Statistical Foundations for Learning on Graphs

Thesis Defence

Aseem Baranwal, October 22, 2024

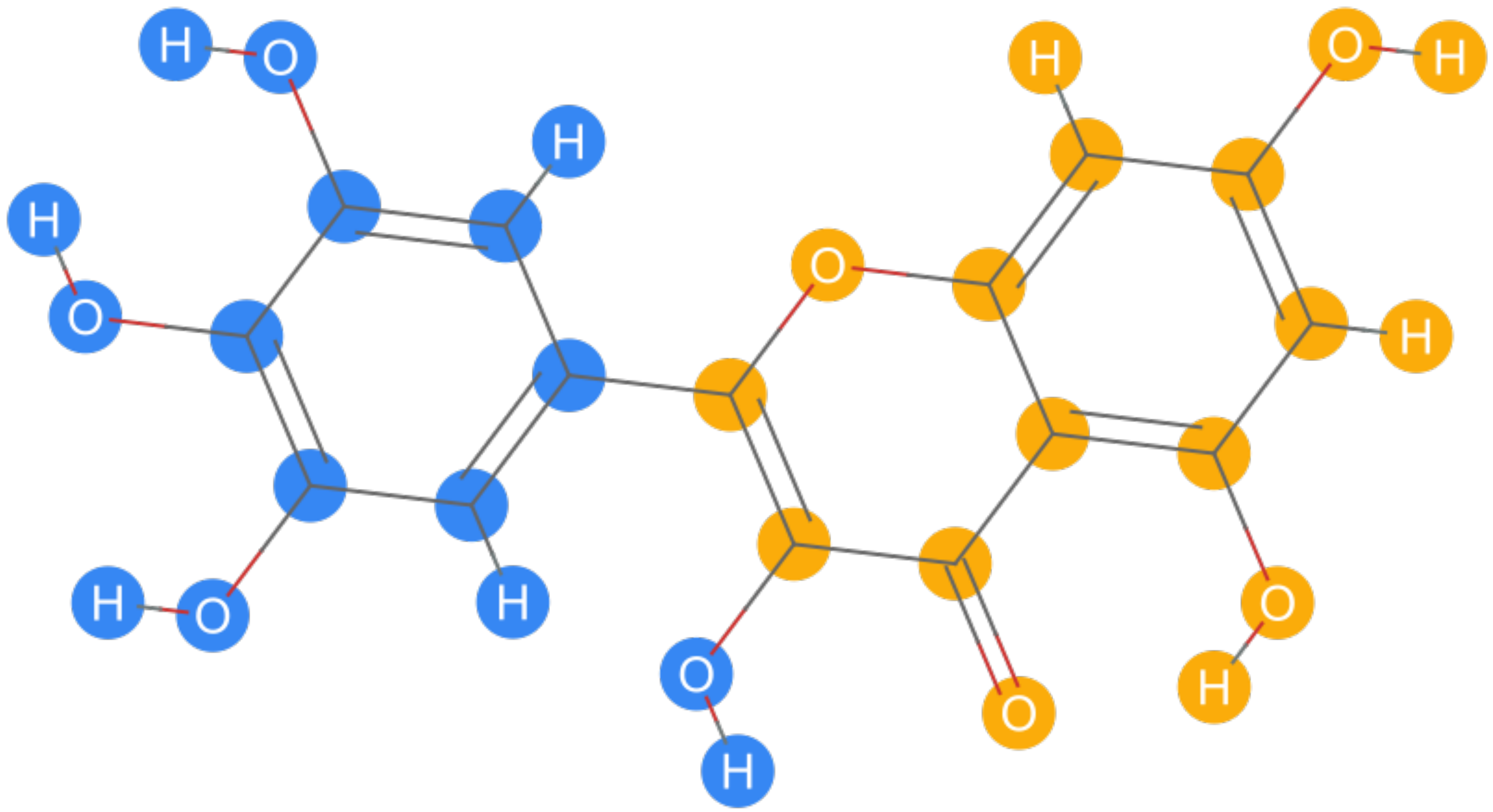
Acknowledgements

- **Supervisors**
    - Kimon Fountoulakis
    - Aukosh Jagannath
- **Committee members**
    - Xavier Bresson
    - Stephen Vavasis
    - Gautam Kamath
    - Yaoliang Yu

Acknowledgements

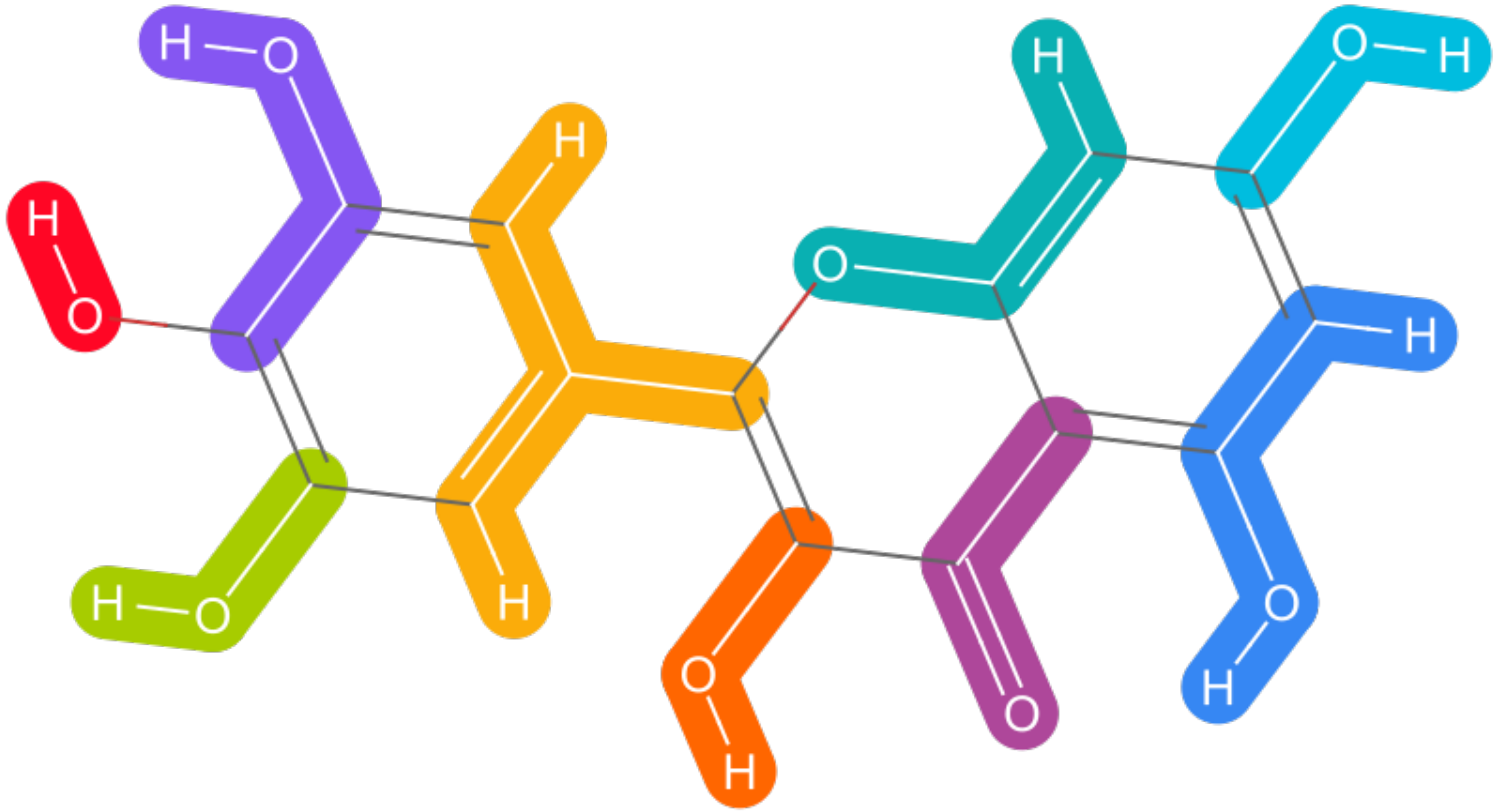
- Shenghao Yang
- Robert Wang
- Justin Ko
- Yiming Xu
- Artur de Luca
- Subhabrata Sen
- Jianqing Fan
- Marianna Pensky

Node-classification



- Molecules of compounds represented as graphs
- Atoms are nodes
- Chemical bonds are edges

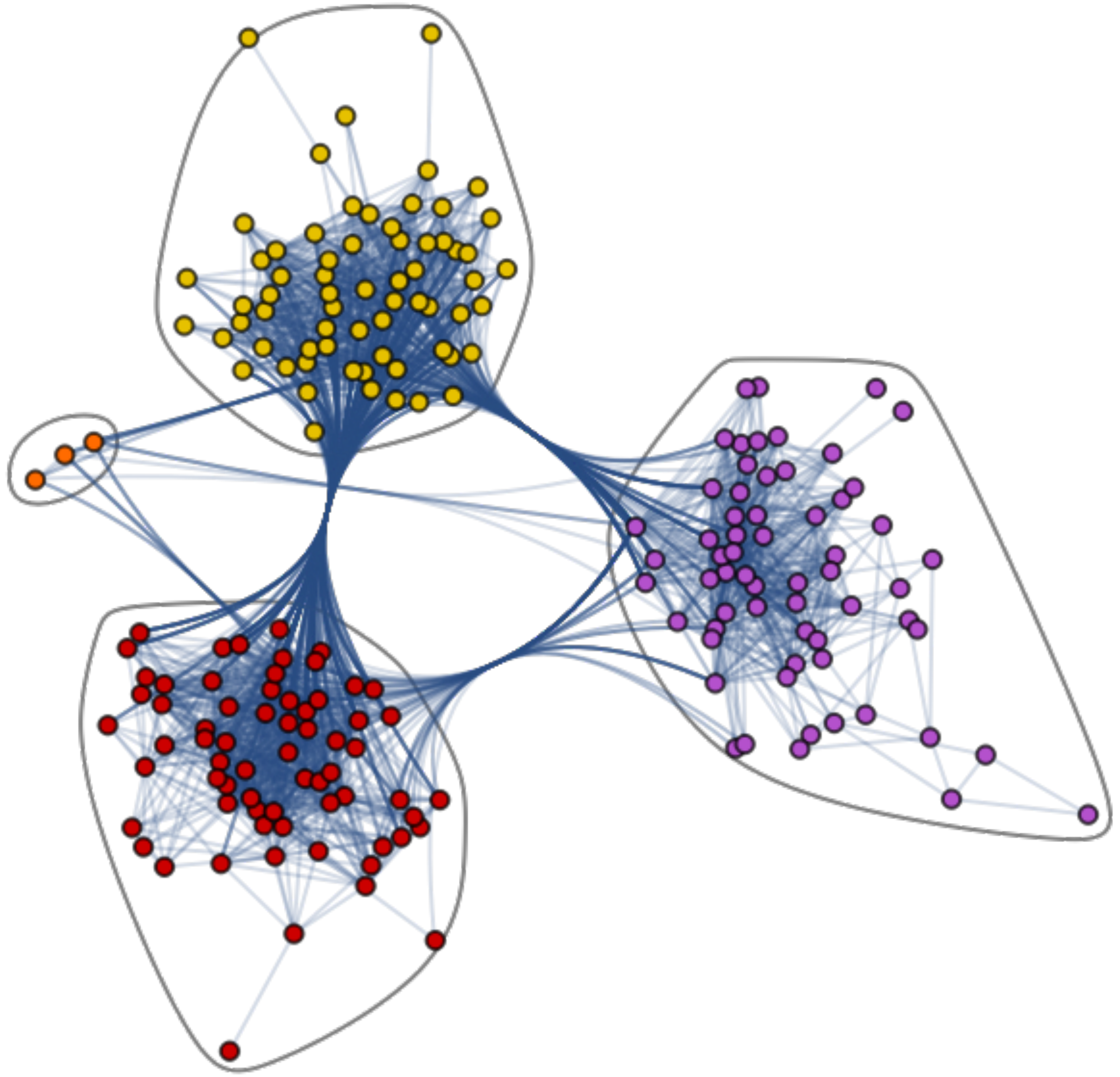
Node-classification



- Molecules of compounds represented as graphs
- Atoms are nodes
- Chemical bonds are edges

Task: Identify a functional group for each atom

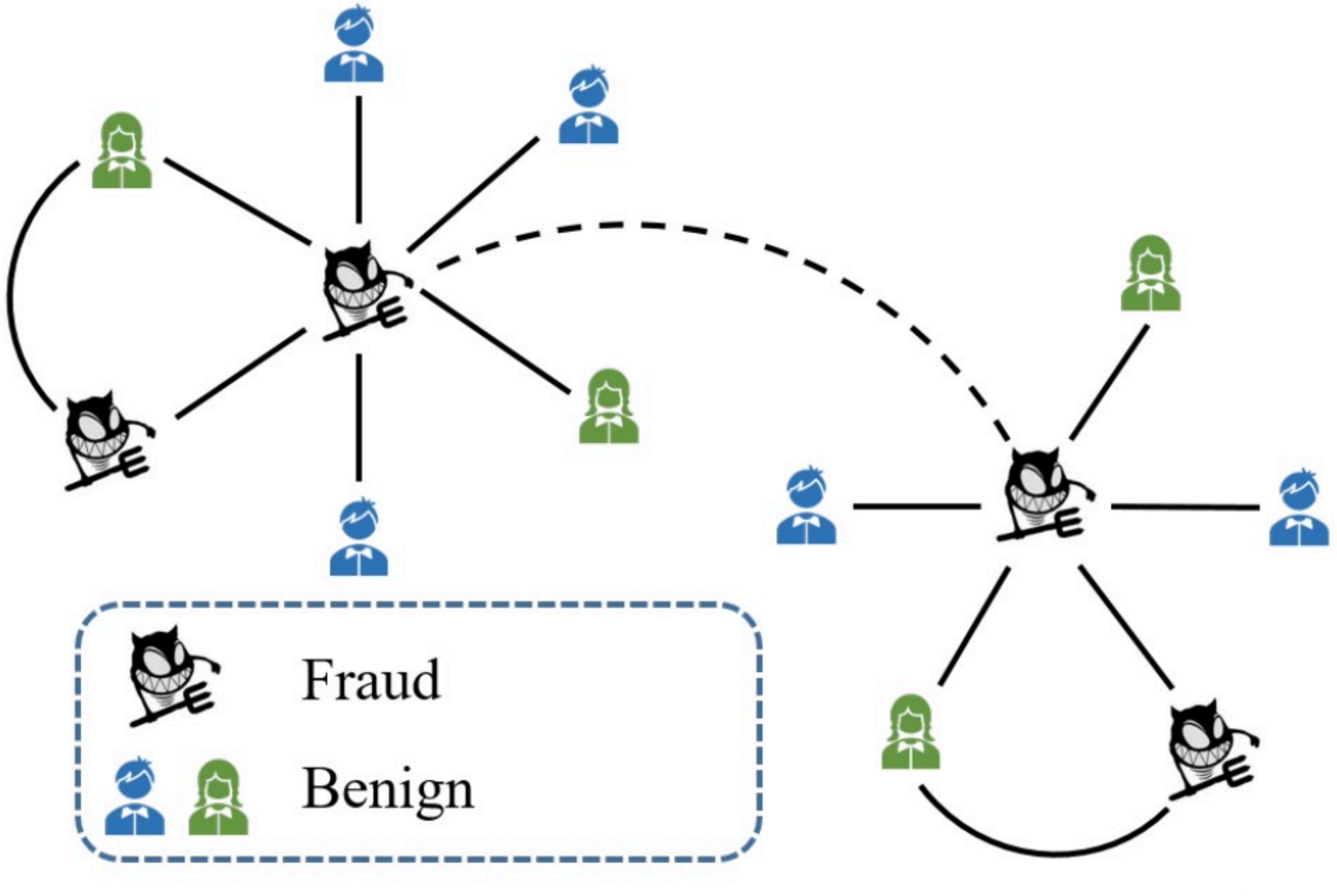
Node-classification



- Social network represented by a graph
- Individuals are nodes
- Social relationships are edges

Task: Identify social communities

Node-classification



Task: Identify fraudulent agents

Node-classification

Feature-rich relational data with n nodes:
$(A, X) \sim \mathscr{D}$ and labels y_u for $u \in [n]$

- $A \in \{0,1\}^{n \times n}$ is the adjacency matrix of the graph
- $X \in \mathbb{R}^{n \times d}$ are d-dimensional features for each node

Task: Infer the labels y_u for $u \in [n]$ given (A, X)

Statistical Data Model

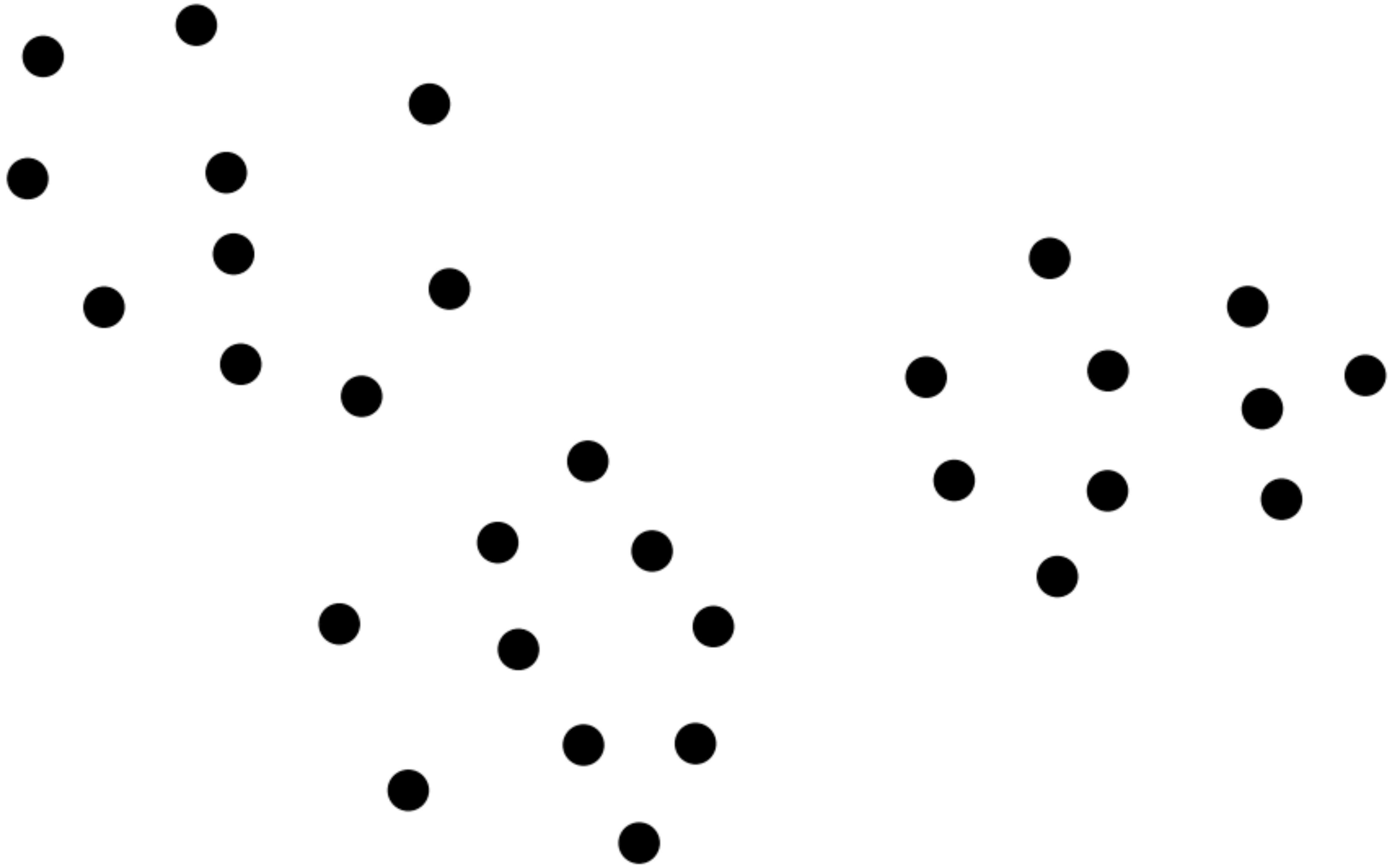
Contextual Stochastic Block Model (CSBM)

C: Number of classes

Edge-probability matrix $\mathbf{Q} = \begin{pmatrix} q_{11} & \cdots\cdots & q_{1C} \\ \vdots & \ddots & \vdots \\ q_{C1} & \cdots\cdots & q_{CC} \end{pmatrix} \in [0,1]^{C \times C}$

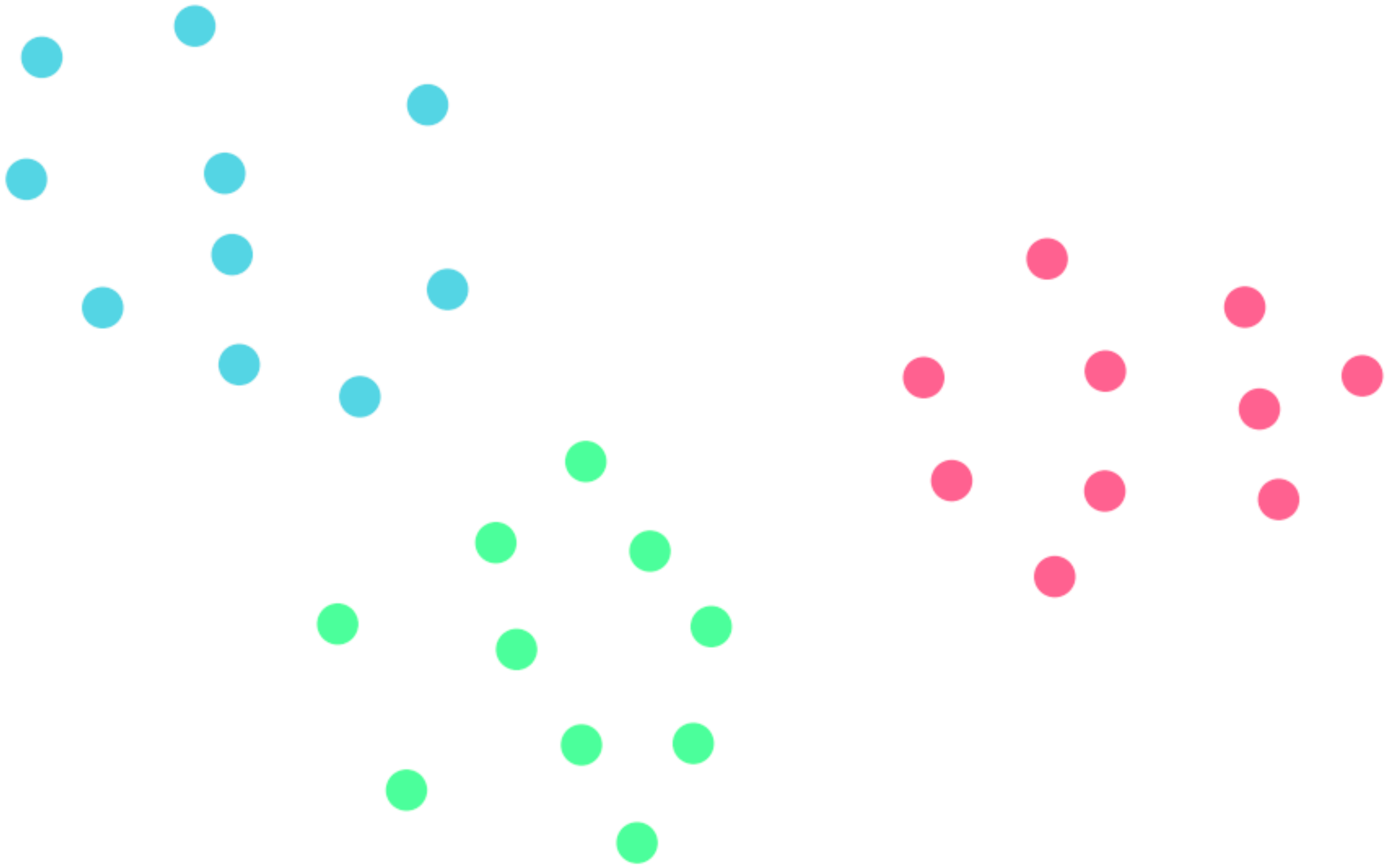
Mixture of C distributions $\mathbf{P} = \{\mathbf{P}_i\}_{i \in [C]}$ on $\mathbb{R}^d$ with densities $\{\rho_i\}_{i \in [C]}$

Contextual Stochastic Block Model (CSBM)



n nodes

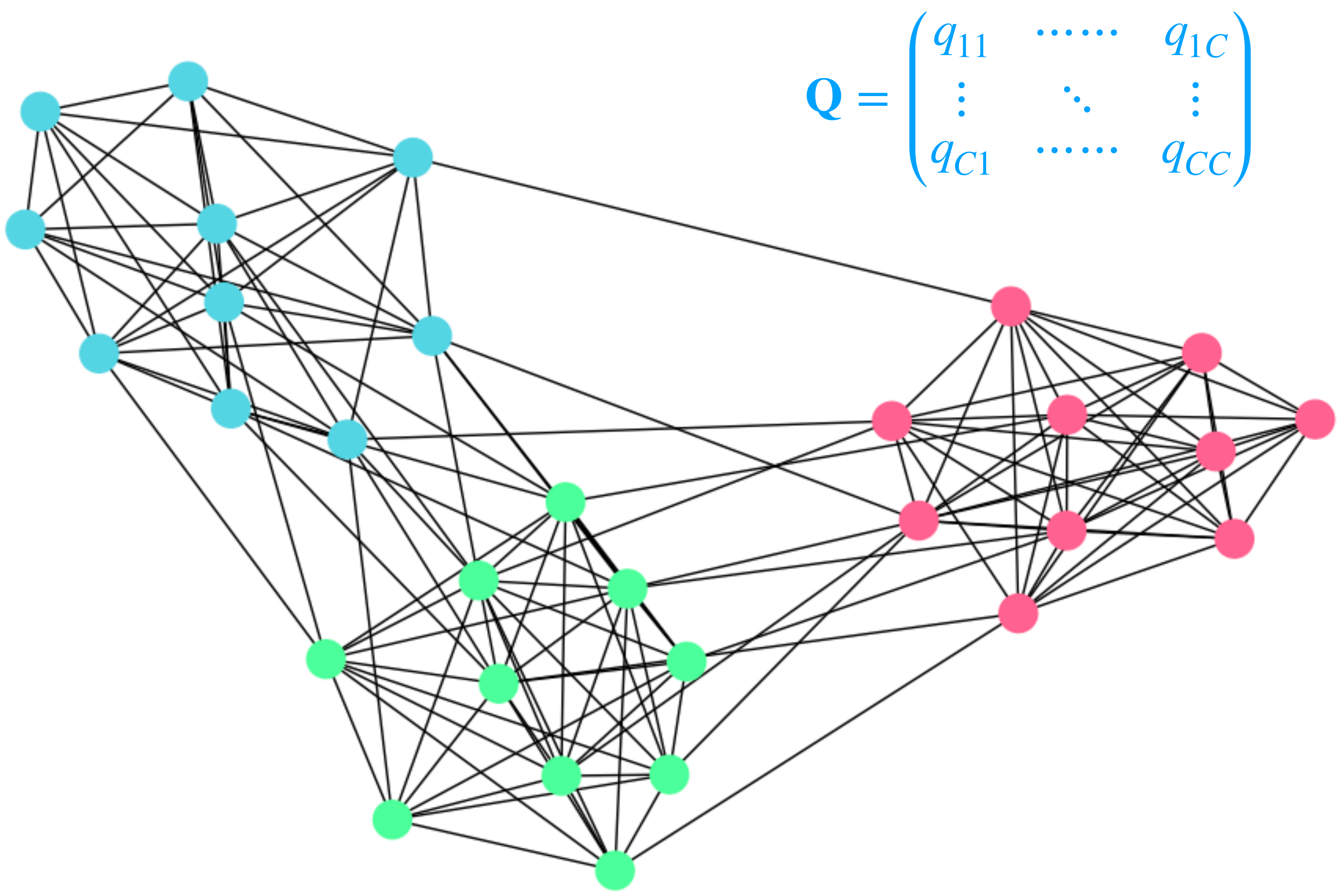
Contextual Stochastic Block Model (CSBM)



n nodes

$y_u \sim \text{Unif}([C])$ for all $u \in [n]$
Latent class labels

Contextual Stochastic Block Model (CSBM)



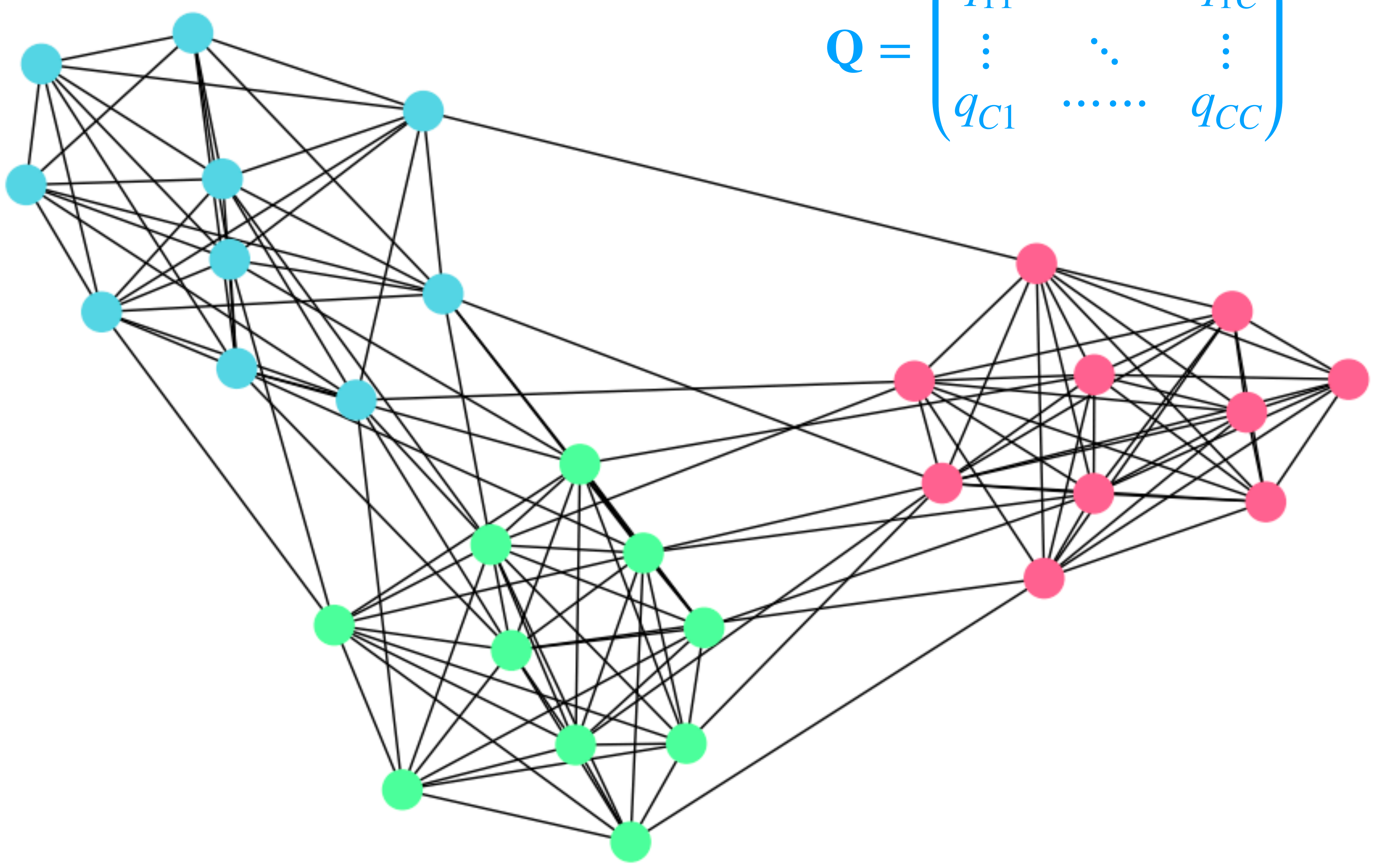
$$\mathbf{Q} = \begin{pmatrix} q_{11} & \cdots\cdots & q_{1C} \\ \vdots & \ddots & \vdots \\ q_{C1} & \cdots\cdots & q_{CC} \end{pmatrix}$$

n nodes

$y_u \sim$ Unif$([C])$ for all $u \in [n]$
Latent class labels

$A = (a_{uv})_{u,v \in [n]}$
$\Pr(a_{uv} = 1 \mid y_u, y_v) = q_{y_u y_v}$

Contextual Stochastic Block Model (CSBM)



$$\mathbf{Q} = \begin{pmatrix} q_{11} & \cdots\cdots & q_{1C} \\ \vdots & \ddots & \vdots \\ q_{C1} & \cdots\cdots & q_{CC} \end{pmatrix}$$

n nodes

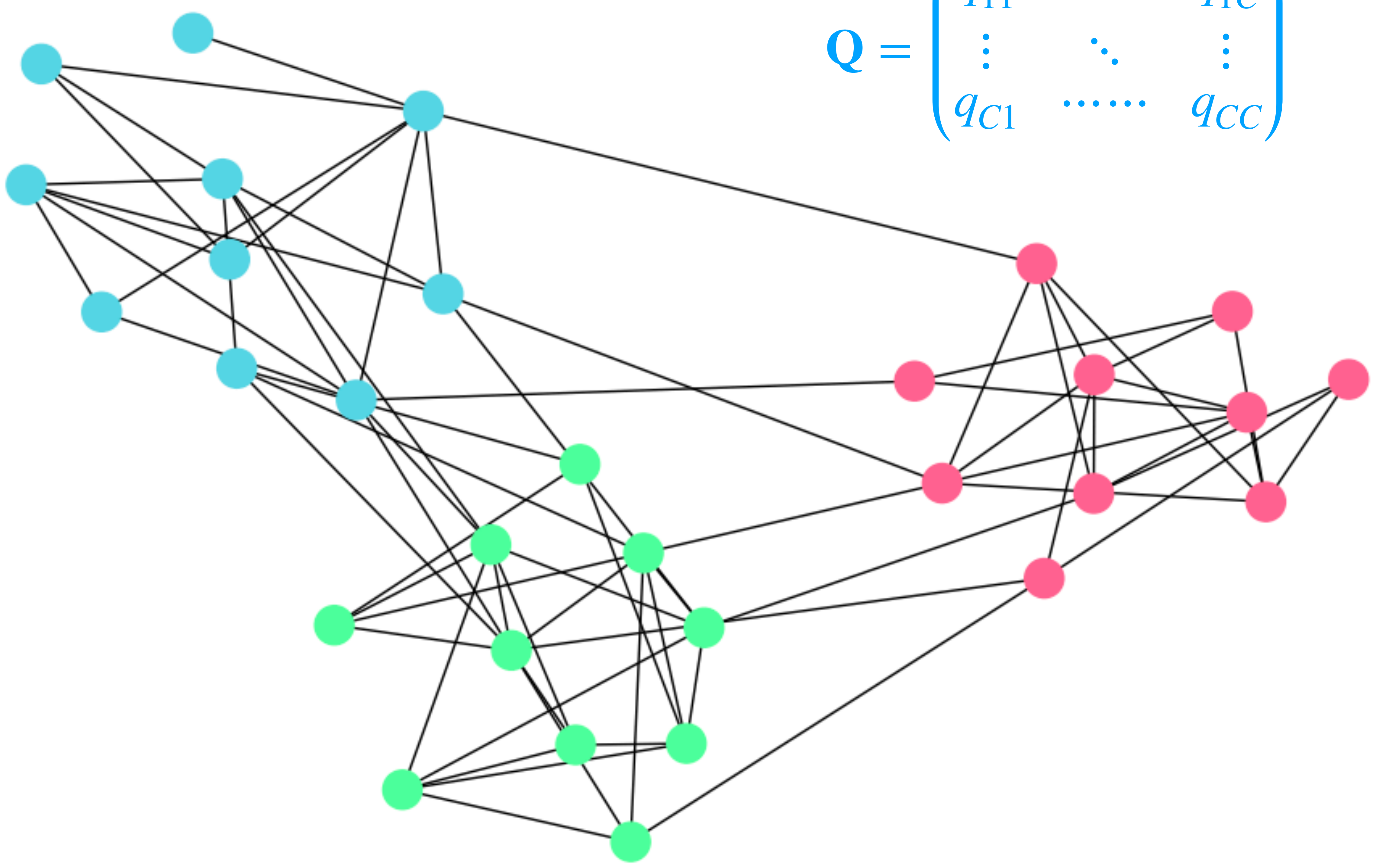
$y_u \sim \text{Unif}([C])$ for all $u \in [n]$
Latent class labels

$A = (a_{uv})_{u,v \in [n]}$
$\Pr(a_{uv} = 1 \mid y_u, y_v) = q_{y_u y_v}$

$$\mathbf{Q} = \frac{\mathbf{B}}{n} = \left(\frac{b_{ij}}{n}\right)_{i,j \in [C]}$$

$b_{ij} = \Omega_n(\log n)$

Contextual Stochastic Block Model (CSBM)



$$\mathbf{Q} = \begin{pmatrix} q_{11} & \cdots\cdots & q_{1C} \\ \vdots & \ddots & \vdots \\ q_{C1} & \cdots\cdots & q_{CC} \end{pmatrix}$$

n nodes

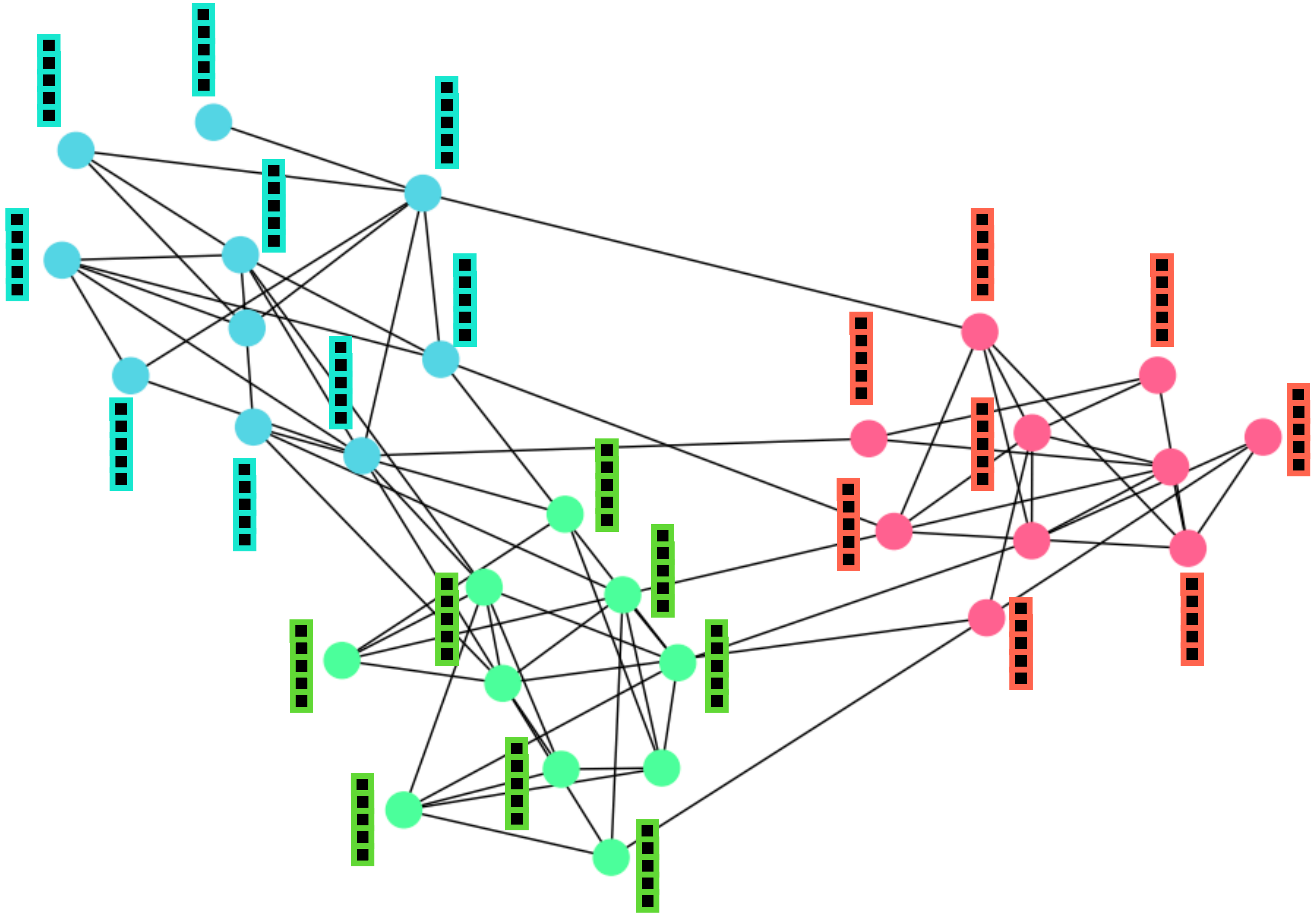
$y_u \sim$ Unif($[C]$) for all $u \in [n]$
Latent class labels

$A = (a_{uv})_{u,v\in[n]}$
$\Pr(a_{uv} = 1 \mid y_u, y_v) = q_{y_u y_v}$

$$\mathbf{Q} = \frac{\mathbf{B}}{n} = \left(\frac{b_{ij}}{n}\right)_{i,j\in[C]}$$

$b_{ij} = O_n(1)$

Contextual Stochastic Block Model (CSBM)

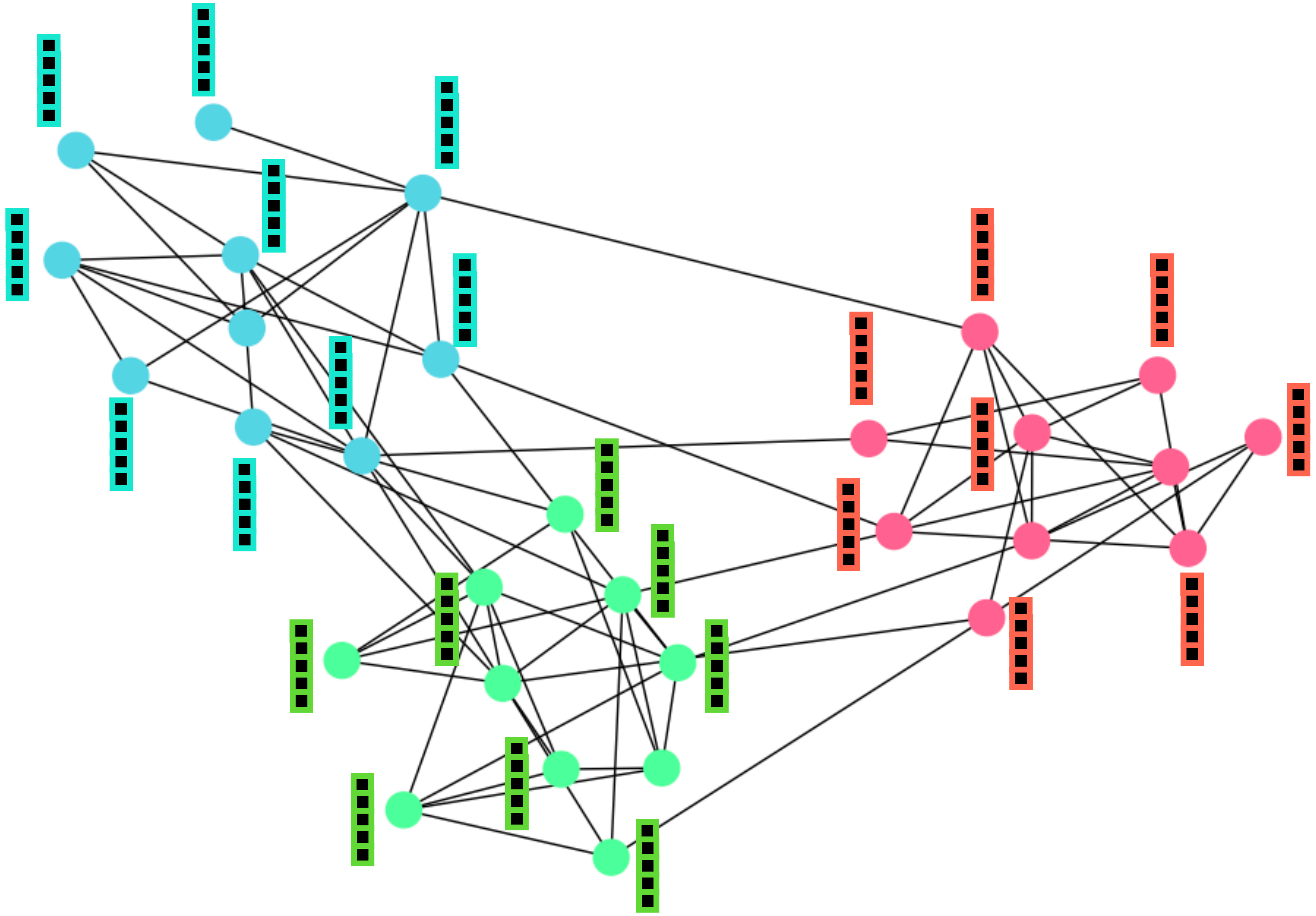


Node attributes:

$X_u \in \mathbb{R}^d \sim \mathbf{P}_{y_u}$ for all $u \in [n]$

Node attributes

Contextual Stochastic Block Model (CSBM)



Node attributes:

$X_u \in \mathbb{R}^d \sim \mathbf{P}_{y_u}$ for all $u \in [n]$
Node attributes

$G_n = (\mathbf{A}, \mathbf{X}) \sim$ CSBM$(n, \mathbf{P}, \mathbf{Q})$

Overview

- Understanding a graph convolution operation [ICML 2021]
  - Improvement in separability threshold
  - Generalization error of the linear classifier
- Effects of graph convolutions in multilayer networks [ICLR 2023]
  - Isolate convolutions from the layers of a neural network
  - Understand effects in terms of relevant signals
- Optimality of message-passing GNNs [NeurIPS 2023]
  - Develop a notion of optimal classifier for node-classification problems
  - Design a neural network architecture that can realize the optimal classifier

Part I

- Understanding a graph convolution operation [ICML 2021]
  - Improvement in separability threshold
  - Generalization error of the linear classifier
- Effects of graph convolutions in multilayer networks [ICLR 2023]
  - Isolate convolutions from the layers of a neural network
  - Understand effects in terms of relevant signals
- Optimality of message-passing GNNs [NeurIPS 2023]
  - Develop a notion of optimal classifier for node-classification problems
  - Design a neural network architecture that can realize the optimal classifier

Part I

- Effect of **one** graph convolution on a binary Gaussian mixture
    - Comparison with baseline — absence of relational information
    - Improvement in linear separability
- Generalization of the linear classifier on out-of-distribution relational data

Model and Assumptions

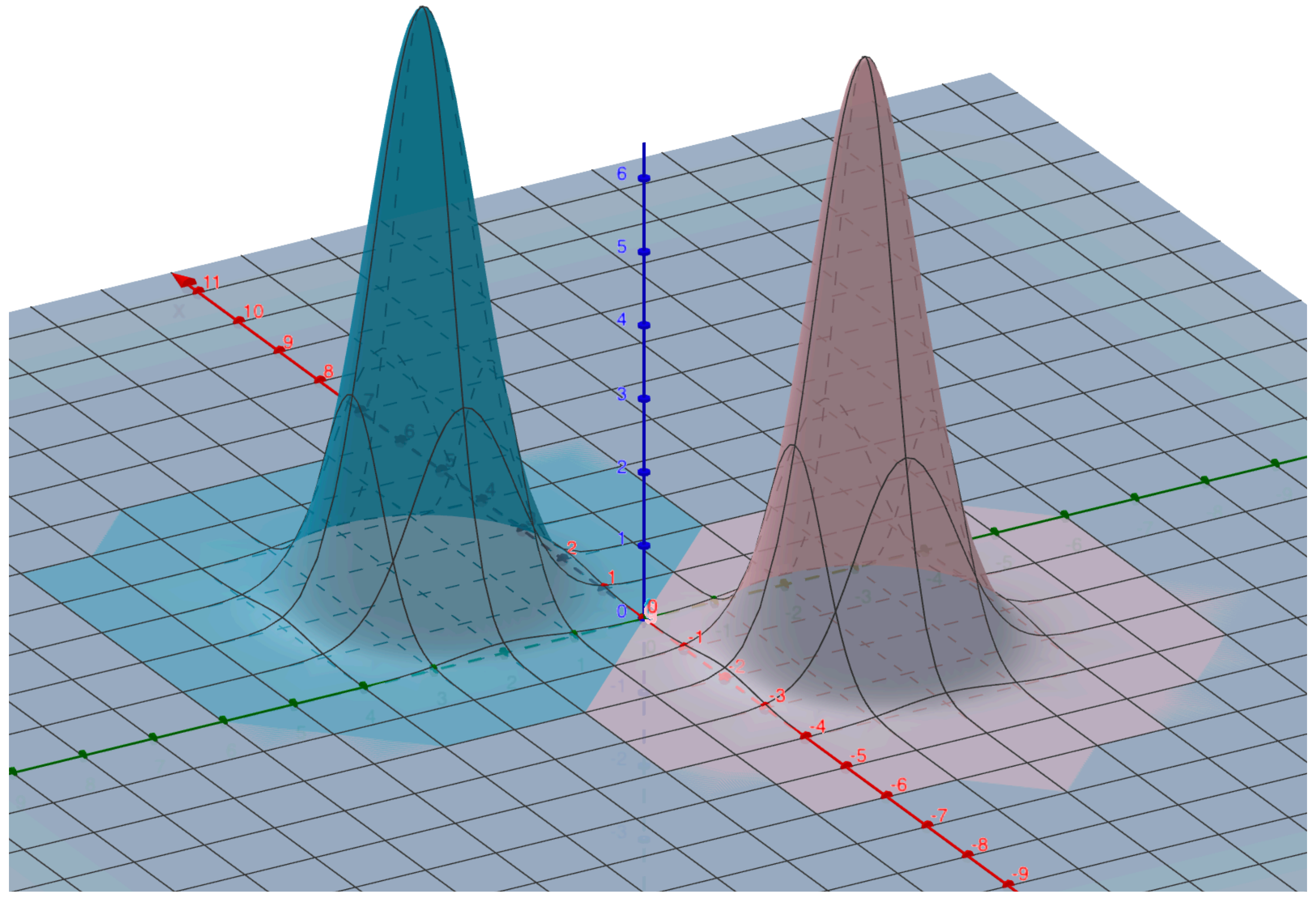
$$\mathbf{P} = \{\mathcal{N}(\mu, \sigma^2 I),$$
$$\mathcal{N}(\nu, \sigma^2 I)\}$$

$$\mathbf{Q} = \begin{pmatrix} p & q \\ q & p \end{pmatrix}$$

Model and Assumptions

$$\mathbf{P} = \{\mathcal{N}(\mu, \sigma^2 I),\\ \mathcal{N}(\nu, \sigma^2 I)\}$$

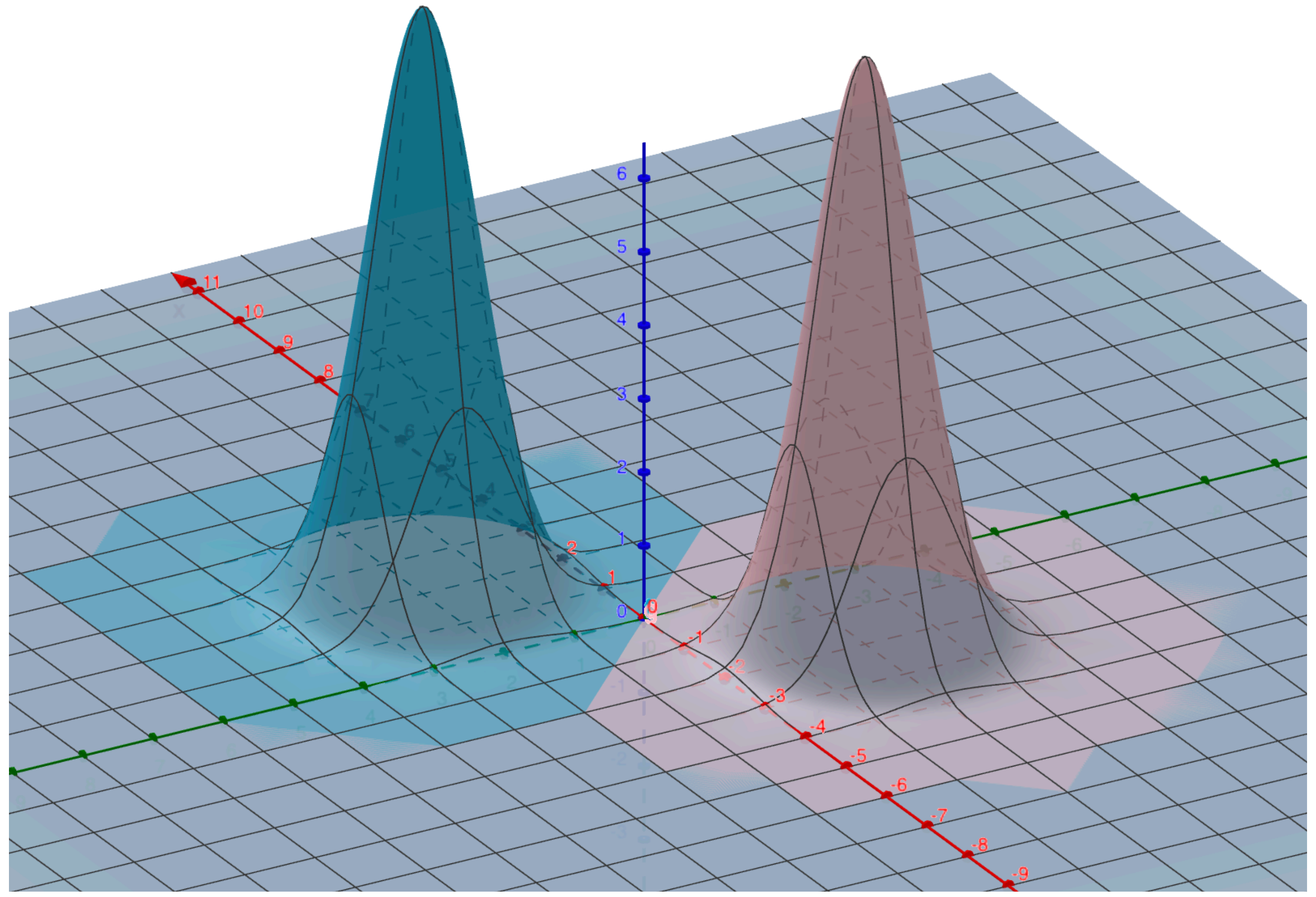
$$\mathbf{Q} = \begin{pmatrix} p & q \\ q & p \end{pmatrix}$$



Model and Assumptions

$$\mathbf{P} = \{\mathcal{N}(\mu, \sigma^2 I),\ \mathcal{N}(\nu, \sigma^2 I)\}$$

$$\mathbf{Q} = \begin{pmatrix} p & q \\ q & p \end{pmatrix}$$



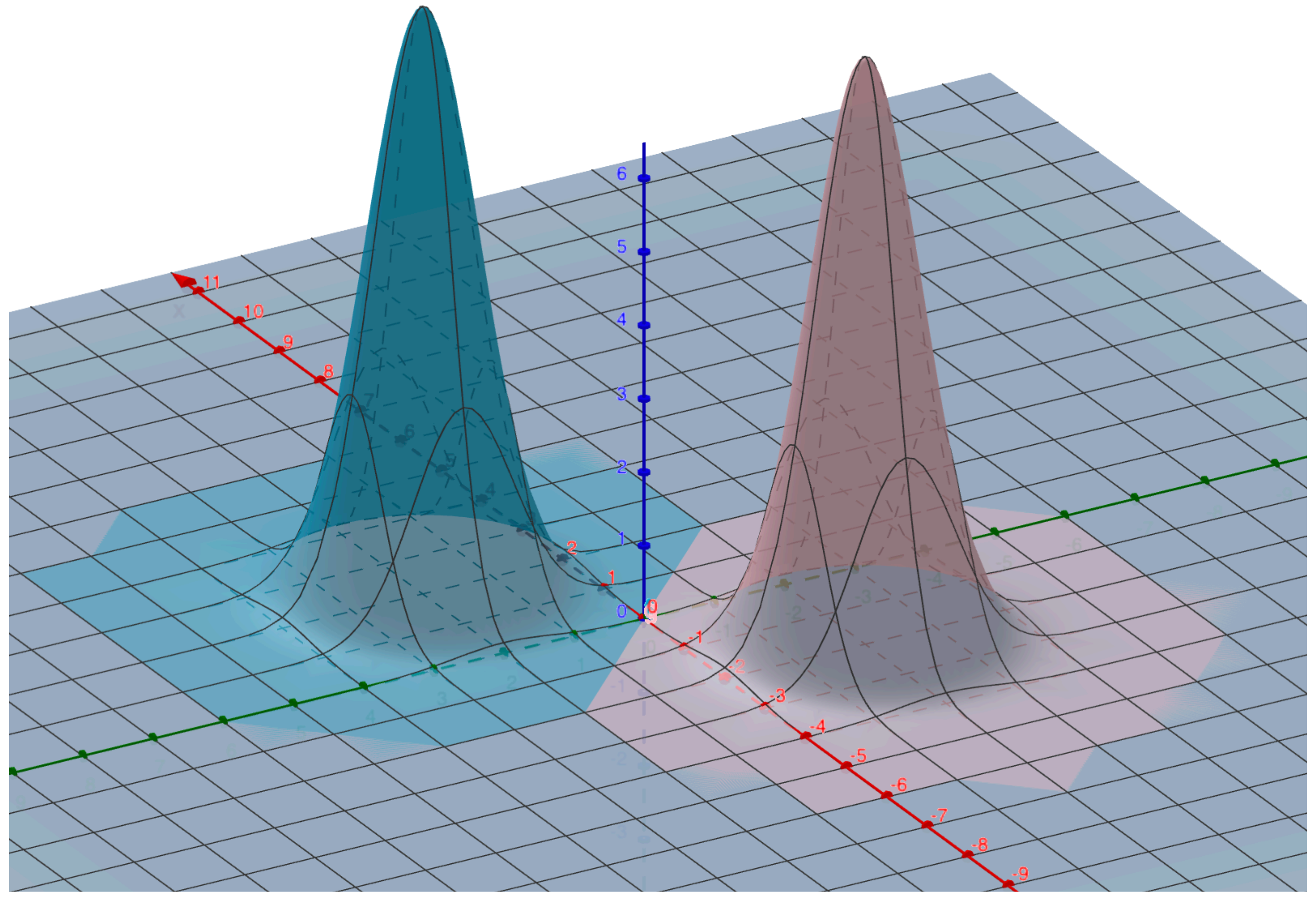
Feature signal $\zeta = \dfrac{2\|\mu\|}{\sigma}$

Graph signal $\gamma = \dfrac{|p-q|}{p+q}$

Model and Assumptions

$$\mathbf{P} = \{\mathcal{N}(\mu, \sigma^2 I), \\ \mathcal{N}(\nu, \sigma^2 I)\}$$

$$\mathbf{Q} = \begin{pmatrix} p & q \\ q & p \end{pmatrix}$$



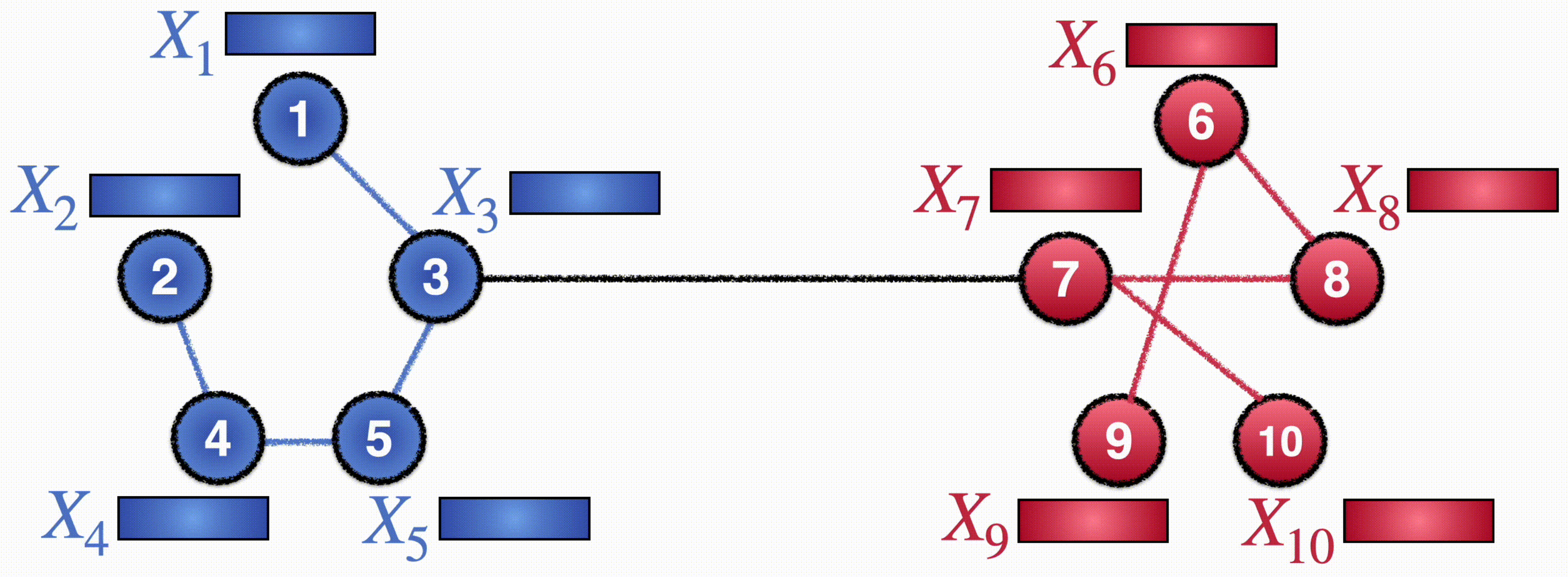
Feature signal $\zeta = \dfrac{2\|\mu\|}{\sigma}$

Graph signal $\gamma = \dfrac{|p - q|}{p + q}$

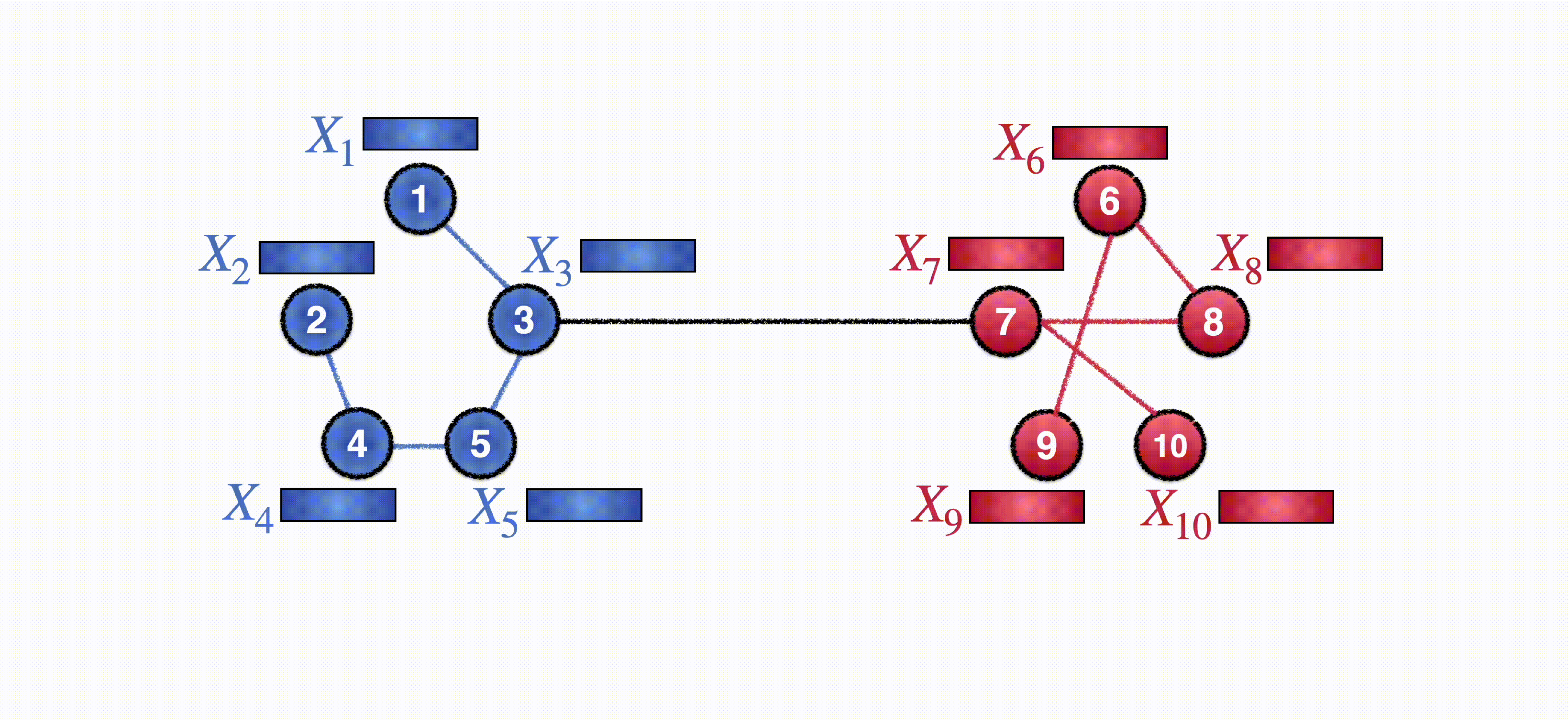
Assumption: $np, nq = \Omega(\log^2 n)$

$$\mathbf{E} \deg = \frac{n}{2}(p + q) = \Omega(\log^2 n)$$

Graph Convolutions



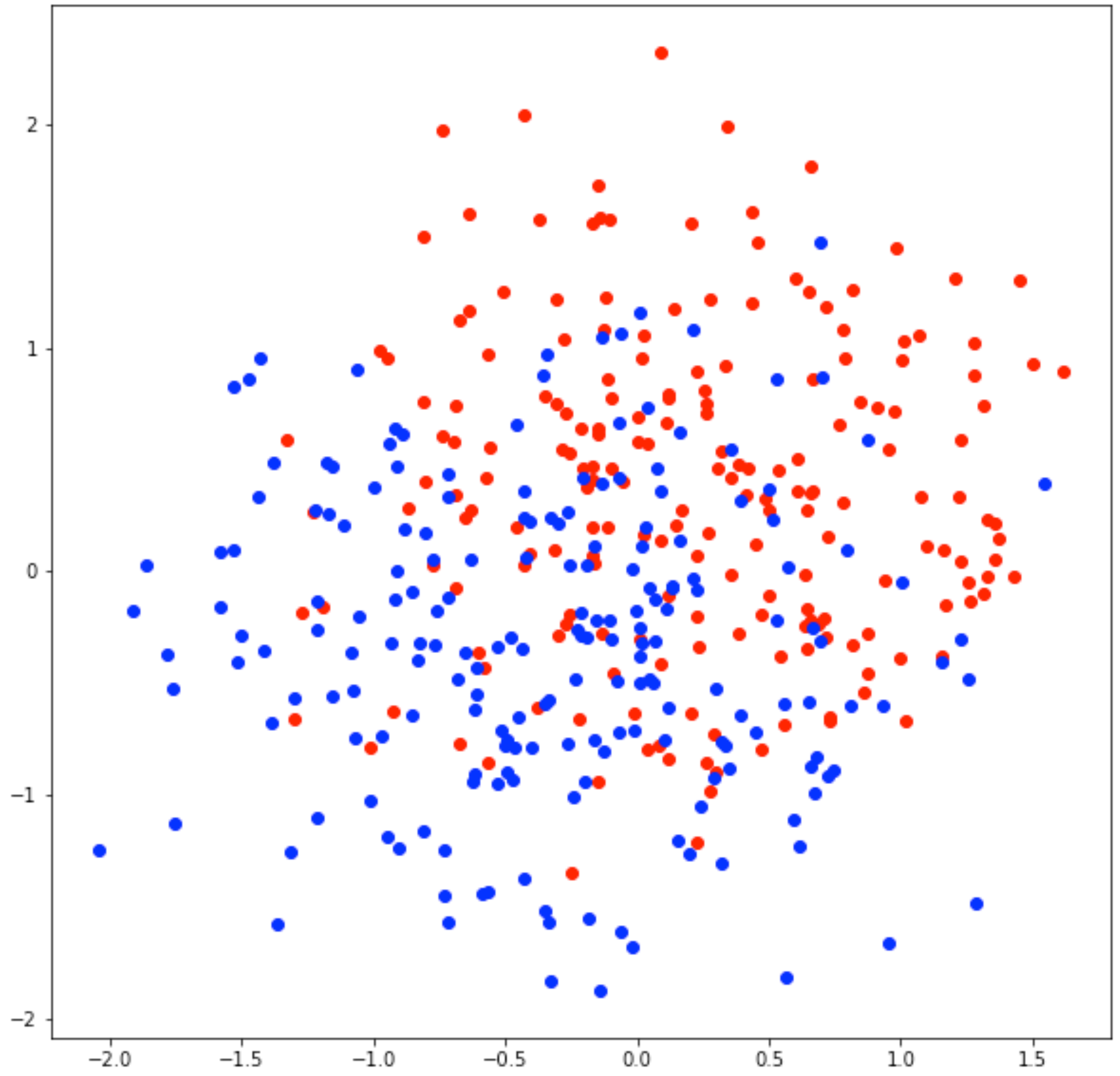
Graph Convolutions



Convolved feature matrix: $X' = D^{-1}AX$

$$X'_u = \frac{1}{\deg(u)} \sum_{v \in [n]} a_{uv} X_v$$

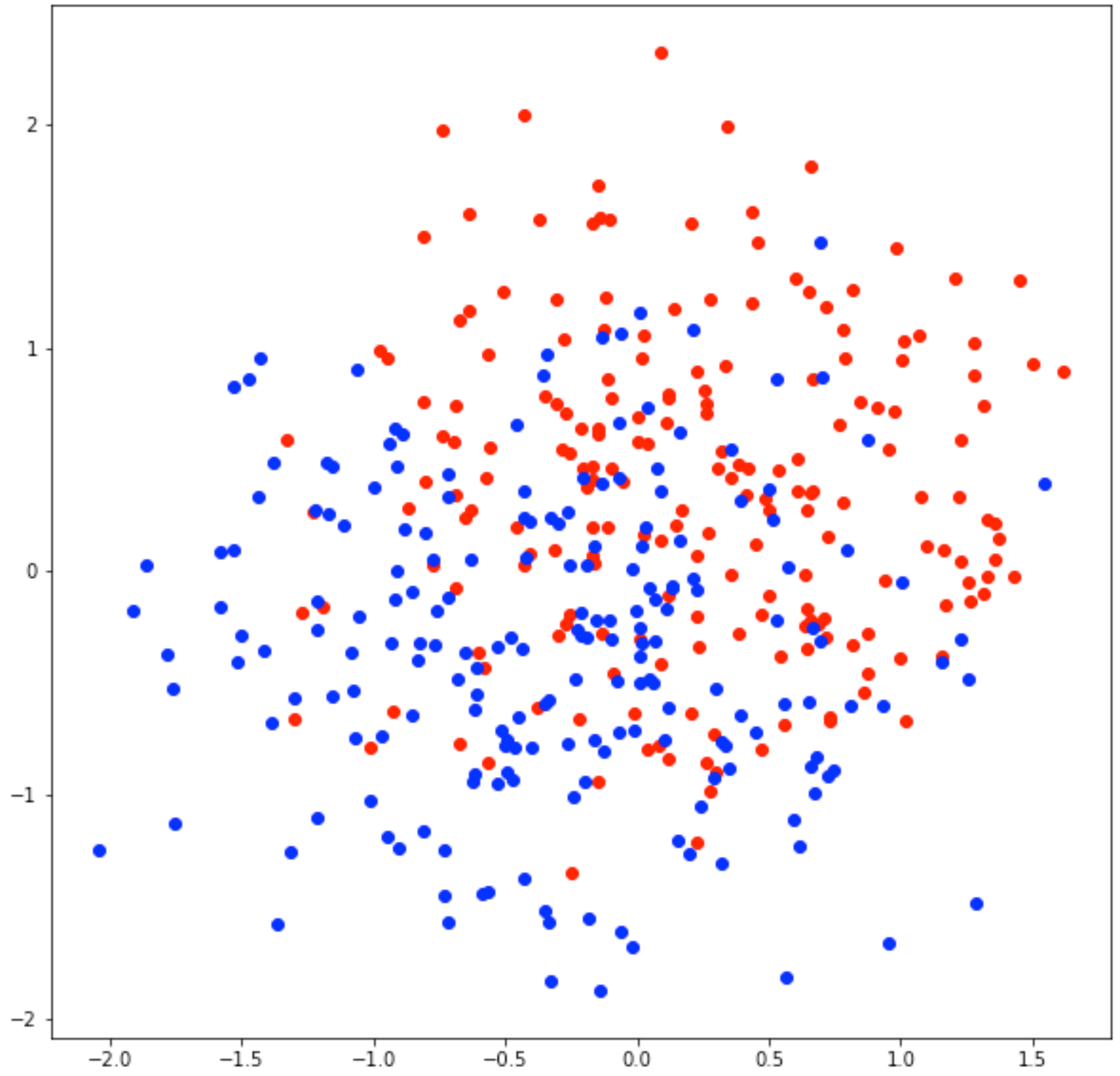
What can graph convolution do?



Original Data

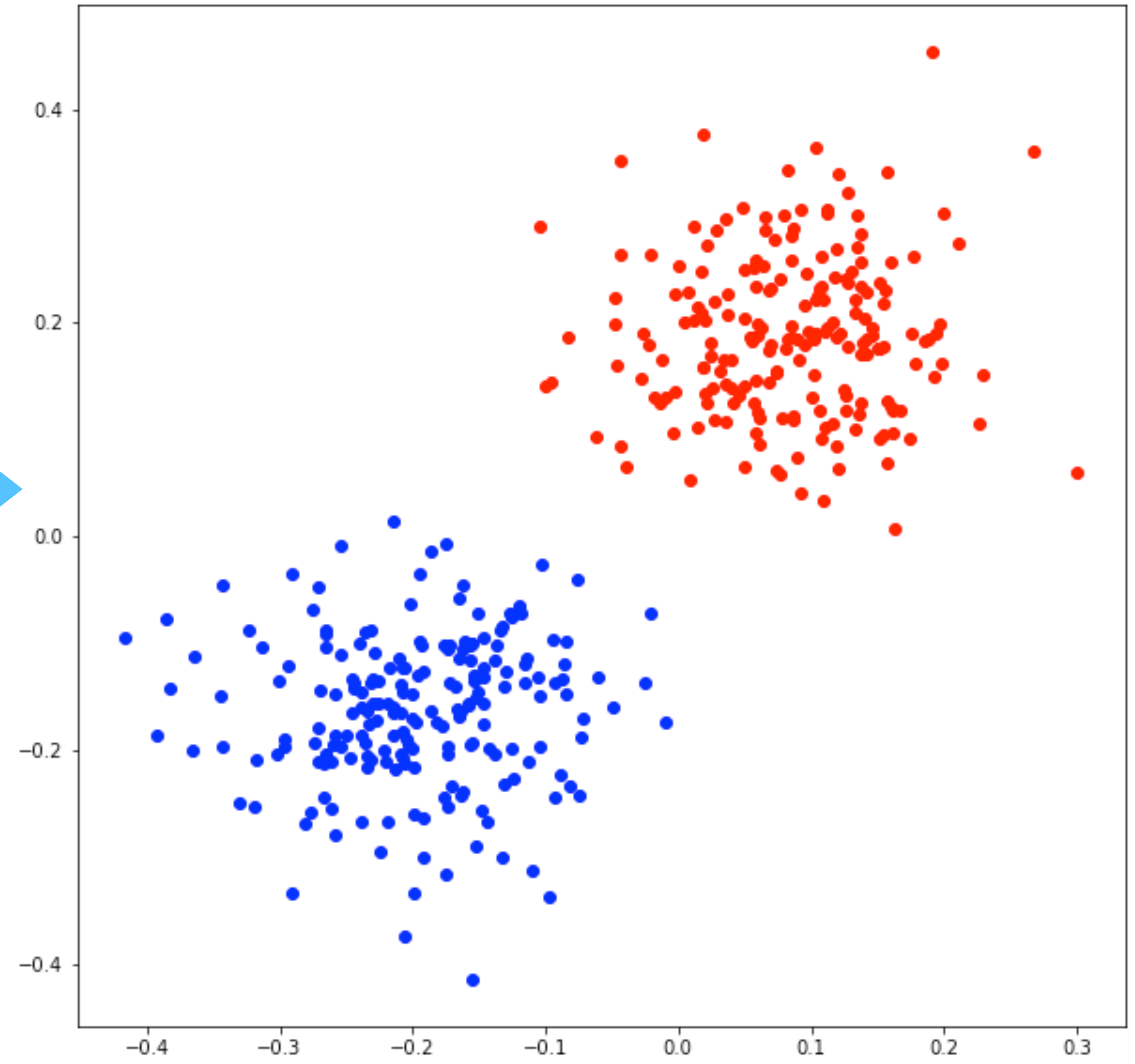
- Consider distributions with 2D features
- We cannot separate the classes linearly due to the large overlap between them

What can graph convolution do?



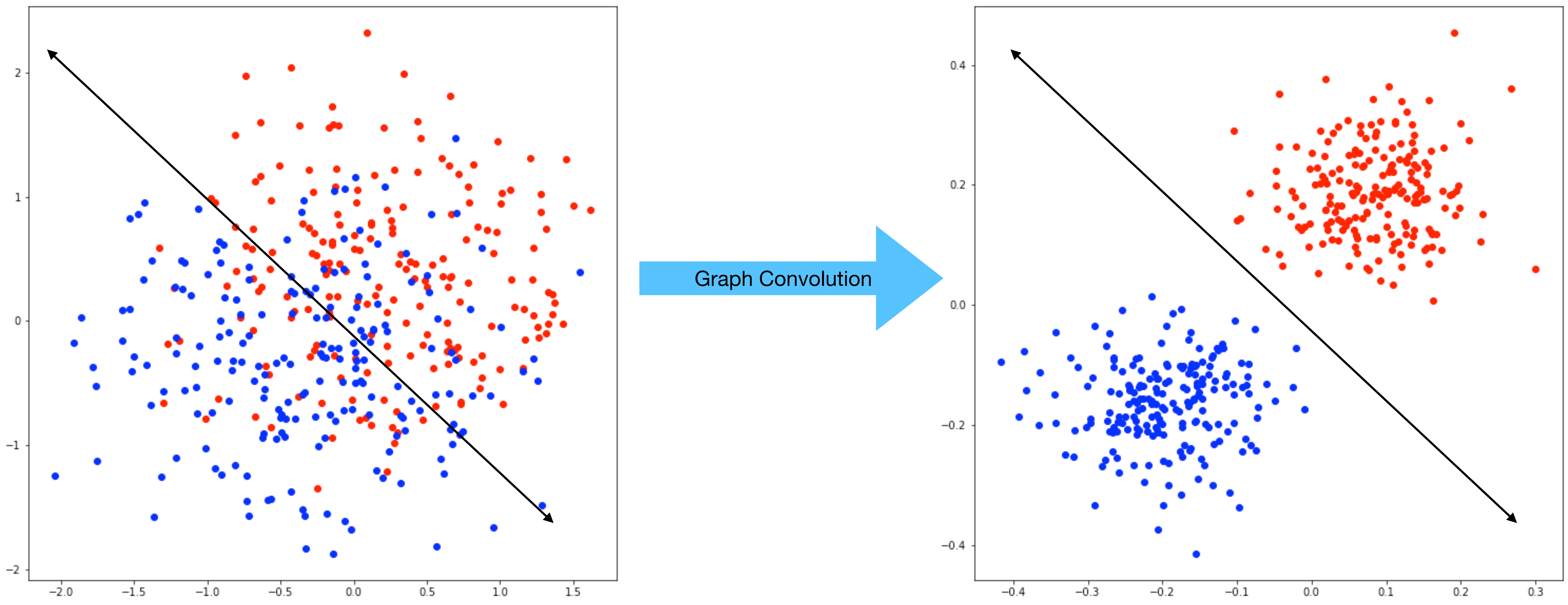
Original Data

Graph Convolution



After graph convolution

What can graph convolution do?



Original Data

After graph convolution

Graph convolution makes the data linearly separable

Graph improves linear separability

- Without the graph,

$$\zeta = \frac{\|\mu - \nu\|_2}{\sigma} = O_n(1) \Longrightarrow \mathbb{P}(\{X_u\}_{u\in[n]} \text{ are linearly separable}) = o_n(1)$$

$$\text{BCE Loss} \geq c \cdot \Phi(-\zeta)$$

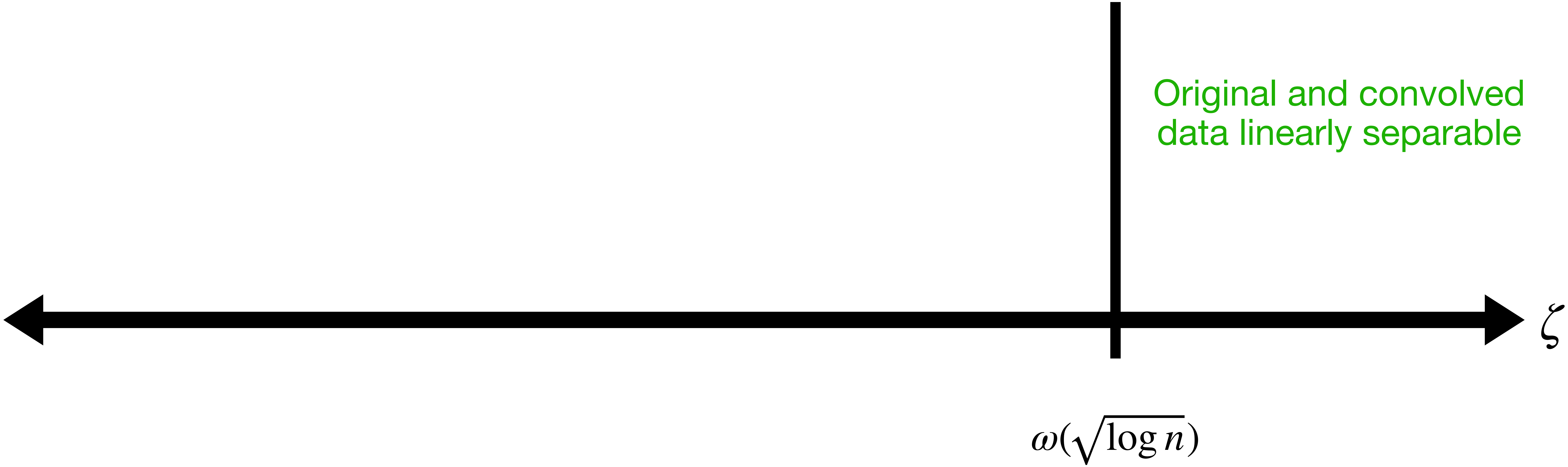
- With graph convolution, this threshold changes to

$$\zeta = O_n\left(\frac{1}{\sqrt{\mathbb{E}\deg}}\right)$$

Expected degree of a node

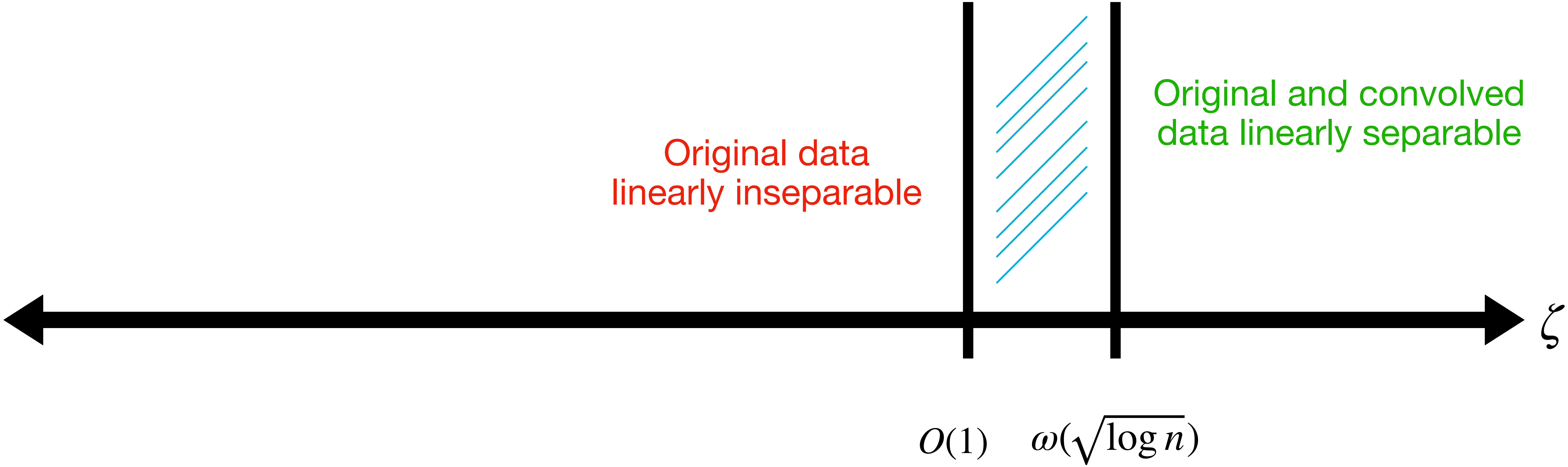
Graph improves linear separability

Graph improves linear separability

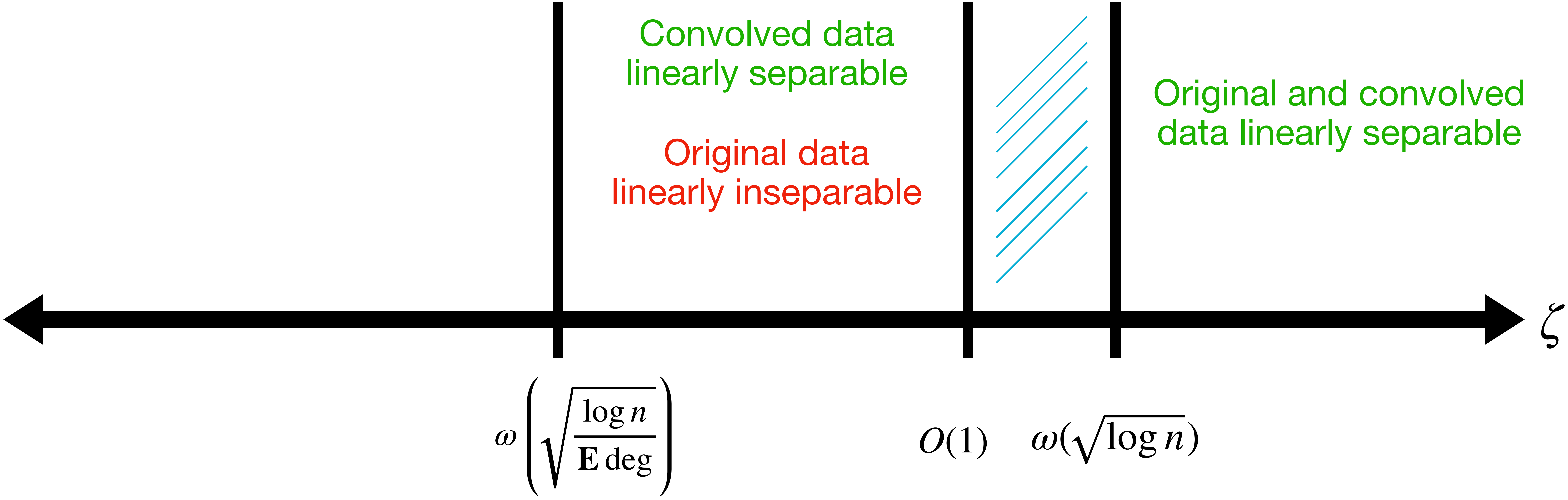


Original and convolved data linearly separable

Graph improves linear separability


Original data linearly inseparable
Original and convolved data linearly separable
ζ
$O(1)$
$\omega(\sqrt{\log n})$

Graph improves linear separability

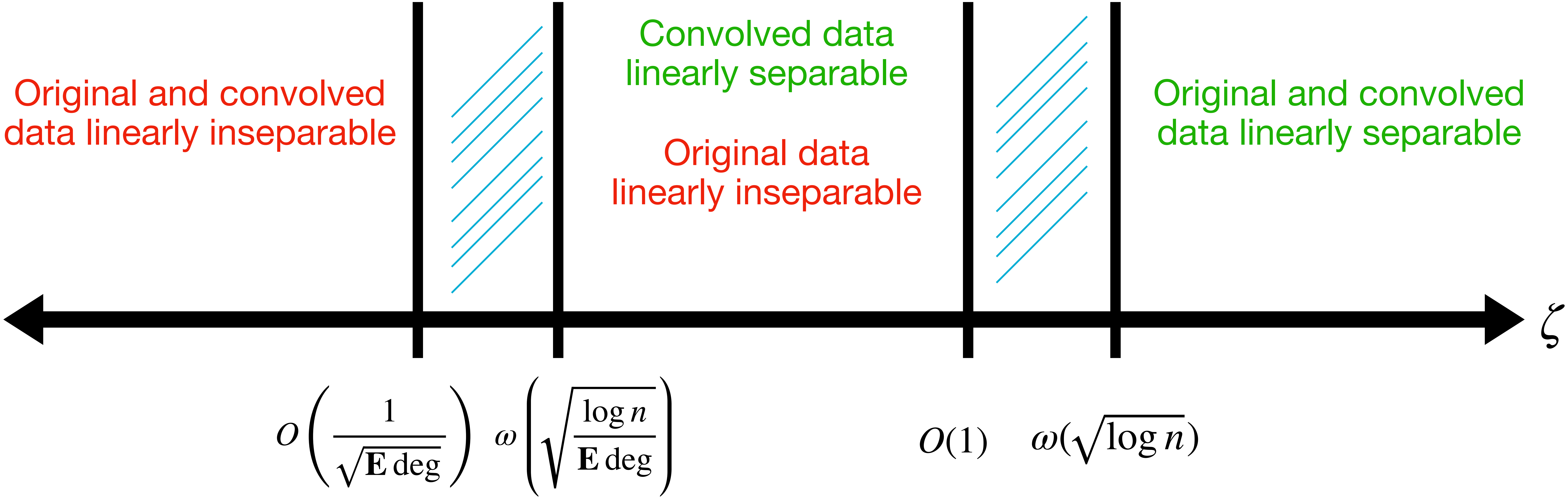


Convolved data linearly separable

Original data linearly inseparable

Original and convolved data linearly separable

Graph improves linear separability



Original and convolved data linearly inseparable

Convolved data linearly separable

Original data linearly inseparable

Original and convolved data linearly separable

$o\left(\frac{1}{\sqrt{\mathbf{E}\deg}}\right)$ $\omega\left(\sqrt{\frac{\log n}{\mathbf{E}\deg}}\right)$ $O(1)$ $\omega(\sqrt{\log n})$

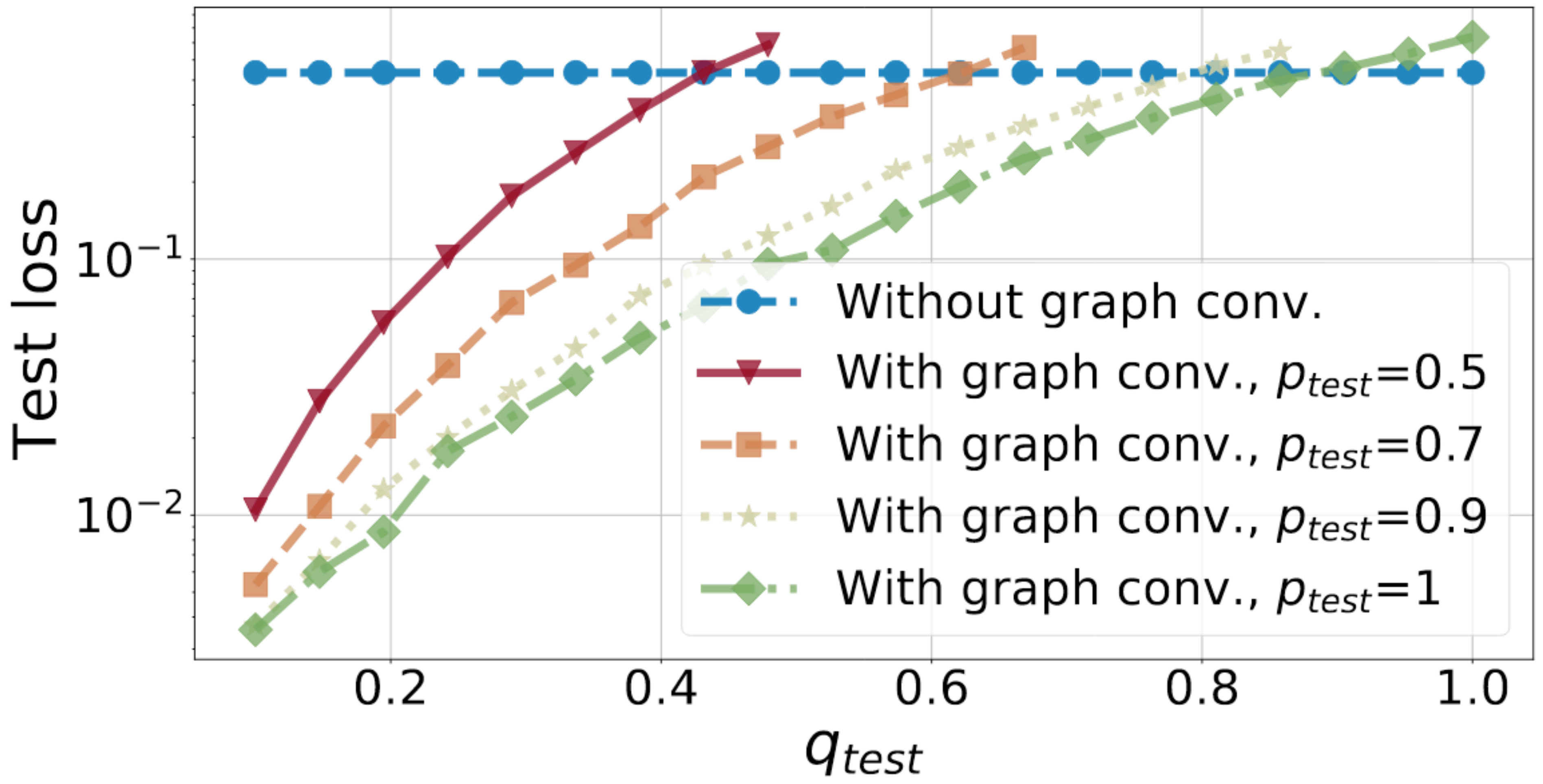
ζ

Generalization

- For any new dataset A, X with different $n, \mathbf{Q}$, the loss is bounded above

$$\text{Loss}(A, X) \leq C \exp(-\zeta\gamma)$$

- Loss increases with inter-class edge probability (noisy graph)



(a) $\|\mu_d - \nu_d\| = 2/\sqrt{d}$

Part II

- Understanding a graph convolution operation [ICML 2021]
  - Improvement in separability threshold
  - Generalization error of the linear classifier

- Effects of graph convolutions in multilayer networks [ICLR 2023]
  - Isolate convolutions from the layers of a neural network
  - Understand effects in terms of relevant signals in the data

- Optimality of message-passing GNNs [NeurIPS 2023]
  - Develop a notion of optimal classifier for node-classification problems
  - Design a neural network architecture that can realize the optimal classifier

Part II

- Complete characterization of up to 2 graph convolutions (GCs) in networks with up to 3 layers
  - Improvement in the classification threshold
  - Comparison of various placement choices for convolutions
- Theoretical analysis on CSBM modelled after XOR data
- Empirical demonstration of results in various settings

Architecture

- Two sources of information: $(\mathbf{A}, \mathbf{X})$

$$\begin{aligned} \mathbf{H}^{(0)} &= \mathbf{X}, \\ \left.\begin{aligned} f^{(l)}(\mathbf{X}) &= (\mathbf{D}^{-1}\mathbf{A})^{k_l}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)} + \mathbf{b}^{(l)} \\ \mathbf{H}^{(l)} &= \mathrm{ReLU}(f^{(l)}(\mathbf{X})) \end{aligned}\right\} &\text{ for } l \in [L], \\ \hat{\mathbf{y}} &= \varphi(f^{(L)}(\mathbf{X})). \end{aligned}$$

- $\mathbf{X} \in \mathbb{R}^{n \times d} \rightarrow$ input data
- $\varphi \rightarrow$ sigmoid function
- $\hat{\mathbf{y}} \rightarrow$ output of the network
- $k_l \rightarrow$ number of GCs in layer l

- A generalization of Kipf and Welling's GCN with variable GCs at each layer
- Similar models analyzed previously with power iterations in the last layer [Gasteiger, Bojchevski, Günnemann (2019)] or first layer [Frasca et al., SIGN (2020)]:
  - Empirically known to have comparable performance to SOTA

Architecture

- Two sources of information: $(\mathbf{A}, \mathbf{X})$

$$\mathbf{H}^{(0)} = \mathbf{X},$$

$$\left.\begin{aligned} f^{(l)}(\mathbf{X}) &= (\mathbf{D}^{-1}\mathbf{A})^{\boxed{k_l}}\mathbf{H}^{(l-1)}\mathbf{W}^{(l)} + \mathbf{b}^{(l)} \\ \mathbf{H}^{(l)} &= \mathrm{ReLU}(f^{(l)}(\mathbf{X})) \end{aligned}\right\} \text{for } l \in [L],$$

$$\hat{\mathbf{y}} = \varphi(f^{(L)}(\mathbf{X})).$$

- $\mathbf{X} \in \mathbb{R}^{n \times d}$ → input data
- φ → sigmoid function
- $\hat{\mathbf{y}}$ → output of the network
- k_l → number of GCs in layer l

- A generalization of Kipf and Welling’s GCN with variable GCs at each layer
- Similar models analyzed previously with power iterations in the last layer [Gasteiger, Bojchevski, Günnemann (2019)] or first layer [Frasca et al., SIGN (2020)]:
  - Empirically known to have comparable performance to SOTA

Data model

- Linear classifiers can be realized using one-layer NNs
- Class of one-layer NNs is too simple to capture the extent of GC effects
- Need to look at multi-layer NNs for placement questions
- Relevant SNR in the data

Data model

- Four-component XOR-based CSBM

- $\mathbf{P} = \{\text{Unif}(\mathcal{N}(\pm\mu, \sigma^2 I)), \text{Unif}(\mathcal{N}(\pm\nu, \sigma^2 I))\}$

$$X_u \sim \mathcal{N}(\pm\mu, \sigma^2 I) \text{ if } u \in C_1$$
$$X_u \sim \mathcal{N}(\pm\nu, \sigma^2 I) \text{ if } u \in C_2$$

- $\mathbf{Q} = \begin{pmatrix} p & q \\ q & p \end{pmatrix}$

Data model

- Four-component XOR-based CSBM

- $\mathbf{P} = \{\text{Unif}(\mathcal{N}(\pm\mu, \sigma^2 I)), \text{Unif}(\mathcal{N}(\pm\nu, \sigma^2 I))\}$

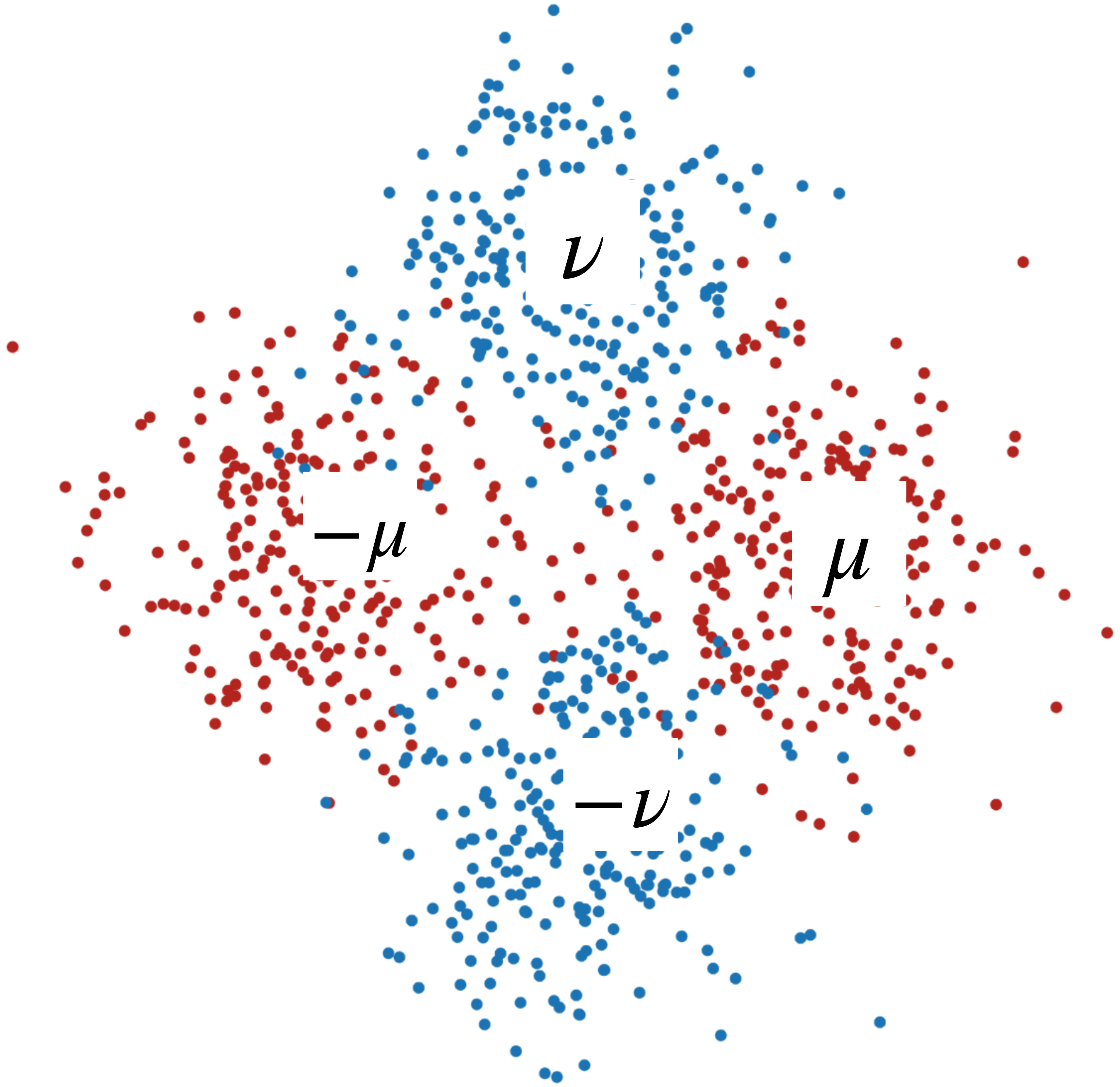
$$X_u \sim \mathcal{N}(\pm\mu, \sigma^2 I) \text{ if } u \in C_1$$
$$X_u \sim \mathcal{N}(\pm\nu, \sigma^2 I) \text{ if } u \in C_2$$

- $\mathbf{Q} = \begin{pmatrix} p & q \\ q & p \end{pmatrix}$

Assumptions:

$$\langle \mu, \nu \rangle = 0$$
$$np, nq = \Omega(\log^2 n)$$

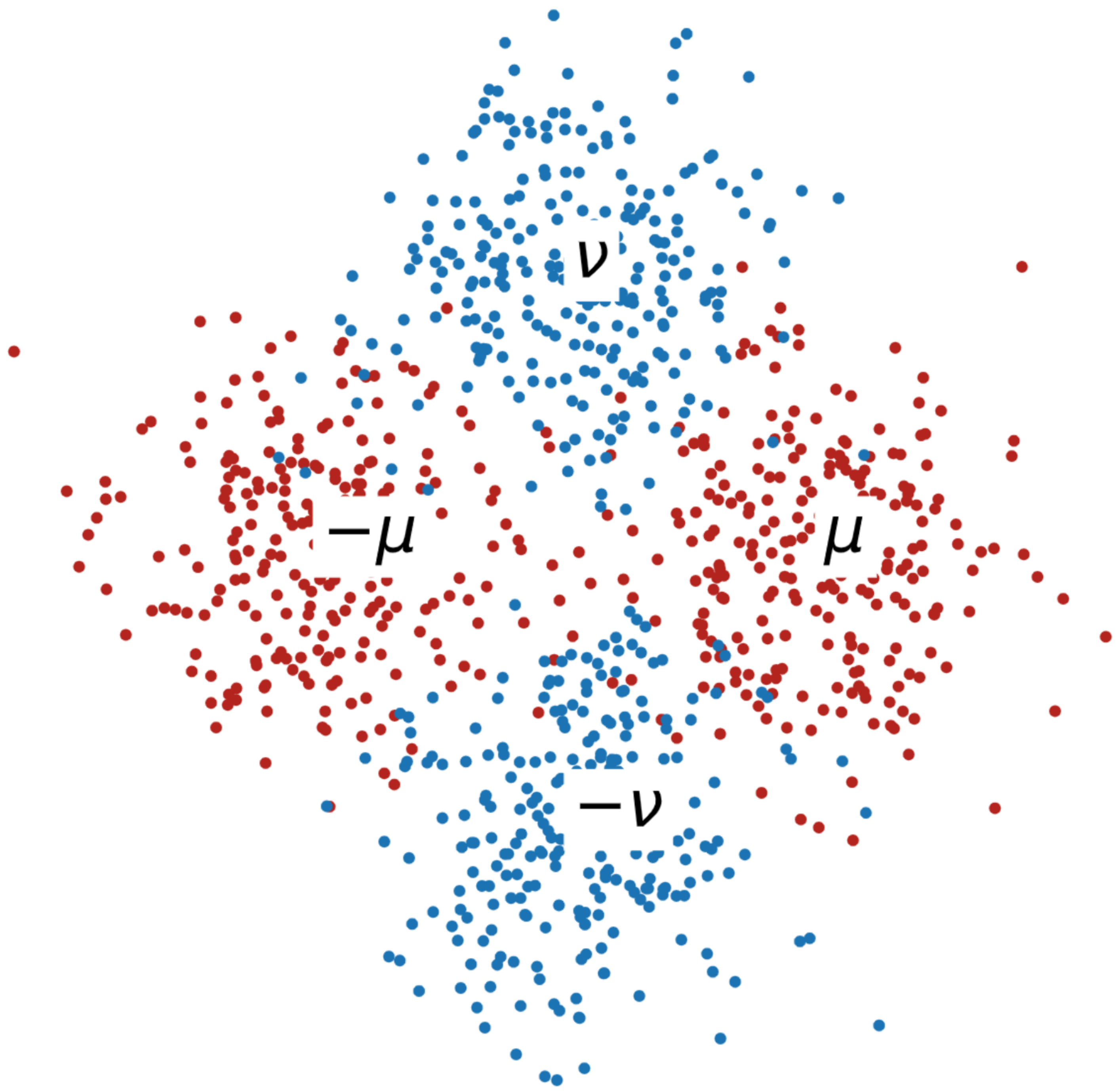
Data model



Identified signals in data

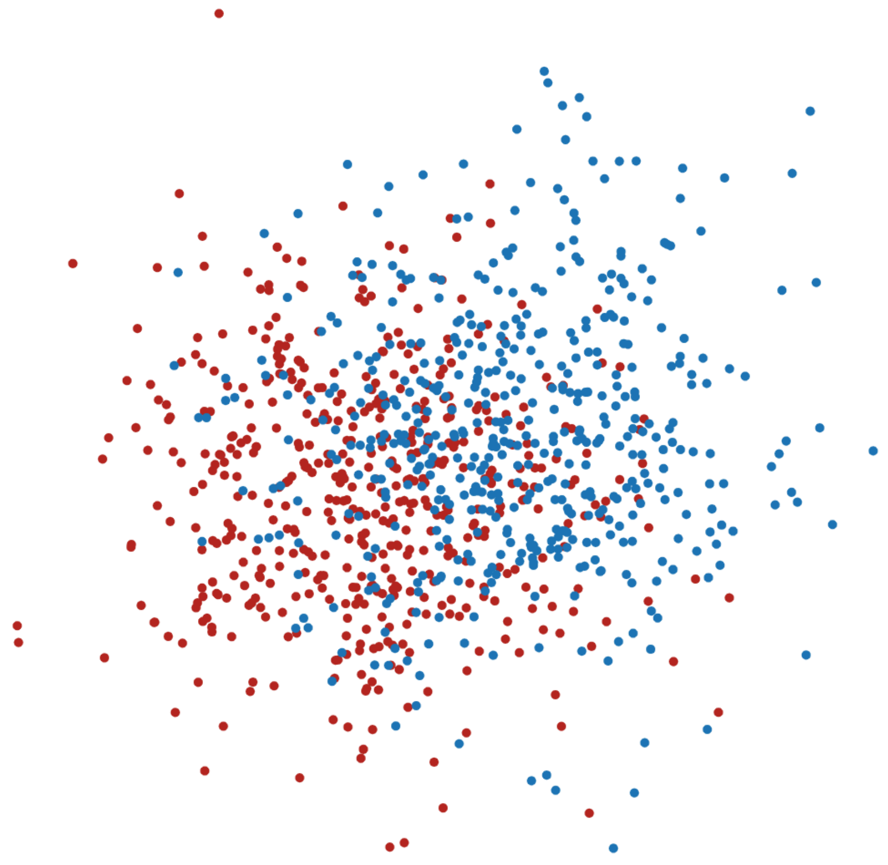
$$\zeta = \frac{\|\mu - \nu\|}{\sigma}, \qquad \gamma = \frac{|p - q|}{p + q}$$

Data model



Original input node features

GC in first layer



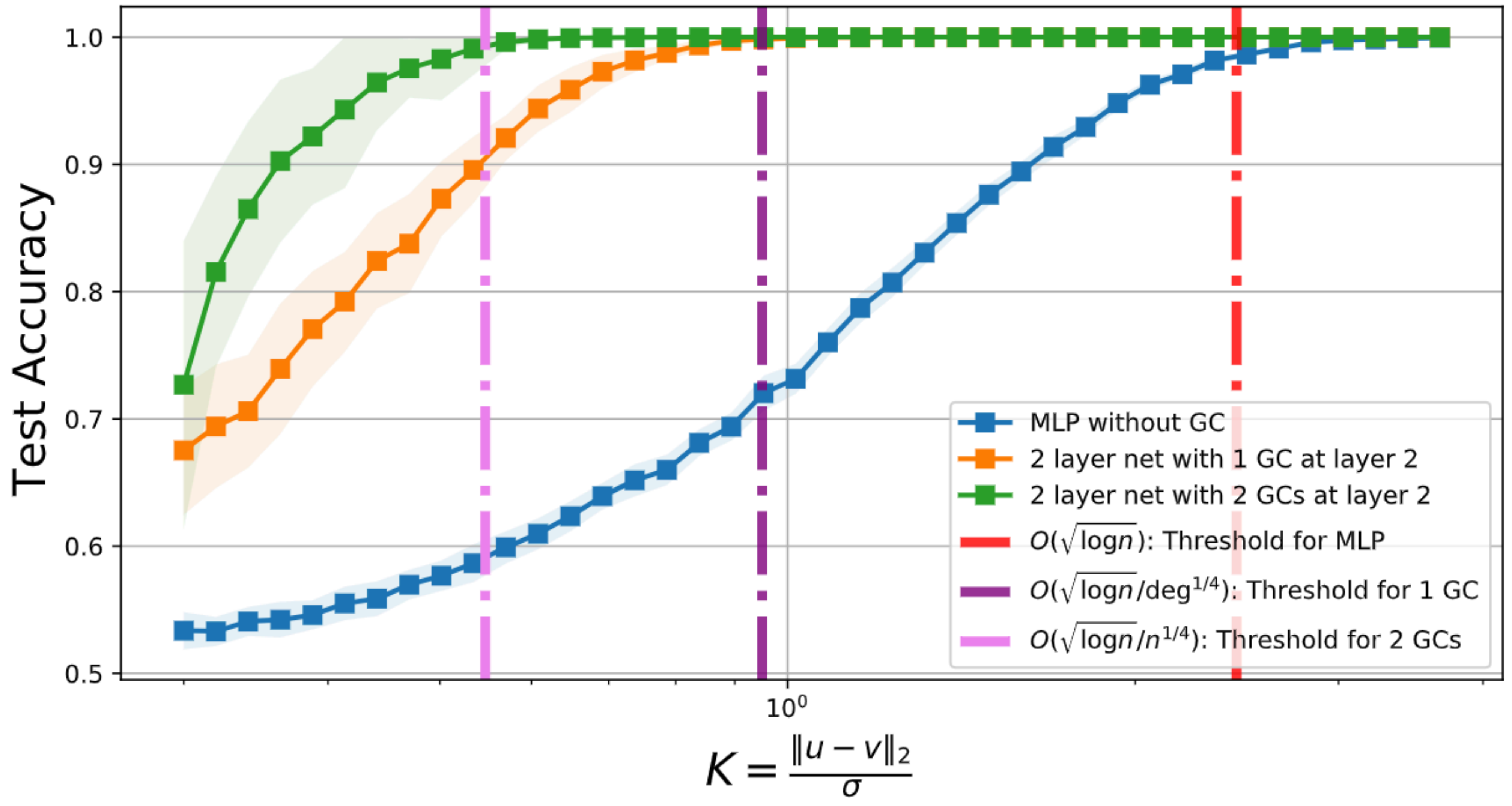
Features after GC at the first layer

A typical two-layer GCN (one GC in each layer) performs poorly on this data

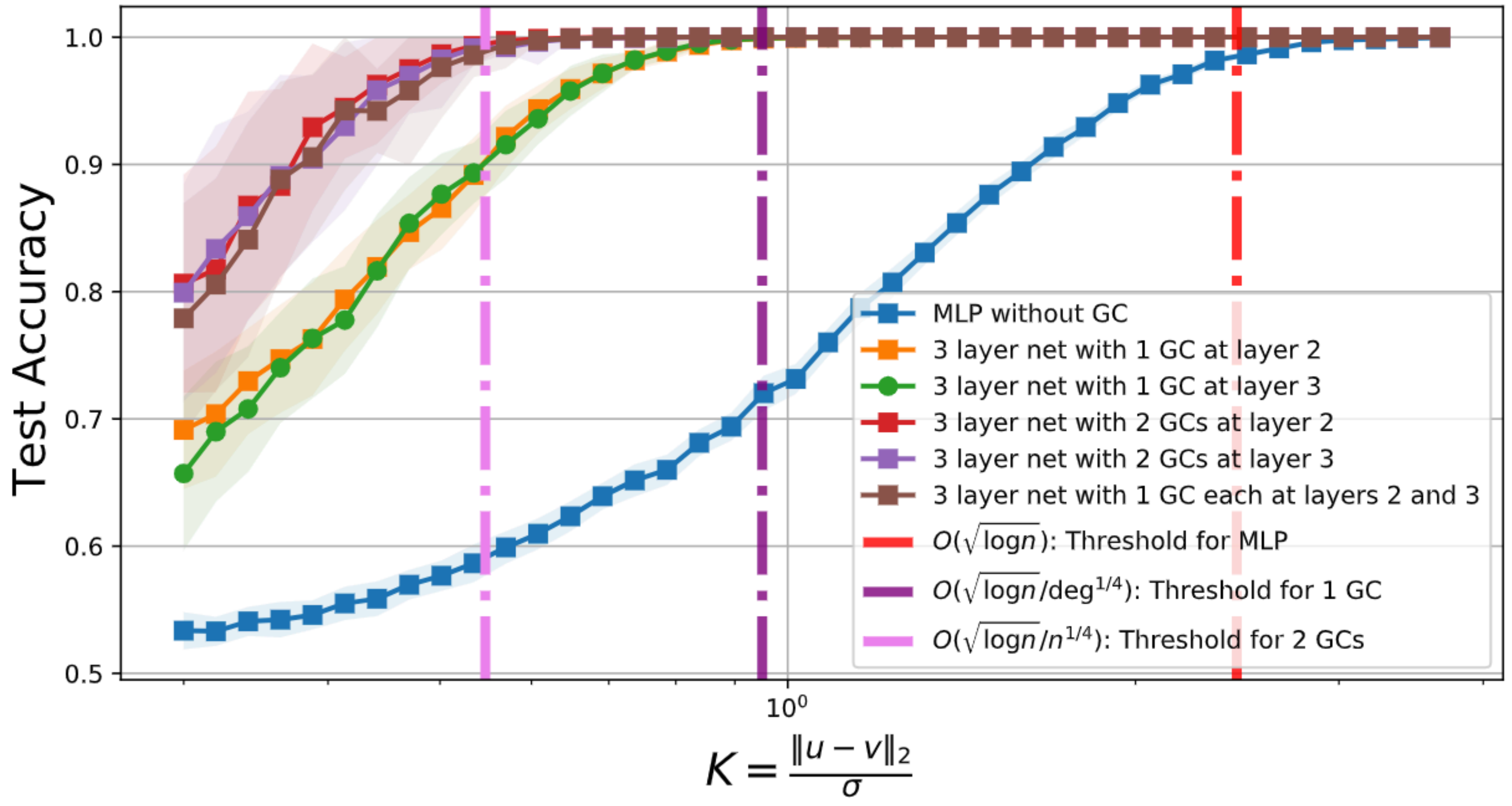
Main Result — With Graph

Number of GCs	Perfect Classification Threshold
0 (Baseline)	$\zeta = \omega(\sqrt{\log n})$
1	$\gamma \cdot \zeta = \omega\left(\sqrt{\frac{\log n}{n(p+q)}}\right)$
2	$\gamma^2 \cdot \zeta = \omega\left(\sqrt{\frac{\log n}{n}}\right)$

Main result



(a) Two-layer networks with $(p, q) = (0.2, 0.02)$.



(b) Three-layer networks with $(p, q) = (0.2, 0.02)$.

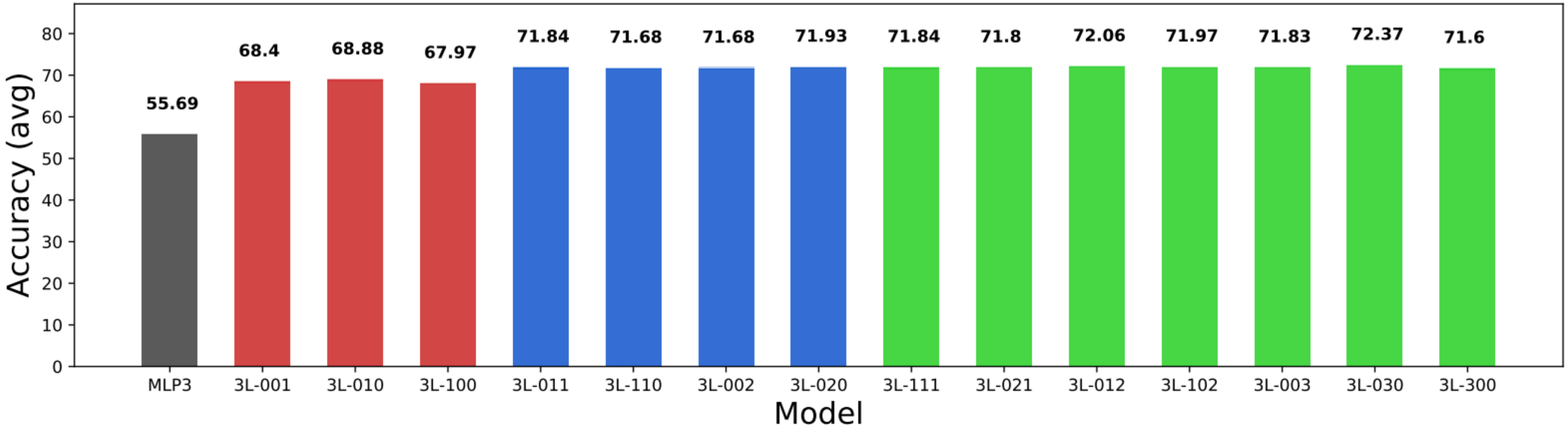
Comparison of the performance of models with 1 GC vs 2 GCs

What did we learn?

- Classification capability is determined by the number of GCs rather than the number of layers in the neural network

- Placing convolutions in any combination among the layers obtains the same result IF there are no convolutions in the first layer

Experiments on real data



3-layer models on OGBN-ARXIV

Part III

- Understanding a graph convolution operation [ICML 2021]
    - Improvement in separability threshold
    - Generalization error of the linear classifier
- Effects of graph convolutions in multilayer networks [ICLR 2023]
    - Isolate convolutions from the layers of a neural network
    - Understand effects in terms of relevant signals
- Optimality of message-passing GNNs [NeurIPS 2023]
    - Develop a notion of optimal classifier for node-classification problems
    - Design a neural network architecture that can realize the optimal classifier

Part III

- Study of node classification on sparse feature-decorated graphs on a fairly general statistical data model

- Define a notion of asymptotically local Bayes optimality

- Design a message-passing GNN that realizes the optimal classifier

- Generalization error bounds in terms of recognizable SNR

Optimal node-classification

- Require a notion of generalization error in a “per example” sense

Optimal node-classification

- Require a notion of generalization error in a “per example” sense
- Without relational information, the natural choice is Bayes risk, and the minimizer $h*$ is the Bayes optimal estimator

$$R* = \min_h \mathbb{E}_{(\mathbf{X},\mathbf{y})\sim\mathbf{P}}[L(\mathbf{y}, h(\mathbf{X}))], \qquad h* = \arg\min_h \mathbb{E}[L(\mathbf{y}, h(\mathbf{X}))]$$

Optimal node-classification

- Require a notion of generalization error in a “per example” sense
- Without relational information, the natural choice is Bayes risk, and the minimizer h^* is the Bayes optimal estimator

$$R^* = \min_h \mathbb{E}_{(\mathbf{X},\mathbf{y})\sim\mathbf{P}}[L(\mathbf{y}, h(\mathbf{X}))], \qquad h^* = \arg\min_h \mathbb{E}[L(\mathbf{y}, h(\mathbf{X}))]$$

- For graphs, size of sample = size of graph → the right extension of Bayes risk for such data is unclear

Optimal node-classification

- Require a notion of generalization error in a “per example” sense

- Without relational information, the natural choice is Bayes risk, and the minimizer h^* is the Bayes optimal estimator

$$R^* = \min_h \mathbb{E}_{(\mathbf{X},\mathbf{y})\sim\mathbf{P}}[L(\mathbf{y}, h(\mathbf{X}))], \qquad h^* = \arg\min_h \mathbb{E}[L(\mathbf{y}, h(\mathbf{X}))]$$

- For graphs, size of sample = size of graph → the right extension of Bayes risk for such data is unclear

- Example: If $h(u, G_n)$ takes node u and the graph $G_n \sim \text{CSBM}(n, \mathbf{P}, \mathbf{Q})$, the risk implicitly depends on the sample size n through G_n

Optimal node-classification

Finding an interpretable notion of optimality:

- Try the infinite sample size limit to remove dependence on n.
  But for a general class of estimators → unclear if the limit exists.
- Restrict attention to estimators that are only allowed “local” information around the nodes

Denote
$N_k(u, G) = \{v \in V(G) : \text{dist}(u, v) = k\}, \quad \eta_k(u, G) = \cup_{0 \leq j \leq k} N_j(u)$

Optimal node-classification

Definition (ℓ-local estimator)

Let $G = (\mathbf{A}, \mathbf{X})$ be a feature-decorated graph of n vertices.

For a fixed $\ell > 0$, an ℓ-local estimator is a function h that takes three inputs and predicts a classification label for each node $u \in [n]$:

$$h(u, \eta_\ell(u), \{X_v\}_{v \in \eta_\ell(u)})$$

Denote $\mathscr{C}_\ell$ to be the class of all ℓ-local estimators.

Local weak convergence

- A rooted graph (G, u) is a graph G with a distinguished vertex u, the root
- $\{(G_n, u_n)\}_{n\geq 1}$ with $G_n \sim \text{CSBM}(n, \mathbf{P}, \mathbf{Q})$ and $u_n \sim \text{Unif}([n])$
- $\{(G_n, u_n)\}_{n\geq 1} \rightsquigarrow (G, u)$, a Poisson Galton-Watson tree
- $\rightsquigarrow$ denotes *local weak convergence* [Bordenave, Ramanan, Banerjee]

Optimal node-classification

Definition (asymptotically ℓ-locally Bayes optimal estimator)

A function $h^*_\ell \in \mathscr{C}_\ell$ is the asymptotically ℓ-locally Bayes optimal estimator of the root of the sequence (G_n, u_n) if it minimizes the probability of misclassification of the root of the local weak limit (G, u), i.e.,

$$h^*_\ell = \arg\min_{h \in \mathscr{C}_\ell} \mathbb{P}[h(u, \eta_\ell(u, G), \{X_v\}_{v \in \eta_\ell(u,G)}) \neq y_u]$$

Optimal node-classification

Theorem (Optimal message-passing)

For any $\ell \geq 1$, the optimal classifier of the root for the sequence (G_n, u_n) where $G_n \sim$ CSBM$(n, \mathbf{P}, \mathbf{Q})$ is

$$h^*_\ell(u, \eta_\ell(u), \{X_v\}_{v \in \eta_\ell(u)}) = \text{arg}\max_{i \in [C]} \left\{ \log \rho_i(X_u) + \sum_{v \in \eta_\ell(u) \setminus \{u\}} M_{ik}(X_v) \right\},$$

where $k = \text{dist}(u, v)$, ρ_i is the density associated with the distribution $\mathbf{P}_i \in \mathbf{P}$, and

$$M_{ik}(\mathbf{x}) = \max_{j \in [C]} \left\{ \log \rho_j(\mathbf{x}) + \log((\mathbf{Q}^k)_{ij}) \right\}$$

What next?

- We obtained the asymptotically ℓ-locally Bayes optimal estimator for our statistical data model
- Interesting follow up questions:

What next?

- We obtained the asymptotically ℓ-locally Bayes optimal estimator for our statistical data model
- Interesting follow up questions:
  - How do we interpret this result? Generalization guarantee? Comparison with other methods? SNR analysis?

What next?

- We obtained the asymptotically ℓ-locally Bayes optimal estimator for our statistical data model

- Interesting follow up questions:
    - How do we interpret this result? Generalization guarantee? Comparison with other methods? SNR analysis?
    - Optimal on the asymptotic model. What about the finite model?

What next?

- We obtained the asymptotically ℓ-locally Bayes optimal estimator for our statistical data model
- Interesting follow up questions:
    - How do we interpret this result? Generalization guarantee? Comparison with other methods? SNR analysis?
    - Optimal on the asymptotic model. What about the finite model?
    - Is this estimator implementable as a neural network?

Interpretation

$$h^*_\ell(u, \eta_\ell(u), \{X_v\}_{v\in\eta_\ell(u)}) = \operatorname*{arg\,max}_{i\in[C]} \left\{ \sum_{v\in\eta_\ell(u)} M_{ik}(X_v) \right\}$$

$$M_{ik}(\mathbf{x}) = \max_{j\in[C]} \left\{ \log \rho_j(\mathbf{x}) + \log((\mathbf{Q}^k)_{ij}) \right\}$$

Interpretation

$$\boxed{\begin{gathered} h_\ell^*(u, \eta_\ell(u), \{X_v\}_{v\in\eta_\ell(u)}) = \arg\max_{i\in[C]} \left\{ \sum_{v\in\eta_\ell(u)} M_{ik}(X_v) \right\} \\ M_{ik}(\mathbf{x}) = \max_{j\in[C]} \left\{ \log \rho_j(\mathbf{x}) + \log((\mathbf{Q}^k)_{ij}) \right\} \end{gathered}}$$

- If $\mathbf{Q} = p\mathbf{I}$ then $h^* = \arg\max_{i\in[C]} \left\{ \sum_{v\in\eta_\ell(u)} \log \rho_i(X_v) \right\}$

  Highly informative graph, gather messages from all nodes in $\eta_\ell(u)$

Interpretation

$$\boxed{\begin{gathered} h^*_\ell(u, \eta_\ell(u), \{X_v\}_{v\in\eta_\ell(u)}) = \arg\max_{i\in[C]} \left\{ \sum_{v\in\eta_\ell(u)} M_{ik}(X_v) \right\} \\ M_{ik}(\mathbf{x}) = \max_{j\in[C]} \left\{ \log \rho_j(\mathbf{x}) + \log((\mathbf{Q}^k)_{ij}) \right\} \end{gathered}}$$

- If $\mathbf{Q} = p\mathbf{I}$ then $h^* = \arg\max_{i\in[C]} \left\{ \sum_{v\in\eta_\ell(u)} \log \rho_i(X_v) \right\}$

  Highly informative graph, gather messages from all nodes in $\eta_\ell(u)$

- If $\mathbf{Q} = p\mathbf{1}\mathbf{1}^\top$ then $h^* = \arg\max_{i\in[C]} \left\{ \log \rho_i(X_u) \right\}$

  Uninformative graph, disregard all messages from other nodes

Interpretation (2-block symmetric case)

$\mathbf{y} \in \{\pm 1\}^n, \quad \mathbf{P} = \{\mathbf{P}_-, \mathbf{P}_+\}$ with densities $\{\rho_-, \rho_+\}, \quad \mathbf{Q} = \frac{1}{n}\begin{pmatrix} a & b \\ b & a \end{pmatrix}$

Interpretation (2-block symmetric case)

$\mathbf{y} \in \{\pm 1\}^n, \quad \mathbf{P} = \{\mathbf{P}_-, \mathbf{P}_+\}$ with densities $\{\rho_-, \rho_+\}, \quad \mathbf{Q} = \frac{1}{n}\begin{pmatrix} a & b \\ b & a \end{pmatrix}$

Define $\gamma = \dfrac{a-b}{a+b}$ (signal in the graph component of the data)

Interpretation (2-block symmetric case)

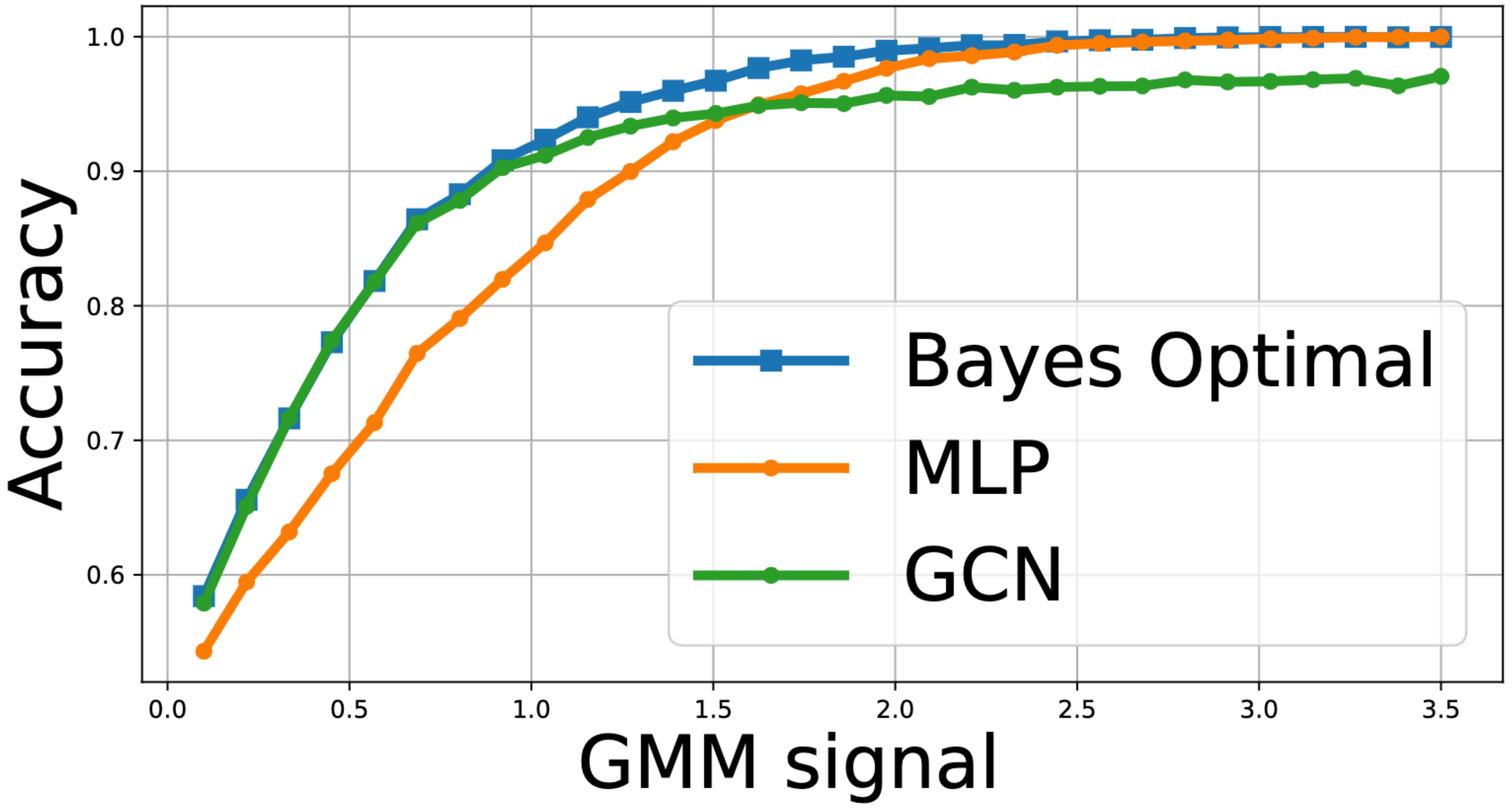
$$\mathbf{y} \in \{\pm 1\}^n, \quad \mathbf{P} = \{\mathbf{P}_-, \mathbf{P}_+\} \text{ with densities } \{\rho_-, \rho_+\}, \quad \mathbf{Q} = \frac{1}{n}\begin{pmatrix} a & b \\ b & a \end{pmatrix}$$

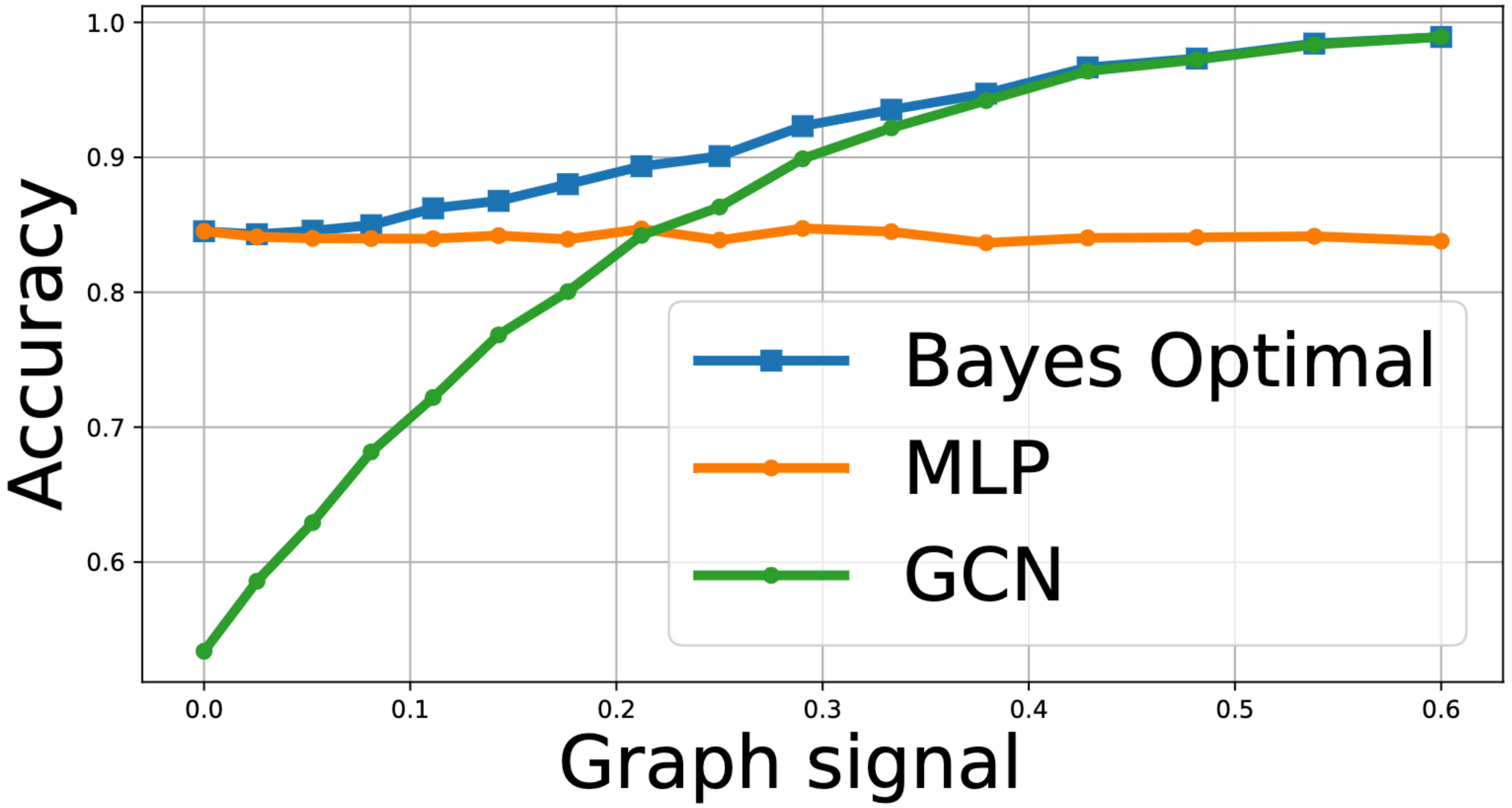
Define $\gamma = \dfrac{a-b}{a+b}$ (signal in the graph component of the data)

$$h_\ell^*(u, \{X_v\}_{v \in \eta_\ell(u)}) = \text{sgn}\Big(\psi(X_u) + \sum_{v \in \eta_\ell(u) \setminus \{u\}} M_{\text{dist}(u,v)}(X_v)\Big),$$

where $M_k(x) = \psi(x)\Big]_{-c(k)}^{c(k)}, \quad \psi(x) = \log \dfrac{\rho_+(x)}{\rho_-(x)}, \quad c(k) = \log\left(\dfrac{1+\gamma^k}{1-\gamma^k}\right)$

Interpretation (2-block symmetric case)





$$\zeta = \frac{2\|\mu\|}{\sigma}, \gamma = 0.42$$

$$\zeta = 1, \gamma = \frac{a-b}{a+b}$$

Non-asymptotic case

Asymptotically optimal estimator is still optimal for “most” nodes

Proposition (Tree neighbourhoods) [Massoulié 2014]

Let $G \sim$ CSBM$(n, \{\mathbf{P}_-, \mathbf{P}_+\}, \mathbf{Q})$ with $\mathbf{Q} = \frac{1}{n}\begin{pmatrix} a & b \\ b & a \end{pmatrix}$

For $\ell = c \log n$ with $c \log((a+b)/2) < 1/4$, w.h.p.,
$1 - o\left(\log^4 n/\sqrt{n}\right)$ fraction of nodes in G have cycle-free neighbourhoods.

Non-asymptotic case

For an estimator $h \in \mathscr{C}_\ell$, denote

- $\mathscr{E}_n(h) =$ Misclassification error on the data model with n nodes (finite n)
- $\mathscr{E}(h) =$ Misclassification error on the limiting data model ($n \to \infty$)

Recall that $\mathscr{E}(h_\ell^*) = \min_{h \in \mathscr{C}_\ell} \mathscr{E}(h)$

- How well does h_ℓ^* do on the finite data model: $\mathscr{E}_n(h_\ell^*)$
- How does it compare to the actual optimal for finite n, $\min_{h \in \mathscr{C}_\ell} \mathscr{E}_n(h)$

Non-asymptotic case

Theorem (Error for fixed n)

For any $1 \le \ell \le c \log n$ such that $c \log((a+b)/2) < 1/4$, we have

$$\mathcal{E}_n(h_\ell^*) = \min_{h \in \mathcal{C}_\ell} \mathcal{E}_n(h) \pm O\left(\frac{1}{\log^2 n}\right)$$

$$\min_{h \in \mathcal{C}_\ell} \mathcal{E}_n(h) = \mathcal{E}(h_\ell^*) \pm O\left(\frac{1}{\log^2 n}\right)$$

Remark: We can compute $\mathcal{E}(h_\ell^*)$

Implementation

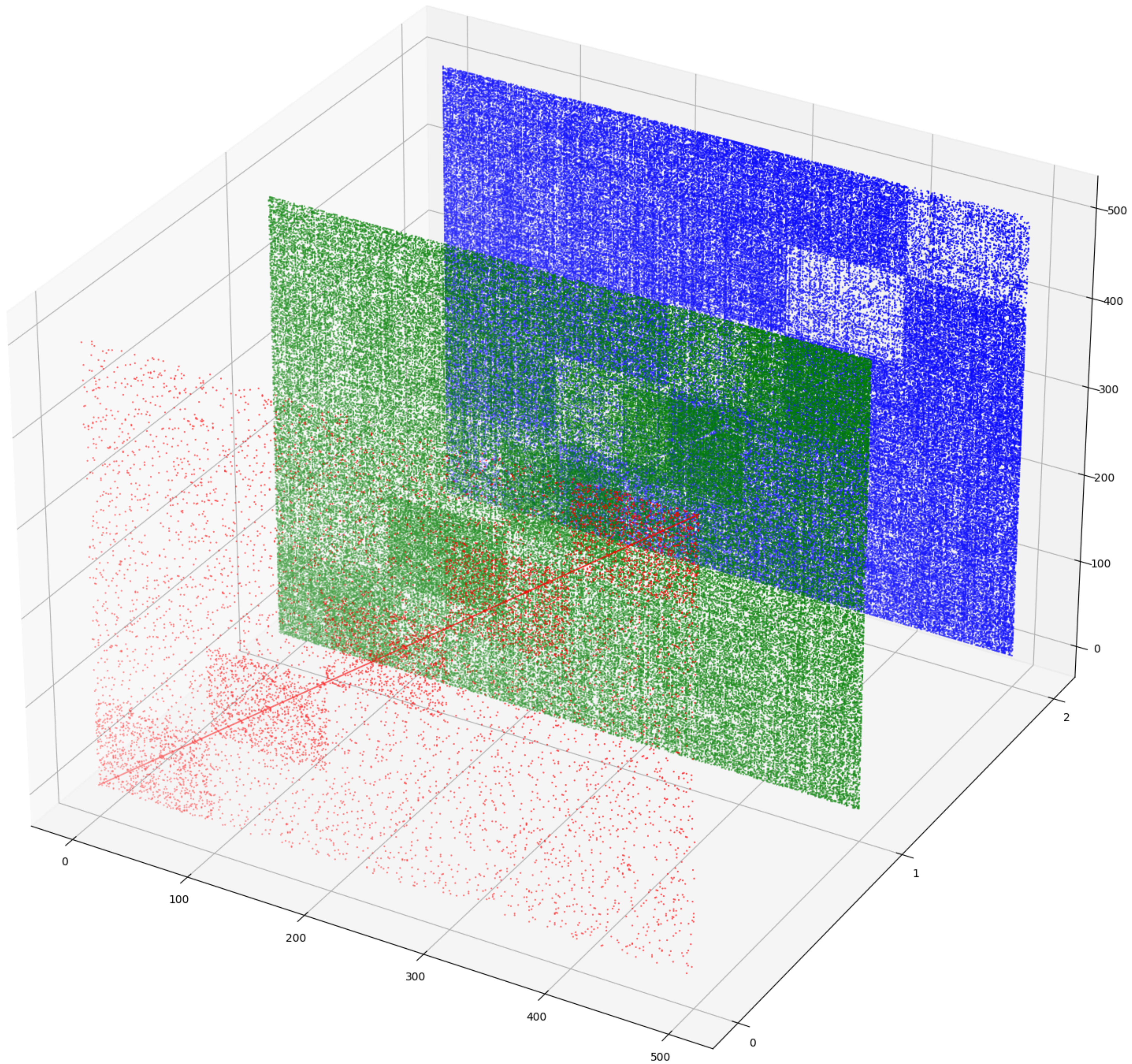
Pre-computation

$$\tilde{\mathbf{A}}^{(k)} = f(\mathbf{A}^k) \wedge \left(\neg f \left(\sum_{m=0}^{k-1} \mathbf{A}^m \right) \right) \text{ for } k \in \{1, \ldots, \ell\}$$

$f(A)$ performs entry-wise flattening $\mathbf{1}(A_{ij} > 0)$

$\tilde{\mathbf{A}}$ is an order 3 tensor, visualized as stacked multi-level adjacency matrices.

Implementation



$\tilde{\mathbf{A}}$ is visualized as stacked adjacency matrices.

Example describes 3-hop neighbourhoods for each node

Implementation

$$\mathbf{H}^{(0)} = \mathbf{X}, \qquad \mathbf{H}^{(l)} = \sigma_l(\mathbf{H}^{(l-1)}\mathbf{W}^{(l)} + \mathbf{1}_n\mathbf{b}^{(l)}) \text{ for } l \in [L],$$

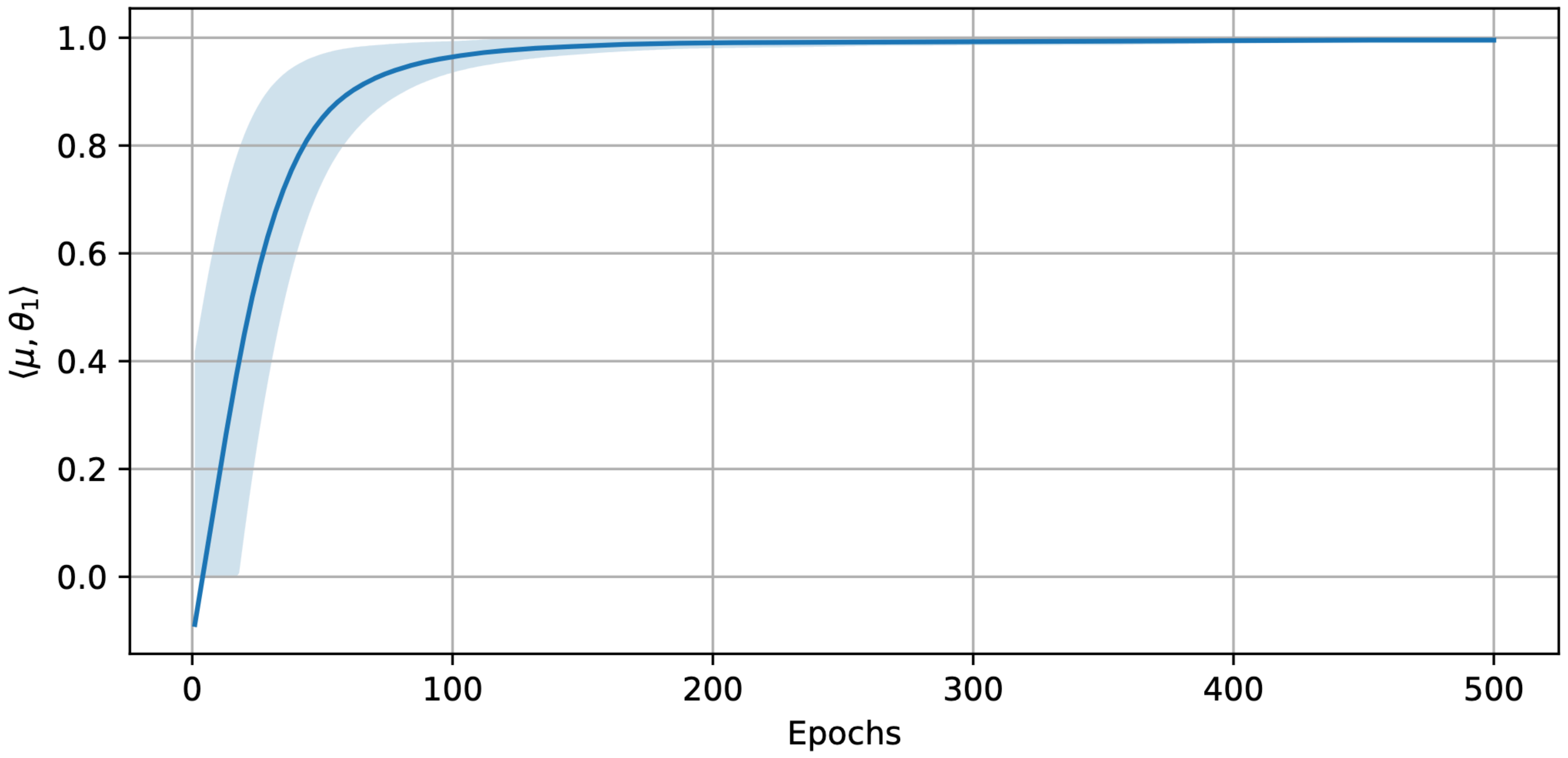
$$\mathbf{Q} = \mathrm{sigmoid}(\mathbf{Z}), \qquad \mathbf{M}^{(k)}_{u,i} = \max_{j\in[C]} \left\{\mathbf{H}^{(L)}_{u,j} + \log(\mathbf{Q}^{k}_{i,j})\right\} \text{ for } k \in [\ell], u \in [n], i \in [C].$$

Label predictions:

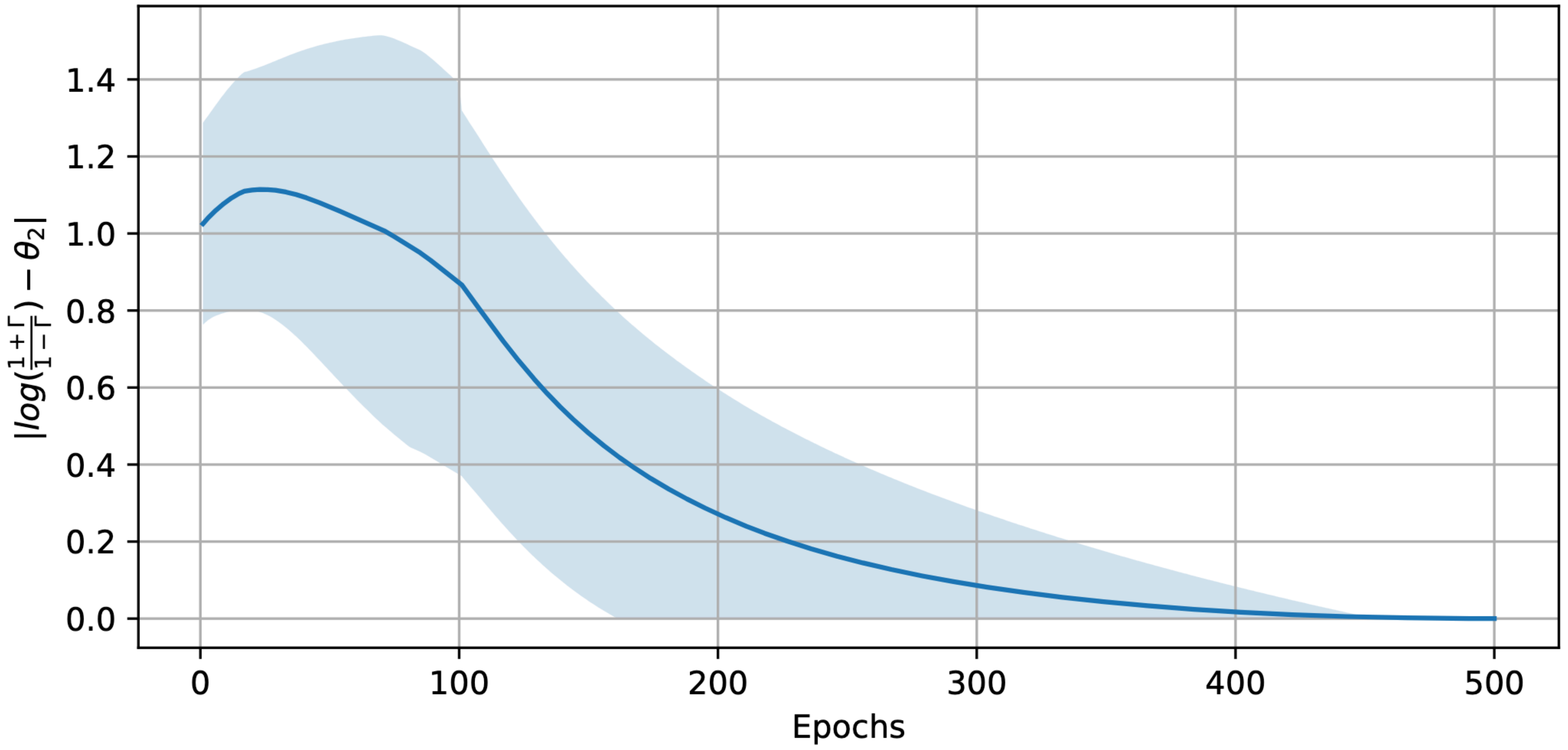
$$\hat{y}_u = \text{arg}\max_{i\in[C]} \left(\mathbf{H}^{(L)}_{u,c} + \sum_{k=1}^{\ell} \tilde{\mathbf{A}}^{(k)}_{u,:} M^{(k)}_{:,i} \right)$$

Interpretation:
Model learns the distributions $\mathbf{P}$ and the connectivity profile $\mathbf{Q}$ via $\mathbf{H}^{(L)}$ and $\mathbf{Q}$.

Convergence of parameters (training)



Weight vector



Clip threshold

Recap

- Understanding a graph convolution operation [ICML 2021]
  - Improvement in separability threshold
  - Generalization error of the linear classifier
- Effects of graph convolutions in multilayer networks [ICLR 2023]
  - Isolate convolutions from the layers of a neural network
  - Understand effects in terms of relevant signals in the data
- Optimality of message-passing GNNs [NeurIPS 2023]
  - Develop a notion of optimality for node-classification problems
  - Design a neural network architecture that can realize the optimal classifier