Graph Neural Networks Do Not Always Oversmooth

Bastian Epping[1], **Alexandre René**[1], **Moritz Helias**[2,3], **Michael T. Schaub**[1]
[1]RWTH Aachen University, Aachen, Germany
[2]Department of Physics, RWTH Aachen University, Aachen, Germany
[3]Institute for Advanced Simulation (IAS-6), Computational and Systems Neuroscience, Jülich Research Centre, Jülich, Germany
epping@cs.rwth-aachen.de, rene@cs.rwth-aachen.de,
m.helias@fz-juelich.de, schaub@cs.rwth-aachen.de

Abstract

Graph neural networks (GNNs) have emerged as powerful tools for processing relational data in applications. However, GNNs suffer from the problem of oversmoothing, the property that features of all nodes exponentially converge to the same vector over layers, prohibiting the design of deep GNNs. In this work we study oversmoothing in graph convolutional networks (GCNs) by using their Gaussian process (GP) equivalence in the limit of infinitely many hidden features. By generalizing methods from conventional deep neural networks (DNNs), we can describe the distribution of features at the output layer of deep GCNs in terms of a GP: as expected, we find that typical parameter choices from the literature lead to oversmoothing. The theory, however, allows us to identify a new, non-oversmoothing phase: if the initial weights of the network have sufficiently large variance, GCNs *do not* oversmooth, and node features remain informative even at large depth. We demonstrate the validity of this prediction in finite-size GCNs by training a linear classifier on their output. Moreover, using the linearization of the GCN GP, we generalize the concept of propagation depth of information from DNNs to GCNs. This propagation depth diverges at the transition between the oversmoothing and non-oversmoothing phase. We test the predictions of our approach and find good agreement with finite-size GCNs. Initializing GCNs near the transition to the non-oversmoothing phase, we obtain networks which are both deep and expressive.

1 Introduction

Graph neural networks (GNNs) reach state of the art performance in diverse application domains with relational data that can be represented on a graph, transferring the success of machine learning to data on graphs [47, 12, 23, 7]. Despite their good performance, GNNs come with the limitation of *oversmoothing*, a phenomenon where node features converge to the same state exponentially fast for increasing depth [35, 45, 30, 2]. Consequently, only shallow networks are used in practice [19, 1]. In contrast, it is known that the depth (i.e. the number of layers) is key to the success of deep neural networks (DNNs) [32, 33]. While for conventional DNNs shallow networks are proven to be highly expressive [6], in practice deep networks are much easier to train and are thus the commonly used architectures [36]. Furthermore, in most GNN architectures each layer only exchanges information between neighboring nodes. Deep GNNs are therefore necessary to exchange information between nodes that are far apart in the graph [9]. In this study, we investigate oversmoothing in graph convolutional networks (GCNs) [19].

To study the effect of depth, we consider the propagation of features through the network: given some input $\boldsymbol{x}_\alpha^{(0)}$, each intermediate layer l produces features $\boldsymbol{x}_\alpha^{(l)}$ which are fed to the next layer. We

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

follow the same approach that has successfully been employed in previous work to design trainable DNNs [37]: consider two nearly identical inputs $\boldsymbol{x}_\alpha^{(0)}$ and $\boldsymbol{x}_\beta^{(0)}$ and ask whether the intermediate features $\boldsymbol{x}_\alpha^{(l)}$ and $\boldsymbol{x}_\beta^{(l)}$ become more or less similar as a function of depth l. In the former case, the inputs may eventually become indistinguishable. In the latter case, the inputs become less similar over layers: the distance between them increases over layers [32, 37] until eventually it is bounded by the non-linearities of the network. The distance then typically converges to a fixed value determined by the network architecture, independent of the inputs.

One can therefore identify two phases: One says that a network is *regular* if two inputs eventually converge to the same value as function of l; conversely, one says that a network is *chaotic* if two inputs remain distinct for all depths [24]. Neither phase is ideal for training deep networks since in both cases all the information from the inputs is eventually lost; the typical depth at which this happens is called the *information propagation depth*. However, this propagation depth diverges at the transition between the two phases, allowing information – in principle – to propagate infinitely deep into the network. While these results are calculated in the limit of infinitely many features in each hidden layer, the information propagation depth has been found to be a good indicator of how deep a network can be trained [37]. A usual approach for conventional DNNs is thus to initialize them at the transition to chaos. Indeed, Schoenholz et al. [37] were able to use this approach to train fully-connected, feedforward networks with hundreds of layers. Similar methods have recently been adapted to the study of transformers, successfully predicting the best hyperparameters for training [5].

In this work we address the oversmoothing problem of GCNs by extending the framework described above in the limit of infinite feature dimensions from DNNs to GCNs: here the two different inputs $\boldsymbol{x}_\alpha^{(0)}$ and $\boldsymbol{x}_\beta^{(0)}$ correspond to the input features on two nodes, labeled α and β. The mixing of information across different nodes implies that output features on node α depend on input features on node β and vice versa. Thus it is not possible to look at the distance of $\boldsymbol{x}_\alpha^{(l)}$ and $\boldsymbol{x}_\beta^{(l)}$ independently for each pair α and β as in the DNN case. Rather, one has to solve for the distance between each distinct pair of nodes in the graph simultaneously: the one dimensional problem for DNNs thus becomes a multidimensional problem for GCNs. However, by linearizing the multidimensional GCN dynamics, we can generalize the notion of information propagation depth to GCNs: instead of being a single value, we find that a given GCN architecture comes with a set of potentially different information propagation depths, each corresponding to one eigendirection of the linearized dynamics of the system.

This approach allows us to extend the concept of a regular and a chaotic phase to GCNs: in the regular phase, which describes most of the GCNs studied in the current literature, distances between node features shrink over layers and exponentially attain the same value. We therefore call this the oversmoothing phase. On the other hand, if one increases the variance of the weights at initialization, it is possible to transition into the chaotic phase. In this phase, distances between node features converge to a fixed but finite distance at infinite depth. The convergence point is fully determined by the underlying graph structure and the hyperparameters of the GCN and may differ for different pairs of nodes. GCNs initialized in this phase thus do not suffer from oversmoothing. We find that the convergence point is informative about the topology of the underlying graph and may be used for node classification with GCNs of more than $1,000$ layers. Near the transition point, GCNs at large depth offer a trade-off between feature information and information contained in the neighborhood relation of the graph. We test the predictions of this theory and find good agreement in comparison to finite-size GCNs applied to the contextual stochastic block model [8]. On the citation network Cora [19] we reach the performance reported in the original work by Kipf and Welling [19] beyond 100 layers. Our approach applies to graphs with arbitrary topologies, depths, and non-linearities.

2 Related Work

Oversmoothing is a well-known challenge within the GNN literature [21, 35]. On the theoretical side, rigorous techniques have been used to prove that oversmoothing is inevitable. The authors in [30] show that GCNs with the ReLU non-linearity exponentially lose expressive power and only carry information of the node degree in the infinite layer limit. While the authors notice that their upper bound for oversmoothing does not hold for large singular values of the weight matrices, they do not

identify a non-oversmoothing phase in their model. These results have been extended in [2] to handle non-linearities different from ReLU. Also graph attention networks have been proven to oversmooth inevitably [45]. Their proof, however, makes assumptions on the weight matrices which, as we will show, exclude networks in the chaotic, non-oversmoothing phase.

On the applied side, a variety of heuristics have been developed to mitigate oversmoothing [49, 4, 3, 22, 40, 15]. E.g., the authors in [49] introduce a normalization layer which can be added to a variety of deep GNNs to make them trainable. Another approach is to introduce residual connections and identity mappings, directly feeding the input to layers deep in the network [4, 46]. Other studies suggest to train GNNs to a limited number of layers to obtain the optimal amount of smoothing [18, 46]. The recent review [17] proposes a unified view to order existing heuristics and guide further research. While these heuristics improve performance at large depths, they also add to the complexity of the model and impose design choices. Our approach, on the other hand, explains why increasing the weight variance at initialization is sufficient to prevent oversmoothing.

3 Background

3.1 Network architecture

In this paper we study a standard graph convolutional network (GCN) architecture [19] with an input feature matrix $\boldsymbol{X}^{(0)} \in \mathbb{R}^{N \times d_0}$, where N is the number of nodes in the graph and d_0 the number of input features. Bold symbols throughout represent vector or matrix quantities in feature space. The structure of the graph is represented by a shift operator $A \in \mathbb{R}^{N \times N}$. We write the features of the network's l-th layer as $\boldsymbol{X}^{(l)} \in \mathbb{R}^{N \times d_l}$; they are computed recursively as

$$\boldsymbol{X}^{(l)} = \phi(A\boldsymbol{X}^{(l-1)}\boldsymbol{W}^{(l)\top} + 1\boldsymbol{b}^{(l)\top}) \,, \tag{1}$$

with ϕ an elementwise non-linear activation function, $\boldsymbol{b}^{(l)}$ an optional bias term, weight matrices $\boldsymbol{W}^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$, and $1 \in \mathbb{R}^N$ a vector of all ones. We note that many GNN architectures studied in the literature are unbiased, which can be recovered by setting $\boldsymbol{b}^{(l)}$ to zero. We use a noisy linear readout, so that the output of the network is given by

$$\boldsymbol{Y} = A\boldsymbol{X}^{(L)}\boldsymbol{W}^{(L+1)\top} + 1\boldsymbol{b}^{(L+1)\top} + \boldsymbol{\epsilon} \,, \tag{2}$$

with $\boldsymbol{\epsilon} \in \mathbb{R}^{N \times d_{L+1}}$ being independent Gaussian random variables: $\epsilon_{\alpha,i} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_{ro}^2)$. The readout noise $\boldsymbol{\epsilon}$ is included both to promote robust outputs and to prevent numerical issues in the matrix inversion in Equations (11) and (12). We use d_{L+1} to denote the dimension of outputs.

For the following it will be useful to consider the activity of individual nodes. To avoid ambiguity in the indexing, we use lower Greek indices for nodes and upper Latin indices for layers. We thus rewrite (1) as

$$\boldsymbol{x}_\alpha^{(l)} = \phi(\boldsymbol{h}_\alpha^{(l)}) \,, \tag{3}$$

$$\boldsymbol{h}_\alpha^{(l)} = \sum_\beta A_{\alpha\beta}\boldsymbol{W}^{(l)}\boldsymbol{x}_\beta^{(l-1)} + \boldsymbol{b}^{(l)} \,, \tag{4}$$

$$\boldsymbol{y}_\alpha = \boldsymbol{h}_\alpha^{(L+1)} + \boldsymbol{\epsilon}_\alpha \,, \tag{5}$$

where $\boldsymbol{x}_\alpha^{(l)} \in \mathbb{R}^{d_l}$ is the feature vector of node α in layer l and $\boldsymbol{y}_\alpha \in \mathbb{R}^{d_{L+1}}$ the network output for node α. The values $\boldsymbol{h}_\alpha^{(l)} \in \mathbb{R}^{d_l}$ are linear functions of the features $\boldsymbol{x}_\beta^{(l-1)}$ and represent the input to the activation functions; we therefore refer to them as preactivations. The non-linearity $\phi(x)$ is applied elementwise to the preactivations $\boldsymbol{h}_\alpha^{(l)}$. While we leave $\phi(x)$ general for the development of the theory, we use $\phi(x) = \operatorname{erf}(\frac{\sqrt{\pi}}{2}x)$ for the experiments in Section 4; this choice allows us to carry out certain integrals analytically. The scaling factor in the erf is chosen such that $\frac{\partial\phi}{\partial x}(0) = 1$. We use independent and identical Gaussian priors for all weight matrices and biases, $W_{ij}^{(l)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \frac{\sigma_w^2}{d_l})$ and $b_i^{(l)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_b^2)$ with $W_{ij}^{(l)}$ and $b_i^{(l)}$ being the matrix or vector entries of $\boldsymbol{W}^{(l)}$ and $\boldsymbol{b}^{(l)}$, respectively. As a shift operator, we choose

$$A = \mathbb{I} - \frac{g}{d_{\max}}(D - \mathcal{A}) \,, \tag{6}$$

where $\mathcal{A}$ is the adjacency matrix, $\mathbb{I}$ the identity in $\mathbb{R}^{N\times N}$, $D_{\alpha\beta} = \delta_{\alpha\beta}\sum_\gamma \mathcal{A}_{\alpha\gamma}$ is the degree matrix and $d_{\max}$ is the maximal degree. The parameter $g \in (0, 1)$ allows us to weigh the off-diagonal elements compared to the diagonal ones. By construction the shift operator is row-stochastic, which means that it has constant sums over columns $\sum_\beta A_{\alpha\beta} = 1$. We will make use of this property in our analysis in Section 4.2. The generalization to non-stochastic shift operators will be shortly addressed later.

3.2 Gaussian process equivalence of GCNs

In a classic machine learning setting, such as classification, one draws random initial values for all parameters and subsequently trains the parameters by optimizing the weights and biases to minimize a loss function. This learned parameter set is then used to classify unlabeled inputs. In this paper we take a Bayesian point of view in which the network parameters are random variables, inducing a probability distribution over outputs which becomes Gaussian in the limit of infinitely many features. Thus infinitely wide neural networks are equivalent to Gaussian processes (GPs) [27, 43, 34]. In the study of DNNs this is a standard approach, yielding results which empirically hold also for finite-size networks trained with gradient descent [37].

In previous work [28, 14, 29] it has been shown that also the GCN architecture described in Section 3.1 is equivalent to a GP in the limit of infinite feature space dimensions, $d_l \to \infty$ for all hidden layers $l = 1, \ldots, L$, while input and readout layer still have tunable, finitely many features. In the GP description, all features are Gaussian random variables with zero mean and identical prior variance in each feature dimension. The description of the GCN thus reduces to a multivariate normal,

$$H^{(l)} \sim \mathcal{N}(0, K^{(l)})\,, \tag{7}$$

where $H^{(l)}$ is the vector of hidden node features of layer l, $H^{(l)} = (h_0^{(l)}, h_1^{(l)}, \ldots, h_N^{(l)})^\top$ under the prior distribution of weights and biases. The covariance matrices $K^{(l)} \in \mathbb{R}^{N\times N}$ are determined recursively: knowing that the $h_\alpha^{(l)}$ follow a zero-mean Gaussian with covariance $\langle h_\delta^{(l)} h_\gamma^{(l)}\rangle = K_{\delta\gamma}^{(l)}$, we define

$$C_{\gamma\delta}^{(l)} = \left\langle \phi\Big(h_\gamma^{(l)}\Big)\phi\Big(h_\delta^{(l)}\Big)\right\rangle_{h_\gamma^{(l)}, h_\delta^{(l)}}\,. \tag{8}$$

For simplicity we use $\phi(x) = \mathrm{erf}(\frac{\sqrt{\pi}}{2}x)$ for which Equation (8) can be evaluated analytically; see Appendix A for details. It follows from (3) that

$$K_{\alpha\beta}^{(l+1)} = \sigma_b^2 + \sigma_w^2 \sum_{\gamma,\delta} A_{\alpha\gamma} A_{\beta\delta} C_{\gamma\delta}^{(l)}\,, \tag{9}$$

as shown in [28, 29].

In a semi-supervised node classification setting, we split the underlying graph into N^{test} unlabeled test nodes and N^{train} labeled training nodes ($N^{\mathrm{test}} + N^{\mathrm{train}} = N$); we correspondingly split the output random variable $Y_i \in \mathbb{R}^N$ for output dimension i into $Y_i^\star \in \mathbb{R}^{N^{\mathrm{test}}}$ and $Y_i^D \in \mathbb{R}^{N^{\mathrm{train}}}$. Features on the test nodes are predicted by conditioning on the values of the training nodes: $p\big(Y_i^* = y_i^* \mid Y_i^D = y_i^D\big)$. This leads to the following posterior for the unobserved labels (see [34, 20] for details):

$$Y_i^\star \sim \mathcal{N}(m_i^{\mathrm{GP}}, K^{\mathrm{GP}})\,, \tag{10}$$

$$m_i^{\mathrm{GP}} = K_{\star D}^{(L+1)}(K_{DD}^{(L+1)} + \mathbb{I}\sigma_{ro}^2)^{-1} Y_i^D\,, \tag{11}$$

$$K^{\mathrm{GP}} = K_{\star\star}^{(L+1)} - K_{\star D}^{(L+1)}(K_{DD}^{(L+1)} + \mathbb{I}\sigma_{ro}^2)^{-1}(K_{\star D}^{(L+1)})^\top\,. \tag{12}$$

Here the $\star$ and D indices represent test and training data, respectively, i.e. $K_{DD} \in \mathbb{R}^{N^{\mathrm{train}}\times N^{\mathrm{train}}}$ is the covariance matrix of outputs of all training nodes and $K_{\star D} \in \mathbb{R}^{N^{\mathrm{test}}\times N^{\mathrm{train}}}$ is the covariance between test data and training data. Finally, $\mathbb{I}$ is here the identity in $\mathbb{R}^{N^{\mathrm{train}}\times N^{\mathrm{train}}}$.

3.3 Feature distance

To measure and quantify how much a given GCN instance oversmoothes we use the squared Euclidean distance between pairs of nodes, and normalize by the number of node features d_l so that the measure

stays finite in the GP limit $d_l \to \infty$. This allows us to quantitatively test the predictions of our approach on the node-resolved distances of features. To summarize the amount of oversmoothing across the GCN, we also define the measure $\mu(X)$ as the average squared Euclidean distance across all pairs of nodes:

$$d(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta) = \frac{1}{d_l}||\boldsymbol{x}_\alpha - \boldsymbol{x}_\beta||_2^2 = C'_{\alpha\alpha} + C'_{\beta\beta} - 2C'_{\alpha\beta}\,, \tag{13}$$

$$\mu(\boldsymbol{X}) = \frac{1}{2N(N-1)} \sum_{\alpha=1}^{N} \sum_{\beta=\alpha+1}^{N} d(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)\,. \tag{14}$$

Here $C'_{\alpha\beta} = \frac{\boldsymbol{x}_\alpha \cdot \boldsymbol{x}_\beta}{d_l}$ is the normalized scalar product. We use the notation $C'_{\alpha\beta}$ to avoid confusion with the expectation value $C_{\alpha\beta}$ defined in the GCN GP (8). In the infinite feature dimensions limit, the quantities $C'_{\alpha\beta}$ in Equation (14) converge to the GCN GP quantities $C_{\alpha\beta}$ defined by (8). In the following sections we will therefore use the $C_{\alpha\beta}$ as predictions for the $C'_{\alpha\beta}$ of finite-size GCNs. The normalization for $d(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ and $\mu(\boldsymbol{X})$ can be interpreted as an average (squared) feature distance, independent of the size of the graph and the number of feature dimensions.

4 Results

4.1 Propagation depths

We are interested in analyzing GCNs at large depth. We a priori assume that at infinite depth the GCN converges to an equilibrium in which covariances are static over layers $K^{(l)}_{\alpha\beta} \xrightarrow{l\to\infty} K^{\mathrm{eq}}_{\alpha\beta}$, irrespective of whether the GCN is in the oversmoothing or the chaotic phase. A posteriori we show that this assumption indeed holds. Since the fixed point K^{eq} is independent of the input, a GCN at equilibrium cannot use information from the input to make predictions (although, as we will see, in the non-oversmoothing phase it can still use the graph structure). In the following we analyze the equilibrium covariance K^{eq} to which GCNs with different σ_w^2, σ_b^2 and A converge to, how they behave near this equilibrium, and at which rate it is approached.

Close to equilibrium, the covariance matrix $K^{(l)}$ can be written as a perturbation around $K^{\mathrm{eq}}_{\alpha\beta}$:

$$K^{(l)}_{\alpha\beta} = K^{\mathrm{eq}}_{\alpha\beta} + \Delta^{(l)}_{\alpha\beta}\,. \tag{15}$$

Under the assumption that the perturbation $\Delta^{(l)}_{\alpha\beta}$ is small, we can linearize the GCN GP

$$\Delta^{(l+1)}_{\alpha\beta} = \sum_{\gamma,\delta} H_{\alpha\beta,\gamma\delta}\Delta^{(l)}_{\gamma\delta} + \mathcal{O}((\Delta^{(l)})^2)\,, \tag{16}$$

$$H_{\alpha\beta,\gamma\delta} = \sigma_w^2 \sum_{\theta,\phi} \frac{1}{2}(1+\delta_{\gamma,\delta}) A_{\alpha\theta} A_{\beta\phi} \frac{\partial C_{\theta\phi}}{\partial K_{\gamma\delta}}[K^{\mathrm{eq}}]\,, \tag{17}$$

where we use square brackets to denote the point around which we linearize. The factor $\frac{1}{2}(1+\delta_{\gamma,\delta})$ is introduced to correctly count on- and off-diagonal elements of the covariance matrix, while the shift operators A and the derivative $\frac{\partial C_{\theta\phi}}{\partial K_{\gamma\delta}}[K^{\mathrm{eq}}]$ originate from the message passing and the non-linearity ϕ, respectively. The latter would result in a Kronecker delta $\frac{\partial C_{\theta\phi}}{\partial K_{\gamma\delta}}[K^{\mathrm{eq}}] = \delta_{\theta\phi,\gamma\delta}$ for linear networks. The calculation for $H_{\alpha\beta,\gamma\delta}$ is done in detail in Appendix B.

A conceptually similar linearization has been done in [37] for DNNs. In the DNN case, different inputs to the networks—which correspond to input features on different nodes here—can be treated separately, leading to decoupling of Equation (16). The shift operator in the GCN dynamics, in contrast, couples features on neighboring nodes – the matrix $H_{\alpha\beta,\gamma\delta}$ is in general not diagonal.

We can still achieve a decoupling by interpreting Equation (16) as a matrix multiplication, if $\alpha\beta$ and $\gamma\delta$ are understood as double indices, and by finding the eigendirections of the matrix $H \in \mathbb{R}^{N^2\times N^2}$. Taking the right eigenvectors $V^{(i)}_{\alpha\beta}$ as basis vectors, we can decompose the covariance matrix $\Delta^{(l)}_{\alpha\beta} = \sum_i \Delta^{(l)}_i V^{(i)}_{\alpha\beta}$ and thus obtain the overlaps $\Delta^{(l)}_i$ which evolve independently over layers. If the fixed

point K^{eq} is attractive, all eigenvalues have absolute values smaller than one: $|\lambda_i| < 1$. This allows us to define the propagation depth $\xi_i \coloneqq -\frac{1}{\ln(\lambda_i)}$ for each eigendirection, very similar to the DNN case [37]. In this form, the linear update equation (16) simplifies to

$$\Delta_i^{(l+d)} = \lambda_i^d \Delta_i^{(l)} = \exp(-d/\xi_i)\Delta_i^{(l)}\,, \tag{18}$$

thus decoupling the system. For details on the linearization and some properties of the transition matrix H refer to Appendix B.

4.2 The non-oversmoothing phase of GCNs

In this section we establish the chaotic, non-oversmoothing phase of GCNs, and show that this phase can be reached by simple tuning of the weight variance σ_w^2 at initialization. We start by noticing that a GCN is at a state of zero feature distance $\mu(\boldsymbol{X}^{(l)}) = 0$, if the covariance matrix has constant entries, $K_{\alpha\beta}^{(l)} = k^{(l)}$: Constant entries in $K_{\alpha\beta}^{(l)}$ imply that all preactivations are the same, $\boldsymbol{h}_\alpha^{(l)} = \boldsymbol{h}_\beta^{(l)}$, which in turn implies $C_{\alpha\beta}^{(l)} = c^{(l)}$ (by Equation (8)); the latter is equivalent to features being the same, $\boldsymbol{x}_\alpha^{(l)} = \boldsymbol{x}_\beta^{(l)}$. Due to our choice of the shift operator, the state of zero distance (and thus of $K_{\alpha\beta}^{(l)} = k^{(l)}$) is always a fixed point. Assuming that $C_{\alpha\beta}^{(l)} = c^{(l)}$, we obtain

$$K_{\alpha\beta}^{(l+1)} = \sigma_b^2 + \sigma_w^2 \underbrace{\sum_\gamma A_{\alpha\gamma}}_{=1} \underbrace{\sum_\delta A_{\beta\delta}}_{=1} c^{(l)} = k^{(l+1)}\,. \tag{19}$$

In an overmoothing GCN, this fixed point is also attractive, meaning that also pairs of feature inputs $\boldsymbol{x}_\alpha^{(0)}, \boldsymbol{x}_\beta^{(0)}$ which initially have non-zero distance $d(\boldsymbol{x}_\alpha^{(0)}, \boldsymbol{x}_\beta^{(0)}) \neq 0$ (and thus $\mu(\boldsymbol{X}^{(0)}) \neq 0$) eventually converge to the point of vanishing distance. The chaotic, non-oversmoothing phase of a GCN is determined by the condition that this point of constant covariance $K_{\alpha\beta}^{(l)} = k^{(l)}$ becomes unstable. More formally, this can be written in terms of eigenvalues of the linearized dynamics as

$$\max\{|\lambda_i^{\text{p}}|\} \overset{?}{>} 1\,. \tag{20}$$

Here and in the following we will use the superscript p to denote that the linearization is done around the state of constant covariance across nodes in both the oversmoothing and non-oversmoothing phase. The propagation depth $\xi_i \coloneqq -\frac{1}{\ln(\lambda_i)}$ diverges at the phase transition where one λ_i approaches 1. Intuitively speaking, Equation (20) asks whether a small perturbation from the zero distance case diminishes ($\max\{|\lambda_i^{\text{p}}|\} < 1$), in which case the network dynamics is regular, or grows ($\max\{|\lambda_i^{\text{p}}|\} > 1$), in which case the network is chaotic and thus does not oversmooth. The value of $\max\{|\lambda_i^{\text{p}}|\}$ depends on the choices of A, σ_w^2 and σ_b^2 (by the dependence of K^{eq} on σ_b^2). In the following we will concentrate on tuning σ_w^2 to reach the non-oversmoothing phase.

4.2.1 Complete graph

To illustrate the implications of the analysis described above, we first consider a particularly simple GCN on a complete graph; this allows us to calculate the condition for the transition to chaos analytically, and gain some insight into the interesting parameter regimes. Moreover, we use this pedagogical example to show that although the GP equivalence is only true in the limit of infinite hidden feature dimensions, $d_l \to \infty$, our results still describe finite-size GCNs well.

For a complete graph with adjacency matrix $\mathcal{A}_{\alpha\beta} = 1 - \delta_{\alpha\beta}$, our choice of shift operator A in (6) has entries $A_{\alpha\beta} = \frac{g}{N-1} + \delta_{\alpha\beta}(1 - \frac{Ng}{N-1})$. This model is a worst-case scenario for oversmoothing, since the adjacency matrix leads to inputs that are shared across all nodes of the network. We make the ansatz that the equilibrium covariance is of the form $K_{\alpha\beta}^{\text{eq}} = K_c^{\text{eq}} + \delta_{\alpha\beta}(K_a^{\text{eq}} - K_c^{\text{eq}})$ due to symmetry which reduces the problem to only two variables. In this formulation we can use similar methods as in the DNN case [37] to determine the non-oversmoothing condition on the l.h.s in (20) (Details are given in Appendix C).

Figure 1 shows how a GCN on a complete graph can be engineered to be non-oversmoothing by simple tuning of the weight variance σ_w^2. Panel a) shows that increasing the weight variance σ_w^2 or decreasing

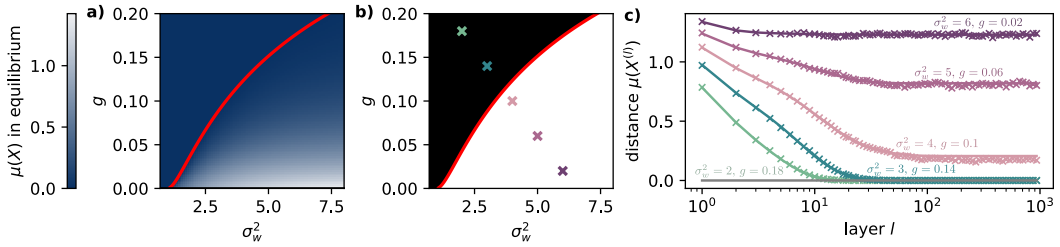

Figure 1: Simulations and GP prior of a GCN on a complete graph with $N = 5$ nodes, shift operator $A_{\alpha\beta} = \frac{g}{N-1} + \delta_{\alpha\beta}(1 - \frac{Ng}{N-1})$, vanishing bias $\sigma_b^2 = 0$ and $\phi(x) = \mathrm{erf}(\frac{\sqrt{\pi}}{2}x)$. **a)** The phase diagram dependent on σ_w^2 and g. The equilibrium feature distance $\mu(\boldsymbol{X})$ obtained from computing the GCN GP prior for $L = 4,000$ layers is shown as a heatmap, the red line is the theoretical prediction for the transition to the non-oversmoothing phase. **b)** Same as in a) but color coding shows whether $\mu(\boldsymbol{X})$ is close to zero (black) or not (white) with precision 10^{-5}. The red line again shows the theoretically predicted phase transition. **c)** Feature distance $\mu(\boldsymbol{X}^{(l)})$ for a random input $X_{\alpha i}^{(0)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$ as a function of layer l. Parameters are written in the panel in matching colors and marked with color coded crosses in the phase diagram in panel b). Feature dimension of the hidden layers is $d_l = 200$, crosses show the mean of 50 network realizations, solid curves the theoretical predictions.

the size of the off-diagonal elements g both shift the network towards the non-oversmoothing phase. Both parameters also increase the equilibrium feature distance beyond the transition. The theoretical prediction for the transition is calculated in Appendix C and shown as the red line. Panel b) confirms the accuracy of this calculation. Larger values of g increase smoothing, and thus larger values of σ_w^2 are needed to compensate. Moreover, our formalism allows us to predict the evolution of feature distances over layers correctly, as can be confirmed in panel c). We find again that GCNs with parameters past the transition do not oversmooth.

4.2.2 General graphs

For general graphs, the transition to the non-oversmoothing phase given by Equation (20) can be determined numerically. As a proof of concept, we demonstrate this approach for the Contextual Stochastic Block Model (CSBM) [8], a common synthetic model which allows generating a graph with two communities and community-wise correlated features on the nodes. Pairs of nodes within the same community have higher probability of being connected and have feature vectors which are more strongly correlated, compared with pairs of nodes from different communities.

Given the underlying graph structure, we can construct the linear map H from Equation (16) and the analytical solution for $C_{\theta\phi}$ in Appendix A. Finding the set of eigenvalues is then a standard task. We show the applicability of our formalism in Figure 2 by showing that GCNs degenerate to a zero distance state state exactly when $\sigma_w^2 < \sigma_{w,\text{crit}}^2$. Panel a) shows how this procedure correctly predicts the transition in the given CSBM instance: The maximum feature distance between any pair of nodes increases from zero at the point where the state $K_{\alpha\beta}^{(l)} = k^{(l)}$ becomes unstable. This means that beyond this point, the GCN has feature vectors that differ across nodes and therefore does not oversmooth. This is more explicitly shown in panels b) and c), where the equilibrium feature distance is plotted as a heatmap. At point A (panel b)), within the oversmoothing phase, all equilibrium feature distances are indeed zero, the network therefore converges to a state in which all features are the same. At point B (panel c)) on the other hand, pairs of nodes exist that have finite distance. In the latter case, one can recognize the community structure of the CSBM: the lower left and upper right quadrants are lighter than the diagonal ones, indicating larger feature distances across communities than within. The equilibrium state thus contains information about the graph topology. This phenomenon is also observed in panel d), where we show the predicted feature distance averaged for nodes within or between classes as a function of layers compared to finite-size simulations. Again, theoretical predictions match with simulations. Thus also on more general graphs the presented formalism predicts the transition point between the oversmoothing and the non-oversmoothing phase, corresponding to a transition between regular and chaotic behavior.

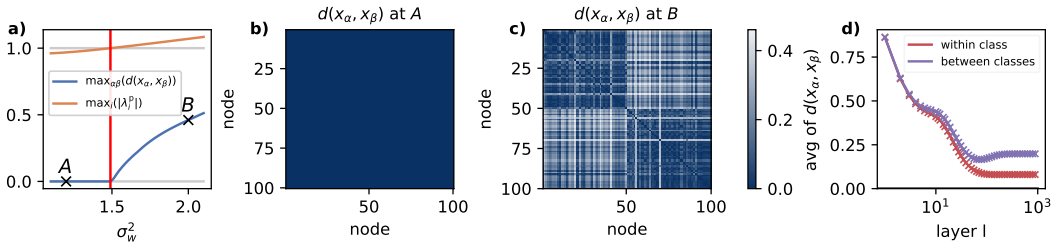

Figure 2: The non-oversmoothing phase in a contextual stochastic block model instance with parameters $N = 100$, $d = 5$, $\lambda = 1$. The shift operator is chosen according to (6) with $g = 0.3$, and $\sigma_b^2 = 0$ and $\phi(x) = \mathrm{erf}(\frac{\sqrt{\pi}}{2}x)$. **a)** The maximum feature distance between any pair of nodes in equilibrium obtained from computing the GCN GP prior for $L = 4,000$ layers (blue) and the largest eigenvalue of the linearized GCN GP dynamics at the zero distance state as a function of weight variance σ_w^2. The red line marks the point where $\max_i\{|\lambda_i^{\mathrm{P}}|\} = 1$. **b)** Heatmap of the equilibrium distance matrix with entries $d_{\alpha\beta} = d(\boldsymbol{x}_\alpha, \boldsymbol{x}_\beta)$ (Equation (13)) at $\sigma_w^2 = 1.3$, marked as point A in panel a). Colorbar shared with the plot in c). **c)** Same as b) but at point B with $\sigma_w^2 = 2$. **d)** Features distances $d_{\alpha\beta}^{(l)} = d(\boldsymbol{x}_\alpha^{(l)}, \boldsymbol{x}_\beta^{(l)})$ as a function of layers for random inputs $X_{\alpha i}^{(0)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,1)$ and a finite-size GCN with $d_l = 200$, averaged for distances for pairs of nodes within the same community (red) and across communities (purple).

We discuss how the assumptions on weight matrices in related theoretical work [2, 45] exclude networks in the chaotic phase in Appendix D, explaining why the non-oversmoothing phase has not been reported before. In Appendix E we observe how increasing the weight variance increases the oversmoothing measure $\mu(X)$ in equilibrium also in the case of the more common shift operator proposed in the original work [19], despite the fact that this shift operator does not have the oversmoothed fixed point in the sense of Equation (19).

4.3 Implications for performance

Lastly we want to investigate the implications of the non-oversmoothing phase on performance. We do this by applying the GCN GP as well as a finite-size GCN to the task of node classification in the CSBM model and measure their performance, shown in Figure 3. Panel a) shows how the generalization error of the GCN GP changes depending on the weight variance σ_w^2 and the number of layers L. In the oversmoothing phase where most GCNs in the literature are initialized (see Appendix D), the generalization error increases significantly already after only a couple of layers. We observe the best performance near the transition to chaos where the GCN GP stays informative up to 100 layers. In panel b) we test the generalization error for even deeper networks. While the generalization error increases to one (being random chance) in the oversmoothing phase, GCN GPs in the chaotic phase stay informative even at more than a thousand layers. This can be explained by Figure 2: For such deep networks, the dynamics are very close to the equilibrium and thus no information of the input features $\boldsymbol{X}^{(0)}$ is transferred to the output. The state, however, still contains information of the network topology from the adjacency matrix, leading to better than random chance performance. In panel c) we explicitly show the layer dependence of the generalization error for the GCN GP at the critical point, in the oversmoothing and in the chaotic phase. Again, we see a fast performance drop for oversmoothing networks, while in the chaotic phase and at the critical point the GCN GP obtains good performance also at large depths, with performance peaking at $L \approx 15$ layers. Tuning the weight variance thus not only prevents oversmoothing, but may also allow the construction of GCNs with more layers and possibly better generalization performance.

In the study of deep networks, results obtained in the limit of infinite feature dimensions $d_l \to \infty$ often are also applicable for finite-size networks [32, 37]. In panel d) we conduct a preliminary analysis for finite-size GCNs by measuring the performance of randomly initialized GCNs for which we only train the readout layer via gradient descent. Indeed, we observe similar behavior as for the GCN GP: Performance drops rapidly over layers in the oversmoothing phase, while performance stays high over many layers at the critical point and in the chaotic phase and peaks at $L \approx 15$ layers.

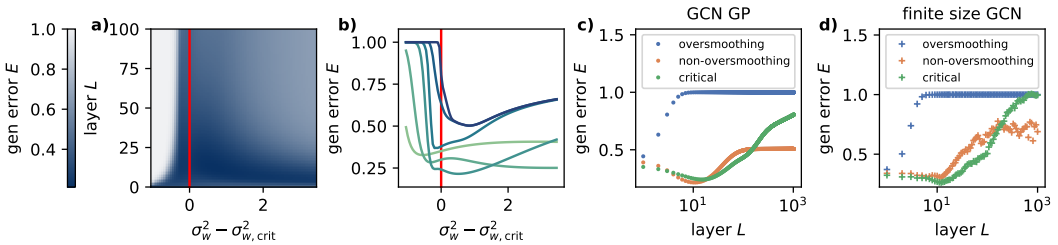

Figure 3: Generalization error (mean squared error) of the Gaussian process for a CSBM with parameters $N = 20$, $d = 5$, $\lambda = 1$, $\gamma = 1$ and $\mu = 4$. The shift operator is defined in (6) with $g = 0.1$, other parameters are $\sigma_b^2 = 0$, $\phi(x) = \text{erf}(\frac{\sqrt{\pi}}{2}x)$ and $\sigma_{ro} = 0.01$. In all panels we use $N^{\text{train}} = 10$ training nodes and $N^{\text{test}} = 10$ test nodes, five training nodes from each of the two communities. Labels are ± 1 for the two communities, respectively. For all panels, we show averages over 50 CSBM instances. **a)** Heatmap of the generalization error of the GCN GP dependent on number of layers L and weight variance σ_w^2. The red line shows the transition to the non-oversmoothing phase. **b)** Generalization error dependent on weight variance σ_w^2 and depths $L = 1, 4, 16, 64, 256, 1024$ from turquoise to dark blue. **c)** Generalization error dependent on the layer for the GCN GP at the critical line $\sigma_w^2 = \sigma_{w,\text{crit}}^2$, in the oversmoothing phase $\sigma_w^2 = \sigma_{w,\text{crit}}^2 - 1$ and the non-oversmoothing phase $\sigma_w^2 = \sigma_{w,\text{crit}}^2 + 1$. **d)** Performance of randomly initialized finite-size GCNs with $d_l = 200$ for $l = 1, \ldots, L$ where only the linear readout layer is trained with gradient descent (details in Appendix F) at the critical line $\sigma_w^2 = \sigma_{w,\text{crit}}^2$, in the oversmoothing phase $\sigma_w^2 = \sigma_{w,\text{crit}}^2 - 1$ and the non-oversmoothing phase $\sigma_w^2 = \sigma_{w,\text{crit}}^2 + 1$.

Additionally, we test the performance of the GCN GP on the real world citation network Cora [38]. Evaluating the eigenvalue condition (20) would be computationally expensive for such a large dataset, therefore we find the transition by numerically evaluating the feature distance $\mu(X)$ in equilibrium and search for the σ_w^2 at which this distance becomes non-zero. This procedure results in $\sigma_{w,\text{crit}}^2 \approx 1$ and is presented in more detail in Appendix G. Figure 4 panel a) shows the performance of the GCN GP dependent on the number of layers L and weight variance σ_w^2: as for the CSBM in Figure 3 we observe that the performance for deep GCN GPs is best near the transition in the non-oversmoothing phase. Furthermore, GCN GPs with more layers achieve lower generalization error. This is shown more directly in panel b). There we observe the layer dependence for GCN GPs near the transition in the non-oversmoothing regime. Indeed, the accuracy increases up to a hundred layers, reaching the accuracy of finite-size GCNs stated in [19].

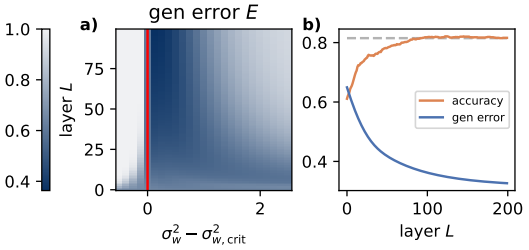

Figure 4: GCN GP performance on the Cora datset [38]. **a)** Generalization error (mean squared error) as a function of layers L and weight variance $\sigma_w^2 - \sigma_{w,\text{crit.}}^2$ for our stochastic shift operator (6) with $g = 0.9$. The value of $\sigma_{w,\text{crit.}}^2 \approx 1$ is determined numerically in Appendix G. **b)** Layer dependent generalization error and accuracy for GCNs near the transition $\sigma_w^2 = \sigma_{w,\text{crit.}}^2 + 0.1$. Grey dashed line shows accuracy obtained for GCNs in the original work [19]. Numerical details in Appendix F.

Near the transition, accuracy increases for up to $L = 100$, and the generalization error improves even beyond this. We hypothesize that this many layers are required for high performance partly due to our choice of the shift operator. The Cora dataset has a maximum degree $d_{\text{max}} = 168$ leading to small off-diagonal elements for the choice of our shift operator: Recall that in Equation (6), the parameter g is constrained to be $g \in (0, 1)$. As a consequence, the off-diagonal elements of the shift operator are $A_{ij} < \frac{1}{d_{\text{max}}} = \frac{1}{168}$. Many convolutional layers are then needed to incorporate information from a node's neighbors.

One might wonder whether it is possible to initialize weights in say the oversmoothing regime, and transition to the non-oversmoothing regime during training. We argue that this is possible in the case of Langevin training (Appendix H).

5 Discussion

In this study we used the equivalence of GCNs and GPs to investigate oversmoothing, the property that features at different nodes converge over layers to the same feature vector in an exponential manner. By extending concepts such as the propagation depth and chaos from the study of conventional deep feedforward neural networks [37], we are able to derive a condition to avoid oversmoothing. This condition is phrased in terms of an eigenvalue problem of the linearized GCN GP dynamics around the state where all features are the same: This state is stable if all eigenvalues are smaller than one, thus the networks do oversmooth. If one eigenvalue, however, is larger than one, the state where the features are the same on all nodes becomes unstable. While most GCNs studied in the literature are in the oversmoothing phase [2, 45], the non-oversmoothing phase can be reached by a simple tuning of the weight variance at initialization. An analogy can be drawn between the chaotic phase of DNNs and the non-oversmoothing phase of GCNs. Previous theoretical works have proven that oversmoothing is inevitable in some GNN architectures, among them GCNs; these works, however, make crucial assumptions on the weight matrices, constraining their variances to be in what we identify as the oversmoothing phase. Near the transition, we find GCNs which are both deep and expressive, matching the originally reported GCN performance [19] on the Cora dataset with GCN GPs beyond 100 layers.

Limitations. The current analysis is based on the equivalence of GCNs and GPs which strictly holds only in the limit of infinite feature dimension. GCNs with large feature vectors ($d_l = 200$) are well described by the theory, as shown in Section 4. For a small number of feature dimensions, however, we expect deviations from the asymptotic results. Throughout the main part of this work, we assumed a row-stochastic shift operator which made the equilibrium K^{eq} in the oversmoothing phase particularly simple. For other shift operators, we expect qualitatively similar results while the equilibrium K^{eq} may look different in detail. In our preliminary experiments on the common shift operator from [19] (Appendix E), we indeed find that increasing the weight variance increases the distances between features also in this case. We hypothesize that this effect makes the equilibrium more informative of the graph topology, as in the stochastic shift operator case. The choice of non-linearity is unrestricted, but in the general case numerical integration of (8) is needed.

To determine whether a given weight variance is in the non-oversmoothing phase, one calculates the eigenvalues of the linearized GCN GP dynamics which take the form of an $N^2 \times N^2$ matrix (see Equation (16)), this has a run time of $\mathcal{O}(N^6)$. While this becomes computationally expensive for large graphs, the conceptual insights of the presented analysis remain. In practical applications with large graphs one may reduce the computational load by determining the transition point via computation of the GCN GP prior until it is close to equilibrium. This procedure has a runtime of $\mathcal{O}(N^3 L^{\mathrm{eq}})$ where L^{eq} is the number of layers after which the process is sufficiently close to equilibrium. One might then do an interval search on the weight variance until the transition point is determined with sufficient accuracy.

Oulook. Formulating GCNs with the help of GPs can be considered the leading order in the number of feature space dimension d_l when approximating finite-size GCNs. Computing corrections for finite numbers of hidden feature dimensions would allow the characterization of feature learning in such networks, similar as in standard deep networks [26, 48, 39]. Moreover, the generalization of this formalism to more general shift operators and other GNN architectures [49, 4] like GATs [41] are possible directions of future research. In the special case of GATs we expect similar results to the GCN analyzed here, since the shift operator is constructed using a softmax and therefore also is row-stochastic.

Acknowledgements

Funded by the European Union (ERC, HIGH-HOPeS, 101039827). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them. We also acknowledge funding by the German Research Council (DFG) within the Collaborative Research Center "Sparsity and Singular Structures" (SfB 1481; Project A07).

References

[1] Uri Alon and Eran Yahav. On the Bottleneck of Graph Neural Networks and its Practical Implications, March 2021. arXiv:2006.05205 [cs, stat].

[2] Chen Cai and Yusu Wang. A Note on Over-Smoothing for Graph Neural Networks, June 2020. arXiv:2006.13318 [cs, stat].

[3] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and Relieving the Over-Smoothing Problem for Graph Neural Networks from the Topological View. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3438–3445, April 2020. Number: 04.

[4] Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and Deep Graph Convolutional Networks. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1725–1735. PMLR, November 2020. ISSN: 2640-3498.

[5] Aditya Cowsik, Tamra Nebabu, Xiao-Liang Qi, and Surya Ganguli. Geometric Dynamics of Signal Propagation Predict Trainability of Transformers, March 2024. arXiv:2403.02579 [cond-mat].

[6] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, December 1989.

[7] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[8] Yash Deshpande, Subhabrata Sen, Andrea Montanari, and Elchanan Mossel. Contextual Stochastic Block Models. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[9] Vijay Prakash Dwivedi, Ladislav Rampášek, Michael Galkin, Ali Parviz, Guy Wolf, Anh Tuan Luu, and Dominique Beaini. Long Range Graph Benchmark. *Advances in Neural Information Processing Systems*, 35:22326–22340, December 2022.

[10] Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(11):113301, November 2020. Publisher: IOP Publishing and SISSA.

[11] Jean Ginibre. Statistical Ensembles of Complex, Quaternion, and Real Matrices. *Journal of Mathematical Physics*, 6(3):440–449, March 1965.

[12] William L. Hamilton. *Graph Representation Learning*. Morgan & Claypool Publishers, September 2020.

[13] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. Publisher: Nature Publishing Group.

[14] Jilin Hu, Jianbing Shen, Bin Yang, and Ling Shao. Infinitely Wide Graph Convolutional Networks: Semi-supervised Learning via Gaussian Processes, February 2020. arXiv:2002.12168 [cs, stat].

[15] Tianjin Huang, Tianlong Chen, Meng Fang, Vlado Menkovski, Jiaxu Zhao, Lu Yin, Yulong Pei, Decebal Constantin Mocanu, Zhangyang Wang, Mykola Pechenizkiy, and Shiwei Liu. You Can Have Better Graph Neural Networks by Not Training Weights at All: Finding Untrained GNNs Tickets, February 2024. arXiv:2211.15335 [cs].

[16] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[17] Yufei Jin and Xingquan Zhu. ATNPA: A Unified View of Oversmoothing Alleviation in Graph Neural Networks, May 2024. arXiv:2405.01663 [cs].

[18] Nicolas Keriven. Not too little, not too much: a theoretical analysis of graph (over)smoothing. *Advances in Neural Information Processing Systems*, 35:2268–2281, December 2022.

[19] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks, February 2017. arXiv:1609.02907 [cs, stat].

[20] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep Neural Networks as Gaussian Processes, March 2018. arXiv:1711.00165 [cs, stat].

[21] Qimai Li, Zhichao Han, and Xiao-ming Wu. Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), April 2018. Number: 1.

[22] Sitao Luan, Mingde Zhao, Xiao-Wen Chang, and Doina Precup. Training Matters: Unlocking Potentials of Deeper Graph Convolutional Neural Networks, October 2022. arXiv:2008.08838 [cs, stat].

[23] Yao Ma and Jiliang Tang. *Deep Learning on Graphs*. Cambridge University Press, September 2021. Google-Books-ID: _AVDEAAAQBAJ.

[24] L. Molgedey, J. Schuchhardt, and H. G. Schuster. Suppressing chaos in neural networks by noise. *Physical Review Letters*, 69(26):3717–3719, December 1992. Publisher: American Physical Society.

[25] Gadi Naveh, Oded Ben David, Haim Sompolinsky, and Zohar Ringel. Predicting the outputs of finite deep neural networks trained with noisy gradients. *Physical Review E*, 104(6):064301, December 2021. Publisher: American Physical Society.

[26] Gadi Naveh and Zohar Ringel. A self consistent theory of Gaussian Processes captures feature learning effects in finite CNNs. In *Advances in Neural Information Processing Systems*, volume 34, pages 21352–21364. Curran Associates, Inc., 2021.

[27] Radford M Neal. Priors for infinite networks (tech. rep. no. crg-tr-94-1). *University of Toronto*, 415, 1994.

[28] Yin Cheng Ng, Nicolò Colombo, and Ricardo Silva. Bayesian Semi-supervised Learning with Graph Gaussian Processes. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[29] Zehao Niu, Mihai Anitescu, and Jie Chen. Graph Neural Network-Inspired Kernels for Gaussian Processes in Semi-Supervised Learning, February 2023. arXiv:2302.05828 [cs, stat].

[30] Kenta Oono and Taiji Suzuki. Graph Neural Networks Exponentially Lose Expressive Power for Node Classification, January 2021. arXiv:1905.10947 [cs, stat].

[31] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.

[32] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[33] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the Expressive Power of Deep Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2847–2854. PMLR, July 2017. ISSN: 2640-3498.

[34] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2006. OCLC: ocm61285753.

[35] T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A Survey on Oversmoothing in Graph Neural Networks, March 2023. arXiv:2303.10993 [cs].

[36] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks, February 2014. arXiv:1312.6120 [cond-mat, q-bio, stat].

[37] Samuel S. Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep Information Propagation, April 2017. arXiv:1611.01232 [cs, stat].

[38] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective Classification in Network Data. *AI Magazine*, 29(3):93–93, September 2008. Number: 3.

[39] Inbar Seroussi, Gadi Naveh, and Zohar Ringel. Separation of scales and a thermodynamic description of feature learning in some CNNs. *Nature Communications*, 14(1):908, February 2023. Number: 1 Publisher: Nature Publishing Group.

[40] Yunchong Song, Chenghu Zhou, Xinbing Wang, and Zhouhan Lin. Ordered GNN: Ordering Message Passing to Deal with Heterophily and Over-smoothing, February 2023. arXiv:2302.01524 [cs].

[41] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, February 2018. arXiv:1710.10903 [cs, stat].

[42] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, and Paul van Mulbregt. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, March 2020. Publisher: Nature Publishing Group.

[43] Christopher Williams. Computing with Infinite Networks. In *Advances in Neural Information Processing Systems*, volume 9. MIT Press, 1996.

[44] Christopher K. I. Williams. Computation with Infinite Neural Networks. *Neural Computation*, 10(5):1203–1216, July 1998.

[45] Xinyi Wu, Amir Ajorlou, Zihui Wu, and Ali Jadbabaie. Demystifying Oversmoothing in Attention-Based Graph Neural Networks, October 2023. arXiv:2305.16102 [cs, stat].

[46] Xinyi Wu, Zhengdao Chen, William Wang, and Ali Jadbabaie. A Non-Asymptotic Analysis of Oversmoothing in Graph Neural Networks, February 2023. arXiv:2212.10701 [cs, stat].

[47] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, January 2021. Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

[48] Jacob A. Zavatone-Veth, William L. Tong, and Cengiz Pehlevan. Contrasting random and learned features in deep Bayesian linear regression. *Physical Review E*, 105(6):064118, June 2022. Publisher: American Physical Society.

[49] Lingxiao Zhao and Leman Akoglu. PairNorm: Tackling Oversmoothing in GNNs, February 2020. arXiv:1909.12223 [cs, stat].

A Analytical solution for expectation values

To evaluate our theory, we need to compute expectation values given in the form of (8) which we restate here for readability

$$C^{(l)}_{\gamma\delta} = \left\langle \phi\left(h^{(l)}_\gamma\right)\phi\left(h^{(l)}_\delta\right)\right\rangle_{h^{(l)}_\gamma, h^{(l)}_\delta}, \tag{21}$$

where $h^{(l)}_\gamma$ and $h^{(l)}_\delta$ are zero mean random Gaussian variables with $\langle h^{(l)}_\gamma h^{(l)}_\delta\rangle = K^{(l)}_{\gamma\delta}$. This can be evaluated numerically for general non-linearities $\phi(x)$. For simplicity, however, we choose $\phi(x) = \mathrm{erf}(\frac{\sqrt{\pi}}{2}x)$ in our experiments where the scaling factor in the erf is chosen such that $\frac{\partial\phi}{\partial x}(0) = 1$. In this case, the expectation value can be evaluated analytically to be

$$C^{(l)}_{\gamma\delta} = \frac{2}{\pi}\,\arcsin\left(\frac{\frac{\pi}{2}K^{(l)}_{\gamma\delta}}{\sqrt{1+\frac{\pi}{2}K^{(l)}_{\gamma\gamma}}\sqrt{1+\frac{\pi}{2}K^{(l)}_{\delta\delta}}}\right). \tag{22}$$

as shown in [44].

B The linearized GP of GCNs

We start from the full GCN GP iterative map (8,(9)), which we restate here for readability

$$K^{(l+1)}_{\alpha\beta} = T_{\alpha\beta}[K^{(l)}] = \sigma_b^2 + \sigma_w^2 \sum_{\gamma,\delta} A_{\alpha\gamma}A_{\beta\delta}C^{(l)}_{\gamma\delta}[K^{(l)}] \tag{23}$$

where $h^{(l)}_\gamma$ and $h^{(l)}_\delta$ are drawn from a 0-mean Gaussian distribution with covariance $\langle h^{(l)}_\gamma h^{(l)}_\delta\rangle = K^{(l)}_{\gamma\delta}$. The full covariance matrix of layer l is denoted as $K^{(l)}$. Here, we distinguish between the iterative maps $T_{\alpha\beta} : \mathbb{R}^{N\times N} \to \mathbb{R}$ of which there are N^2, one for each pair of nodes α and β, and the entries $K^{(l+1)}_{\alpha\beta}$ of the covariance matrix in the layer $l+1$. Notice that $T_{\alpha\beta} = T_{\beta\alpha}$ due to the symmetry of the covariance matrix. In the maps $T_{\alpha\beta}$, the covariance matrix of the previous layer only shows up in the expectation value $C_{\gamma\delta} = \left\langle \phi\left(h^{(l)}_\gamma\right)\phi\left(h^{(l)}_\delta\right)\right\rangle_{h^{(l)}_\gamma, h^{(l)}_\delta}$ such that the linearized dynamics around a fixed point (being equilibrium $K^{\mathrm{fix}}_{\alpha\beta} = K^{\mathrm{eq}}_{\alpha\beta}$ or zero distance state $K^{\mathrm{fix}}_{\alpha\beta} = k$) with $K^{(l)}_{\alpha\beta} = K^{\mathrm{fix}}_{\alpha\beta} + \Delta^{(l)}_{\alpha\beta}$ read

$$K^{(l+1)}_{\alpha\beta} = K^{\mathrm{fix}}_{\alpha\beta} + \Delta^{(l+1)}_{\alpha\beta} = \underbrace{T_{\alpha\beta}[K^{\mathrm{fix}}]}_{=K^{\mathrm{fix}}_{\alpha\beta}} + \sum_{\gamma<\delta}\frac{\partial T_{\alpha\beta}}{\partial K_{\gamma\delta}}[K^{\mathrm{fix}}]\Delta^{(l)}_{\gamma\delta} + \mathcal{O}((\Delta^{(l)})^2) \tag{24}$$

$$= K^{\mathrm{fix}}_{\alpha\beta} + \sum_{\gamma,\delta}\underbrace{\sigma_w^2\sum_{\theta,\phi}\frac{1}{2}(1+\delta_{\gamma,\delta})A_{\alpha\theta}A_{\beta\phi}\frac{\partial C_{\theta\phi}}{\partial K_{\gamma\delta}}[K^{\mathrm{fix}}]}_{\equiv H_{\alpha\beta,\gamma\delta}}\Delta^{(l)}_{\gamma\delta} + \mathcal{O}((\Delta^{(l)})^2), \tag{25}$$

where we restrict the sum in (24) to $\gamma < \delta$ since $K_{\gamma\delta}$ and $K_{\delta\gamma}$ are the same quantity. From (24) follows

$$\Delta^{(l+1)}_{\alpha\beta} = \sum_{\gamma,\delta} H_{\alpha\beta,\gamma\delta}\Delta^{(l)}_{\gamma\delta} + \mathcal{O}((\Delta^{(l)})^2) \tag{26}$$

which is Equation (16) in the main text. While H is not symmetric in general, $H_{\alpha\beta,\gamma\delta} \neq H_{\gamma\delta,\alpha\beta}$, it is symmetric in the first and second pair of covariance indices, $H_{\alpha\beta,\gamma\delta} = H_{\beta\alpha,\gamma\delta}$ and $H_{\alpha\beta,\gamma\delta} = H_{\alpha\beta,\delta\gamma}$ due to symmetry of the covariance matrices, $K_{\alpha\beta} = K_{\beta\alpha}$.

In the main text, we look for the right eigenvectors of H fulfilling

$$\lambda_i V^{(i)}_{\alpha\beta} = \sum_{\gamma,\delta} H_{\alpha\beta,\gamma\delta}V^{(i)}_{\gamma\delta}. \tag{27}$$

These are for general non-symmetric matrices not orthogonal. In order to decompose $\Delta^{(l)}$ to overlaps with the eigenvectors $V^{(i)}_{\alpha\beta}$ we need to find the dual vectors $U^{(i)}_{\alpha\beta}$ fulfilling

$$\sum_{\alpha,\beta} U^{(i)}_{\alpha\beta} V^{(j)}_{\alpha\beta} = \delta_{ij} \tag{28}$$

from which we can define

$$\Delta^{(l)}_i = \sum_{\alpha,\beta} U^{(i)}_{\alpha\beta} \Delta^{(l)}_{\alpha\beta} \tag{29}$$

such that

$$\Delta^{(l)}_{\alpha\beta} = \sum_i \Delta^{(l)}_i V^{(i)}_{\alpha\beta} \tag{30}$$

as stated in the main text.

C Investigation of the complete graph model

In this section we analytically investigate the complete graph model as defined in the main text. Specifically, we consider networks with the shift operator

$$A_{\alpha\beta} = \frac{g}{N-1} + \delta_{\alpha\beta}(1 - \frac{Ng}{N-1}) \tag{31}$$

and $\phi = \mathrm{erf}(\sqrt{\pi}x/2)$. Due to the symmetry of the system we make the ansatz

$$K^{\mathrm{eq}}_{\alpha\beta} = K^{\mathrm{eq}}_{\mathrm{a}} + \delta_{\alpha\beta}(K^{\mathrm{eq}}_{\mathrm{c}} - K^{\mathrm{eq}}_{\mathrm{a}}) \,, \tag{32}$$

meaning that we assume constant variances $K^{\mathrm{eq}}_{\mathrm{a}}$ across nodes and that all pairs of nodes have the same covariance $K^{\mathrm{eq}}_{\mathrm{c}}$, reducing the system to only two unknown variables. The equilibrium is a fixed point of the iterative map of the GCN GP. With the special choice of shift operator in (31) this becomes

$$K^{(l+1)}_{\mathrm{a}} = \sigma_b^2 + g_{\mathrm{a}} C^{(l)}_{\mathrm{a}} + g_{\mathrm{c}} C^{(l)}_{\mathrm{c}} \tag{33}$$

and

$$K^{(l)}_{\mathrm{c}} = \sigma_b^2 + h_{\mathrm{a}} C^{(l)}_{\mathrm{a}} + h_{\prime\mathrm{c}} C^{(l)}_{\mathrm{c}} \tag{34}$$

with constants

$$g_{\mathrm{a}} = (1 + \frac{g^2}{N-1})\sigma_w^2 \tag{35}$$

$$g_{\mathrm{c}} = 2(g + \frac{g^2(N-2)}{N-1})\sigma_w^2 \tag{36}$$

$$h_{\mathrm{a}} = 2(\frac{g}{N-1} + \frac{g^2(N-2)}{(N-1)^2})\sigma_w^2 \tag{37}$$

$$h_{\mathrm{c}} = (1 + \frac{g^2}{(N-1)^2} + 2\frac{g^2(N-2)(N-3)}{(N-1)^2} + 4\frac{g(N-2)}{(N-1)})\sigma_w^2 \tag{38}$$

and

$$C^{(l)}_{\mathrm{a}} = \langle \phi(h^{(l)}_\alpha)\phi(h^{(l)}_\alpha)\rangle \tag{39}$$

$$C^{(l)}_{\mathrm{c}} = \langle \phi(h^{(l)}_\alpha)\phi(h^{(l)}_\beta)\rangle \qquad \text{for } \alpha \neq \beta \,. \tag{40}$$

The preactivations are Gaussian distributed with zero mean and covariance $\langle h^{(l)}_\alpha h^{(l)}_\beta \rangle = K^{(l)}_{\alpha\beta}$. In the oversmoothing phase, we know that $\mu(\boldsymbol{X}) = 0$ in equilibrium. We have seen in Section 4.2 that this corresponds to $K^{\mathrm{eq}}_{\alpha\beta} = k^{\mathrm{eq}}$, implying for our ansatz that $K^{\mathrm{eq}}_{\mathrm{c}} = K^{\mathrm{eq}}_{\mathrm{a}}$. We will find the transition to chaos by calculating where this state becomes unstable with regard to small perturbations. Specifically, we define $c^{(l)} = \frac{K^{(l)}_{\mathrm{c}}}{K^{(l)}_{\mathrm{a}}}$ and look for the parameter point where

$$1 \overset{?}{>} \frac{\partial c^{(l+1)}}{\partial c^{(l)}}\Big|_{c^{(l)}=1} \,. \tag{41}$$

The authors in [37] used this approach to find the transition to chaos for DNNs. The correlation coefficient is

$$c^{(l+1)} = \frac{K_{\mathrm{c}}^{(l+1)}}{K_{\mathrm{a}}^{(l+1)}} = \frac{\sigma_b^2 + h_{\mathrm{a}} C_{\mathrm{a}}^{(l)} + h_{\mathrm{c}} C_{\mathrm{c}}^{(l)}}{\sigma_b^2 + g_{\mathrm{a}} C_{\mathrm{a}}^{(l)} + g_{\mathrm{c}} C_{\mathrm{c}}^{(l)}} \tag{42}$$

and Equation (41) thus becomes

$$\frac{\partial c^{(l+1)}}{\partial c^{(l)}}\Big|_{c^{(l)}=1} = \frac{h_{\mathrm{c}} \frac{\partial C_{\mathrm{c}}^{(l+1)}}{\partial c^{(l)}} K_{\mathrm{a}}^{(l+1)} - g_{\mathrm{c}} \frac{\partial C_{\mathrm{c}}^{(l+1)}}{\partial c^{(l)}} K_{\mathrm{c}}^{(l+1)}}{(K_{\mathrm{a}}^{(l+1)})^2} \,. \tag{43}$$

Since we look at this equation at the perfectly correlated state $c^{(l)} = 1$, we know that $K_{\mathrm{a}}^{(l)} = K_{\mathrm{c}}^{(l)}$ (implying that $C_{\mathrm{a}}^{(l)} = K_{\mathrm{c}}^{(l)}$) and can determine $K_{\mathrm{a}}^{(l)}$ as the solution of the fixed point equation

$$K_{\mathrm{a}}^{(l+1)} = \sigma_b^2 + (g_{\mathrm{a}} + g_{\mathrm{c}}) C_{\mathrm{a}}^{(l)} \tag{44}$$

to which the GCN GP dynamics reduce in the zero distance state (by using the fact that $\sum_\beta A_{\alpha\beta} = 1$ an $C_{\alpha\beta}^{(l)} = c^{(l)}$). Lastly, we can calculate

$$\frac{\partial C_{\mathrm{c}}^{(l+1)}}{\partial c^{(l)}}\Big|_{c=1} = \frac{\partial}{\partial c^{(l)}} \left(\frac{2}{\pi} \arcsin \left(\frac{\frac{\pi}{2} K_{\mathrm{a}}^{(l)} c^{(l)}}{1 + \frac{\pi}{2} K_{\mathrm{a}}^{(l)}} \right) \right) \Bigg|_{c^{(l)}=1} \tag{45}$$

$$= \frac{2}{\pi} \frac{1}{\sqrt{1 - \left(\frac{\frac{\pi}{2} K_{\mathrm{a}}^{(l)} c^{(l)}}{1 + \frac{\pi}{2} K_{\mathrm{a}}^{(l)}} \right)^2}} \frac{K_{\mathrm{a}}^{(l)}}{\frac{2}{\pi} + K_{\mathrm{a}}^{(l)}} \Bigg|_{c^{(l)}=1} \tag{46}$$

$$= \frac{2}{\pi} \frac{1}{\sqrt{1 - \left(\frac{K_{\mathrm{a}}^{(l)}}{\frac{2}{\pi} + K_{\mathrm{a}}^{(l)}} \right)^2}} \frac{K_{\mathrm{a}}^{(l)}}{\frac{2}{\pi} + K_{\mathrm{a}}^{(l)}} \,, \tag{47}$$

where we used the known solution for the expectation value C_{c} for $\phi(x) = \mathrm{erf}(\sqrt{\pi} x/2)$ from Appendix A. Plugging this into Equation (43) lets us calculate $\frac{\partial c^{(l+1)}}{\partial c^{(l)}}\Big|_{c^{(l)}=1}$ and thus determine the transition to the non-oversmoothing phase. This is plotted as a red line in Figure 1 panel a) and b).

D Restriction of weight matrices in related work

In this section we will show histograms of critical weight variances $\sigma_{w,\mathrm{crit}}^2$ and discuss how the assumptions in [2] and [45] exclude networks in the non-oversmoothing phase. Our results thus stand in no conflict with the results of these works.

Here we want to argue that the assumptions on the weight matrices in related work constrains their architectures to the oversmoothing phase. We start with [45] in which the authors study graph attention networks (GATs). Although we study a standard GCN here, we hypothesize that increasing the weight variance at initialization likewise prevents oversmoothing in other architecture, such as the GAT in [45]. The critical assumption constraining them to the oversmoothing phase is their assumption A3, stating that $\{|| \prod_{l=0}^{k} |W^{(l)}| ||_{\max}\}_{k=0}^{\infty}$ is bounded where $||M||_{\max} = \max_{i,j} |M_{i,j}|$. For our setting of randomly drawn weight matrices $W_{ij}^{(l)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \frac{\sigma_w^2}{d_l})$ with $W_{ij}^{(l)} \in \mathbb{R}^{N \times N}$, this restricts us to values $\sigma_w^2 \leq 1$. This can be seen by using the circular law from random matrix theory [11]: It is known that the eigenvalues of a matrix with i.i.d. random Gaussian entries of the form above have eigenvalues uniformly distributed in a circle around 0 in the complex plane with radius $\sqrt{\sigma_w^2}$ in the limit $d_l \to \infty$. Thus, the maximal real part of any eigenvalue of this matrix is $\sqrt{\sigma_w^2}$. Thus we can estimate $|| \prod_{l=0}^{k} |W^{(l)}| ||_{\max} \leq c(\sqrt{\sigma_w^2})^k$ with a constant c. To enter the non-oversmoothing phase, we need $\sigma_w^2 > 1$. In this case, the latter expression diverges for $k \to \infty$, thus being excluded by the proof in [45]. Indeed, for the CSBMs we investigated in this work, all critical weight variances $\sigma_{w,\mathrm{crit}}^2$ are larger than 1 as shown in Figure 5. Also in our model of the complete graph and the CSBM investigated in Section 4.2.2 all $\sigma_{w,\mathrm{crit}}^2$ are larger than 1, compare Figure 1 and Figure 2.

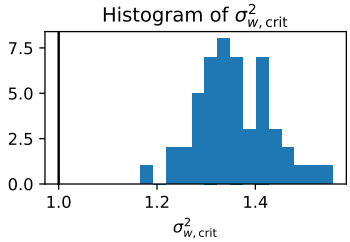

Figure 5: Histogram of $\sigma^2_{w,\text{crit}}$ for the 50 CSBM instances used in the experiment of Figure 3. The point 1 is marked for comparison to related work.

The authors of [2] and [30] also study GCNs,; however they consider a different shift operator than in this work. In both of [2] and [30] the authors find that their GCN models exponentially loose expressive power if $s\overline{\lambda} < 1$, with $\overline{\lambda}$ being the maximal singular value/eigenvalue of the shift operator and s being the maximal singular value of all weight matrices. Again, the maximal singular value is limited (dependent on $\overline{\lambda}$), which in our approach translates to a limit on σ_w^2. While the authors notice that their bounds do not hold for large singular values s, they do not observe a non-oversmoothing phase in their models.

E Non-oversmoothing GCNs with non-stochastic shift operators

Here we investigate how increasing the weight variance can mitigate oversmoothing also in the case of the original shift operator proposed by Kipf and Welling in [19] being

$$A_{\text{KW}} = (D')^{-1/2}\mathcal{A}'(D')^{-1/2} \tag{48}$$

with $\mathcal{A}' = \mathbb{I} + \mathcal{A}$ and $D'_{ij} = \delta_{ij}\sum_k \mathcal{A}'_{ik}$. This shift operator A_{KW} is not row-stochastic, therefore the state $K_{\alpha\beta} = k$ is not a fixed point for $k \neq 0$, as can be seen from Equation (19). Thus we need to differentiate between two kind of fixed points: Either, all features are zero, for which $K_{\alpha\beta} = 0$ and $\mu(X) = 0$, or some $K_{\alpha\beta} \neq 0$ in which case we have $\mu(X) > 0$. Importantly, for this shift operator, there is no intermediate case for which $K_{\alpha\beta} = k$ with $k \neq 0$. Consequently there is no oversmoothing regime with respect to the measure $\mu(X)$ where node features are non-zero. For an in depth study of this case (48), the definition of another oversmoothing measure μ' incorporating the different values of $\sum_\beta (A_{\text{KW}})_{\alpha\beta}$ for different α may be more appropriate. For our purposes, it will suffice to analyze the shift operator (48) with the measure $\mu(X)$ from Equation (14).

Figure 6 panel a) shows how $\mu(X)$ in equilibrium increases for larger weight variances σ_w^2 for the shift operator (48). For comparison, we also show the results obtained with our row-stochastic shift operator (6). Thus we find that also for the shift operator A_{KW} the pairwise distance between features can be increased by increasing the weight variance σ_w^2. The non-existence of an oversmoothed fixed point except for the special case $K_{\alpha\beta} = 0$ which we argued for above is observed in panel b) and c): The oversmoothing measure $\mu(X)$ is zero for A_{KW} if and only if all entries of the covariance matrix are zero, implying that all preactivations and thus also all features are zero. This is a qualitative difference to our shift operator A for which we find equilibrium states with non-zero $\max_\alpha K^{\text{eq}}_{\alpha\alpha}$ but still $\mu(X) = 0$.

F Details of numerical experiments

To conduct our experiments we use NumPy [13], SciPy [42] (both available under a BSD-3-Clause License) and Scikit-learn (sklearn) [31] (available under a New BSD License). The code is publicly available under `https://github.com/bepping/non-oversmoothing-gcns`. For our experiments with the Cora dataset, we use the readin methods from [19] which are available under a MIT license (Copyright (c) 2016 Thomas Kipf). Computations were performed on CPUs.Requirements for the experiments with synthetic data are (approximately):

- Figure 1: 10mins on a single core laptop.

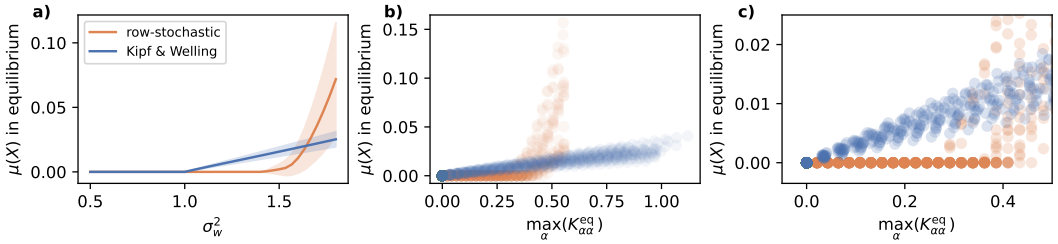

Figure 6: Oversmoothing in GCN GPs with the commonly used shift operator (48) (called Kipf & Welling in the label) and our row-stochastic shift operator (6). **a)** Feature distance $\mu(X)$ in equilibrium dependent on weight variance σ_w^2 obtained from simulating the GCN GP priors for $L^{\text{eq}} = 4,000$ layers. Shown are the averages (solid lines) and standard deviations (shaded areas) over 20 CSBM initializations with $\lambda = 1$, $d = 5$ and $N = 30$ nodes.**b)** Scatter plot of maximal variance of all nodes $\max_\alpha\{K^{\text{eq}}_{\alpha\alpha}\}$ and feature distance $\mu(X)$ in equilibrium for the same data as in plot a). **c)** Same as b) but zoomed into lower left corner.

- Figure 2: 10h on a single node on an internal CPU cluster. Most of the computation time is needed for evaluating $\max(\lambda_i^{\text{p}})$ in panel a).
- Figure 3: 2h on a single core laptop. In panel d), the last layer of finite-size GCNs is trained with the standard settings from sklearn.linear_model.SGDRegressor().
- Figure 5: Byproduct of Figure 2.
- Figure 6: 10min on a single core laptop.

We also experimented on the real world benchmark dataset Cora [38]. This is a citation network with 2708 nodes, representing publications, and 5429 edges, representing the citations between them (we use undirected edges). The publications are divided into seven classes, and the task is to predict these classes for the unlabeled nodes. Node features are of 0/1 valued vectors indicating the absence/presence of words from a dictionary in the titles of the respective publication. The dictionary consists of 1433 unique words. Requirements for the experiments with the Cora dataset are:

- Figure 4: 1h on a single core laptop.
- Figure 7: 15h on a single core laptop.

G Non-oversmoothing transition in the Cora dataset

For Figure 4 we numerically determined the transition to the non-oversmoothing regime for the Cora dataset. The transition was estimated by measuring the distance $\mu(\boldsymbol{X})$ at equilibrium (i.e., after many layers), and determining at which value of σ_w^2 it becomes larger than a small distance $\epsilon = 10^{-5}$. The results of this experiment are shown in Figure 7. We find the critical point to be $\sigma^2_{w,\text{crit}} = 1$ while using a step size of $\delta\sigma_w^2 = 0.01$.

H Transitioning between regimes during training

In this section argue why we think it is possible to transition from the oversmoothing to the non-oversmoothing regime or vice versa during training.

In the GP limit of infinitely many hidden features with a finite amount of training data, the variance of weights of a neural network is the same before and after training. This is because weights only change marginally in this limit, also known as the lazy training regime [16, 10]. We will use this fact to argue that Langevin training is capable of transitioning from one regime to the other.

Langevin training is a gradient based training scheme with external noise and a decay term. The gradient flow equation is given as

$$\frac{\mathrm{d}W_{ij}}{\mathrm{d}t} = -\gamma W_{ij} - \nabla_{W_{ij}} L + B_{ij}\,,$$

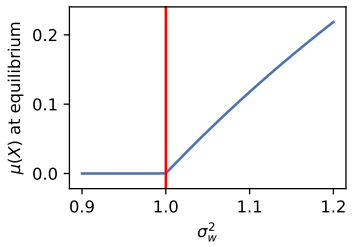

Figure 7: Node distance measure $\mu(\boldsymbol{X})$ at equilibrium obtained from computing the GCN GP prior for $L = 4,000$ layers as a function of σ_w^2. The transition to the non-oversmoothing regime is estimated by checking where the node distance measure is larger than $\epsilon = 10^{-5}$, marked as the red line.

where W_{ij} are the weights to be trained, L is the loss function and B_{ij} denotes external white noise $\langle B_{ij}(t)B_{kl}(s)\rangle = \delta_{i,k}\delta_{j,l}\delta(t-s)D$ with strength D where $\delta_{a,b}$ and $\delta(a-b)$ denote the Kronecker or Dirac delta, respectively. Both γ and D are parameters of this training scheme. It is known that the distribution of weights converges to the posterior weight distribution of a neural network GP with weight variance $\sigma_w^2 = \frac{D}{2\gamma}$ in the infinite feature limit [25] which, as we have noticed above, is the same as the prior distribution.

Since the Langevin distribution converges to the GP weight posterior for any initial distribution, one can imagine an initial distribution with a variance that initializes the network in the oversmoothing regime, while the parameters γ and D are chosen such that $\sigma_w^2 = \frac{D}{2\gamma} > \sigma_{w,\text{crit}}^2$ implying that after training most weight realizations will be in the non-oversmoothing regime. Thus, in this case the GCN would have transitioned from one regime to the other. While this argument is made in the limit of infinitely many hidden features, we think that qualitatively similar results are possible for a large but finite number of hidden feature dimensions. The transition in the reverse direction is possible by the same argument only with the initial and the final variances exchanged.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We describe our approach, the model we study and discuss the contribution of our work.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations of our work are addressed in the discussion.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provide the calculations leading to our results in the main text or appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The procedures for producing the experimental results in this work are described clearly, all necessary parameters are given in the figure captions.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: While the paper provides sufficient instructions to reproduce all experimental results in this paper, we publish our code in a git repository.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All details for the experiments can be found in the respective figure captions.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: In our experiments we compare mean values with theoretical prediction. The numbers of instances used for each experiment are given in the respective figure captions.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The computing resources needed to reproduce our experiments are given in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work presented in this paper conforms with the NeurIPS Code of Ethics in every aspect.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The work presented in this paper is of theoretical nature, we do not foresee any direct societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release any code or data that poses such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We mention the assets we use in the appendix, cite the creators and state the licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.