates to update them towards a descent direction
- Finally, the graphs estimates are obtained by projecting the updated variables on the feasible set

function AA-GCN TRAINING(**Inputs**)

for $t \in [1, E]$ **do**

$$\widehat{\mathbf{H}}_{t+1} = \Delta_\mu\Big(\nabla_{\mathbf{H}}\mathcal{L}\left(\widehat{\mathbf{H}}_t; \mathcal{B}_t, \{\widehat{\boldsymbol{\alpha}}_{l,t}\}_l, \widehat{\mathbf{W}}_t\right)\Big)$$

$$\widehat{\mathbf{W}}_{t+1} = \Delta_\mu\Big(\nabla_{\mathbf{W}}\mathcal{L}\left(\widehat{\mathbf{W}}_t; \mathcal{B}_t, \{\widehat{\boldsymbol{\alpha}}_{l,t}\}_l, \widehat{\mathbf{H}}_t\right)\Big)$$

$$\widehat{\boldsymbol{\alpha}}_{l,t+1} = \Pi\Big(\Delta_\mu\Big(\nabla_{\boldsymbol{\alpha}_l}\mathcal{L}\left(\{\widehat{\boldsymbol{\alpha}}_{l,t}\}_l, \mathcal{B}_t, \widehat{\mathbf{W}}_t, \widehat{\mathbf{H}}_t\right)\Big)\Big), \ \forall l$$

return $\{\widehat{\boldsymbol{\alpha}}_l\}_l = \{\widehat{\boldsymbol{\alpha}}_{l,E}\}_l, \ \widehat{\mathbf{W}} = \widehat{\mathbf{W}}_E, \ \widehat{\mathbf{H}} = \widehat{\mathbf{H}}_E$

Experimental Results: Robustness to Compression

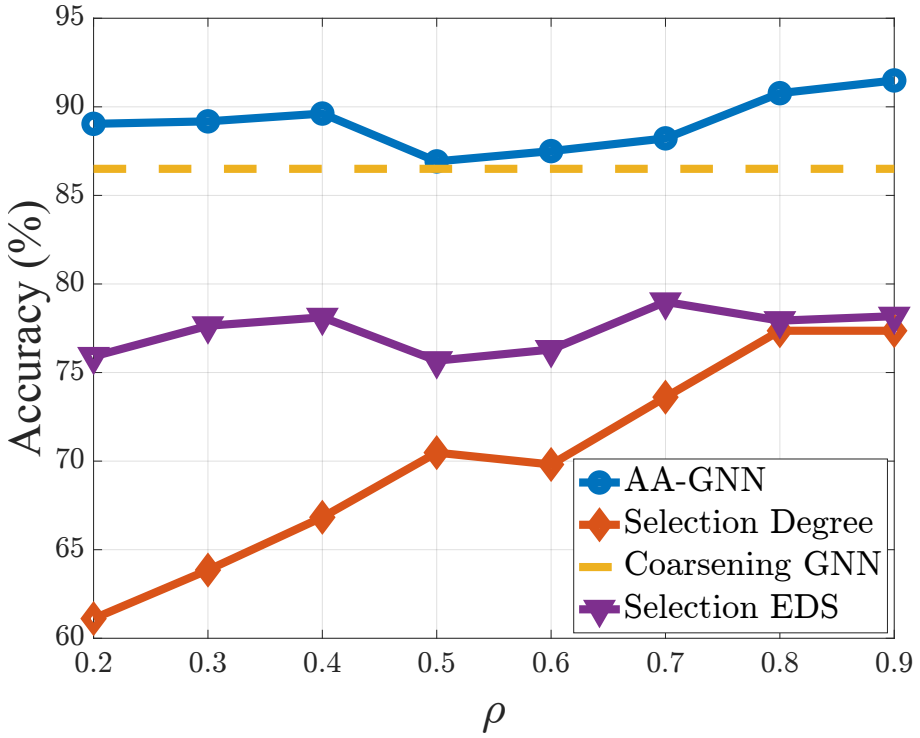
- We assessed the performance of the proposed architecture and training procedure we evaluated on the authorship attribution task

- The results show the accuracy score compared to the compression ratio:

$$\rho = \frac{N_1}{N}$$

- For a fair comparison, the second hidden layer does not provide a coarser version of the first one

- AA-GNN outperforms all the other SOTA's architectures even in a huge compression setting



Experimental Results: Robustness to AWGN

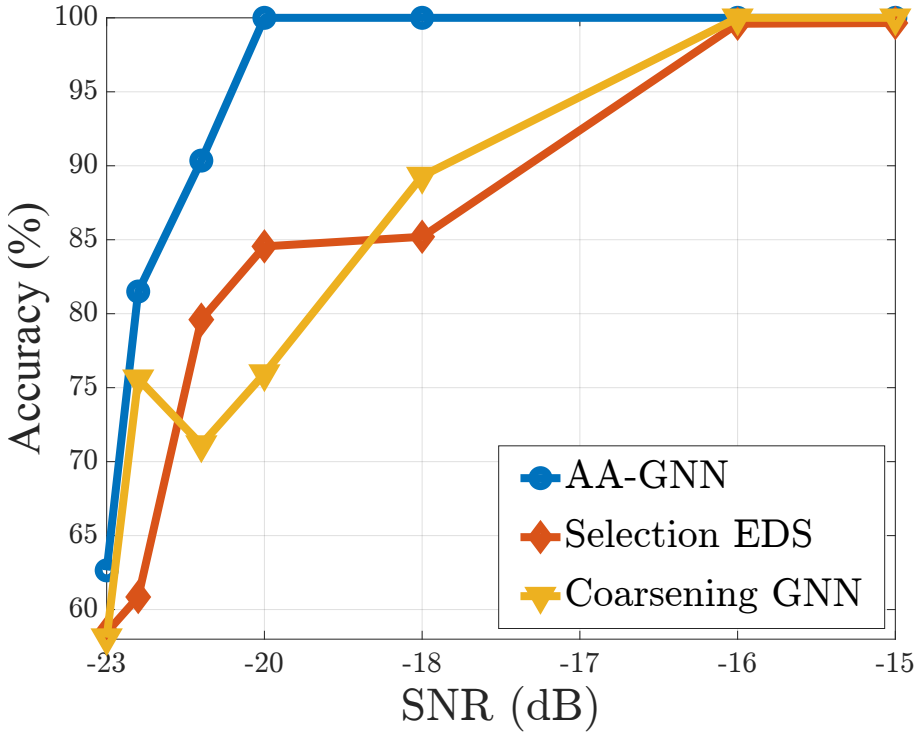
- The performance of the proposed architecture and training procedure are evaluated on the source localisation task

- The results show the accuracy score compared to the SNR of the training data:

$$\text{SNR} = 10\log_{10}\left(\frac{\sigma_{\mathcal{T}}^2}{\sigma_{\epsilon}^2}\right)$$

- $\sigma_{\mathcal{T}}^2$ is the variance of the data used for training our model

- σ_{ϵ}^2 is the variance of the AWGN



SAPIENZA
UNIVERSITÀ DI ROMA

Conclusions and Future Developments

- We have enabled tunable compression of the convolutional features, while learning different graph representations jointly with the GNN parameters
- The architecture scales well with the number of nodes of the input graph, extracting higher level representations of the convolutional features.
- Experiments illustrate the competitive performance of our architecture with respect to state of the art methods
- Future developments of this research trend include:
  - Topological Neural Networks
  - Explainability
  - Add regularisations to the autoencoders' loss