Generalization Error of Graph Neural Networks in the Mean-field Regime

Gholamali Aminian* [1], Yixuan He* [2], Gesine Reinert[1,2], Łukasz Szpruch [1,3], Samuel N. Cohen[1,2]
[1] The Alan Turing Institute [2] University of Oxford [3] The University of Edinburgh * Equal contribution

ICML International Conference On Machine Learning

1. Objectives

- **A novel framework for exploring the generalization errors of Graph Neural Networks, i.e., Graph Convolutional Neural (GCN) and Message Passing Graph Neural Networks (MPGNN), through functional derivative and Rademacher Complexity analyses in Mean-field regime**
- The generalization error convergence rate, when training on a sample of size n, is $\mathcal{O}(1/n)$ for **KL-regularized empirical risk minimization problem** via functional derivative
- Investigating the generalization error of one-hidden-layer graph neural network for the effect of hidden neurons.

4. Generalization Error

$$\mathrm{R}(\mathfrak{m}(\mu_n),\mu) = \underbrace{\big(\mathrm{R}(\mathfrak{m}(\mu_n),\mu) - \mathrm{R}(\mathfrak{m}(\mu_n),\mu_n)\big)}_{\text{generalization error}} + \underbrace{\mathrm{R}(\mathfrak{m}(\mu_n),\mu_n)}_{\text{training error}}.$$

- Expected Generalization Error: $\overline{\mathrm{gen}}(\mathfrak{m},\mu) \triangleq \mathbb{E}_{\mathbf{Z}_n}\big[\mathrm{R}(\mathfrak{m}(\mu_n),\mu) - \mathrm{R}(\mathfrak{m}(\mu_n),\mu_n)\big]$.
- **Replace-one sample empirical measure**: $\mu_{n,(1)} = \mu_n + \frac{1}{n}(\delta_{\widetilde{Z}_1} - \delta_{Z_1})$, where $\widetilde{Z}_1$ is i.i.d. with respect to $\mathbb{Z}_{\mathbb{N}}$
- (Aminian et al, 2023):

$$\overline{\mathrm{gen}}(\mathfrak{m},\mu) = \mathbb{E}_{\mathbf{Z}_n,\widetilde{Z}_1}\Big[\ell\big(\mathfrak{m}(\mu_n),\widetilde{Z}_1\big) - \ell\big(\mathfrak{m}(\mu_{n,(1)}),\widetilde{Z}_1\big)\Big].$$

2. Problem Formulation GCN

- $X \in \mathcal{X}$ graph sample as input and $Y \in \mathcal{Y} = \{-1, 1\}$, binary classification.
- $(X, Y) = Z \in \mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ and $Z \sim \mu$.
- Training dataset, $\mathbb{Z}_N = \{Z_i\}_{i=1}^n$ with **i.i.d. assumption,**
- **Empirical measure**, $\mu_n := \frac{1}{n}\sum_{i=1}^n \delta_{Z_i}$.
- Interested in learning a function $f_W : \mathcal{X} \to \mathcal{Y}$ parameterized via a number of parameters from $\mathcal{W}$.
- **Learning algorithm**: $\mu_n \mapsto \mathfrak{m}(\mu_n) \in \mathcal{P}(\mathcal{W})$ outputs a probability distribution (measure) on parameter space.
- loss function $(m, z) \mapsto \ell(m, z) \in \mathbb{R}^+$.
- **Risk function:** $\mathrm{R}(m,\mu) := \int_{\mathcal{Z}} \ell(m,z)\mu(\mathrm{d}z)$.
- **Empirical risk:** $\mathrm{R}(m,\mu_n) = \int_{\mathcal{Z}} \ell(m,z)\mu_n(\mathrm{d}z) = \frac{1}{n}\sum_{i=1}^n \ell(m,z_i)$.
- **KL divergence:** $\mathrm{KL}(m'\|m)$.
- **Symmetrized KL divergence:** $\mathrm{KL}_{\mathrm{sym}}(m\|m') = \mathrm{KL}(m\|m') + \mathrm{KL}(m'\|m)$.

5. KL-Regularized Risk Minimization

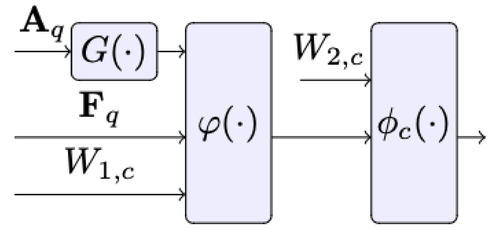
- **Setup:** $\mathcal{V}^\alpha(m,\mu) = \mathrm{R}(m,\mu) + \frac{1}{\alpha}\mathrm{KL}(m\|\pi)$
  - $R(m,\mu) = \mathbb{E}_{Z\sim\mu}[\ell(m,z)]$,
  - Prior: $\pi(w)$,
  - α: inverse temperature

Solution (Gibbs Measure):

$$\mathfrak{m}^\alpha(\mu_n) := \frac{\pi}{S_{\alpha,\pi}(\mu_n)} \exp\Big\{-\alpha\Big[\frac{\delta \mathrm{R}}{\delta m}(m,\mu_n,w)\Big]\Big\},$$

where $S_{\alpha,\pi}(\mu_n)$ is is the normalizing constant.

3. Neuron Unit and Mean-field



- **Input pair** of a graph sample with N nodes: $\mathbf{X} = (\mathbf{F}, \mathbf{A}) \in \mathcal{X}$ where $\mathbf{F}$ is nodes feature matrix and $\mathbf{A}$ is graph adjacency matrix, $d_{\max}$ and $d_{\min}$: maximum and minimum node degrees among all graph samples.
- **Parameters:** $W_{1,c}$ the parameters of each neuron where $W_{1,c} \in \mathcal{S}^k \subset \mathbb{R}^k$ and $W_{2,c} \in \mathcal{S} \subset \mathbb{R}$.
- **Activation Function:** $\varphi(W_{1,c} \cdot x)$.
- **Neuron Unit:** $\phi(W_c, x) = W_{2,c}(i)\varphi(W_{1,c}(i) \cdot X(j))$.
- **Number of Neuron Units:** h.
- **Empirical parameter measure:** $m_h := \frac{1}{h}\sum_{i=1}^h \delta_{(W_{1,c}(i),W_{2,c}(i))}$.
- **Readout function:** Mean-readout $\Psi(m_h^c(\mu_n),\mathbf{X}) := \frac{1}{N}\sum_{j=1}^N \mathbb{E}_{W_c\sim m_h^c(\mu_n)}\big[\phi_c(W_c, G(\mathbf{A})[j,:]\mathbf{F})\big]$.
- **Prediction:** $\hat{y} := \hat{f}(\mathbf{X}) = \frac{1}{h}\sum_{i=1}^h \phi_c(W_c(i),\mathbf{X}) = \int \phi(W_c,\mathbf{X})\, m_h(\mathrm{d}W_c) = \mathbb{E}_{W_c\sim m_h}[\phi(W_c,\mathbf{X})]$.
- **Mean-field:** $h \to \infty$ then $m_h(\mu_n) \to m(\mu_n)$.
- **Logistic loss:** $\ell(\Psi(m(\mu_n),\mathbf{x}),y) = \log(1+\exp(-\Psi(m(\mu_n),\mathbf{x})y))$.

6. Assumptions

1. Bounded loss function, $0 \le \ell(\hat{y}, y) \le M_\ell$.
2. Bounded gradient of loss function, $|\partial_{\hat{y}}\ell(\hat{y}, y)| \le M_{\ell'}$.
3. Convex loss function, $\ell(\hat{y}, y)$ with respect to $\hat{y}$.
4. Bounded Neuron unit function, $|\phi_c(.,.)| \le M_\phi$.
5. Bounded node features, $\|F[i,:]\| \le B_f$.

7. Main Results (Under Assumptions) and Comparison

Proposition 1 (Upper Bound) *There exists constant C, such that*

$$\overline{\mathrm{gen}}(\mathrm{m}(\mu_n),\mu) \le C\mathbb{E}_{\mathbf{Z}_n,\widetilde{Z}_1}\Big[\sqrt{\mathrm{KL}\big(\mathrm{m}(\mu_n)\|\mathrm{m}(\mu_{n,(1)})\big)}\Big],$$

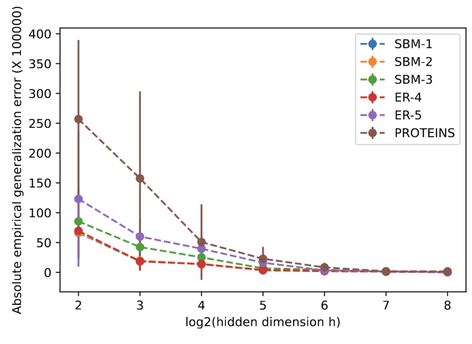
Proposition 2 (Lower Bound) *For Gibbs measure, we have,*

$$\overline{\mathrm{gen}}(\mathrm{m}^\alpha(\mu_n),\mu) \ge \frac{n}{2\alpha}\mathbb{E}_{\mathbf{Z}_n,\widetilde{Z}_1}\Big[\mathrm{KL}_{\mathrm{sym}}\big(\mathrm{m}^\alpha(\mu_n)\|\mathrm{m}^\alpha(\mu_{n,(1)})\big)\Big].$$

Theorem 1 *There exists constant C, such that* $\overline{\mathrm{gen}}(\mathrm{m}^\alpha(\mu_n),\mu) \le \frac{\alpha C}{n}$.

Approach	$\tilde{d}_{\max}, \tilde{d}_{\min}$	Width of GCN (h)	Number of graph samples (n)	Bound Type
VC-Dimension (Scarselli et al., 2018)	N/A	$O(h^4)$	$O(1/\sqrt{n})$	HP
Rademacher Complexity (Garg et al., 2020)	$O(\tilde{d}_{\max}\log^{1/2}(\tilde{d}_{\max}))$	$O(h\sqrt{\log(h)})$	$O(1/\sqrt{n})$	HP
PAC-Bayesian (Liao et al., 2020)	$O(\tilde{d}_{\max})$	$O(\sqrt{h\log(h)})$	$O(1/\sqrt{n})$	HP
PAC-Bayesian (Ju et al., 2023)	N/A	$O(\sqrt{h})$	$O(1/\sqrt{n})$	HP
Continuous MPGNN (Maskey et al., 2022)	N/A	N/A	$O(1/\sqrt{n})$	P
Rademacher Complexity (this paper)	$O((\tilde{d}_{\max}/\tilde{d}_{\min})^{3/4})$	N/A	$O(1/\sqrt{n})$	HP
Functional Derivative (this paper)	$O(\tilde{d}_{\max}/\tilde{d}_{\min})$	**N/A**	$O(1/n)$	E

8. Experiments



- We investigate the effect of the number of hidden neurons (number of hidden units) h on the true generalization error of GCNs
- Stochastic Block Models (SBMs), Erdos-Rényi (ER) models, and PROTEINS dataset
- As the value of h increases, the absolute generalization error decreases. This observation shows that the upper bounds dependent on the width of the layer fail to capture the trend of generalization error in the over-parameterized regime.

References

- Aminian et al. (2023). Mean-field analysis of generalization errors. ArXiv:2306.11623
- Scarselli et al. (2018). The vapnik–chervonenkis dimension of graph and recursive neural networks. Neural Networks.
- Garg et al. (2020). Generalization and representational limits of graph neural networks. In ICML.
- Liao et al. (2020). A pac-bayesian approach to generalization bounds for graph neural networks. In Neurips.
- Ju et al. (2023). Generalization in graph neural networks: Improved pac-bayesian bounds on graph diffusion. In AISTATS.
- Maskey et al. (2022). Generalization analysis of message passing neural networks on large random graphs. In NeurIPS.

Useful Links

ArXiv ID: 2402.07025
Code repo: https://github.com/SherylHYX/GNN_MF_GE
Email: gaminian@turing.ac.uk

arXiv link code link

Acknowledgements

- The UKRI Prosperity Partnership Scheme (FAIR)
- EPSRC Grants EP/V056883/1, EP/W037211/1 and EP/R018472/1
- The Alan Turing Institute and G-Research