INTERDISCIPLINARY DOCTORAL SCHOOL

Faculty of Electrical Engineering and Computer Science

Adrian-Victor MOLDOVAN

Automated Data Analysis

SUMMARY

Scientific supervisor

Prof.Univ.Dr.Mat. Răzvan ANDONIE

BRAȘOV, 2024

Contents

CHAPTER 1

Introduction

1.1 Motivation and importance of research

The PhD thesis explores the application of Transfer Entropy (TE) across three domains—neural networks for image classification, convolutional neural network compression under rate distortion theory, and graph convolutional networks—showcasing TE's effectiveness in enhancing, monitoring, and evolving algorithms. The research highlights TE's versatility in identifying asymmetric information dependencies, demonstrating significant benefits and few limitations. TE's integration with complex deep neural architectures facilitates model optimization and self-tuning, although choosing appropriate estimation methods can be challenging. With neural networks growing in complexity and parameter count, TE emerges as a tool to extract performance gains, diagnose architectural issues, and mitigate overfitting, despite difficulties in fine-tuning and computational demands.

1.2 Original Contributions

Our research has demonstrated the potential of incorporating TE feedback into various neural network architectures, from simple feedforward networks [65] to more complex CNNs [66], [67] and GCNs[68]. The *FF+FB* algorithm has shown promising results in terms of training efficiency, stability, and performance. When examined TE as a metric for describing the information planes that show the compression evolution in feedforward networks and CNNs [67], TE has shown all the established properties of the Information Bottleneck method and additional novel properties. In CNNs for image classification tasks, our improvements reduced the number of epochs needed for established accuracy limits. In GCNs, we have successfully used the relational properties of the dataset to improve the validation accuracy. Although there are still open questions and areas for further investigation, our work contributes to ongoing efforts to improve neural network training

and understanding of information flow within these systems.

1.3 Thesis Structure

The thesis "Automated Data Analysis" is structured into two major segments: theoretical foundations covered in Chapters 2 and 3, and published findings detailed in Chapters 4 through 7. Chapter 2 delves into the core concepts of information theory, including entropy, Kullback-Leibler divergence, and Granger causality, elucidating their roles in capturing different facets of information content and flow in complex systems.

Chapter 3 builds upon these fundamentals by introducing Transfer Entropy, Information Bottleneck, and Graph Neural Networks, which are central to the research. It discusses computational challenges in estimating information-theoretic measures, particularly Transfer Entropy, and the impact of discretization on neural network activations. The chapter also examines the Information Bottleneck method within the context of neural networks, exploring its implications for generalization and training dynamics, while acknowledging its limitations. Graph Neural Networks (GNNs) and Graph Convolutional Networks (GCNs) are introduced as powerful tools for handling graph-structured data. The chapter outlines the mathematical underpinnings of GCNs, their architecture, and challenges such as oversmoothing and heterophily, along with mitigation strategies.

The published results section begins with Chapter 4, introducing a novel training algorithm for feedforward neural networks called FF+FB. This algorithm uses TE as a feedback mechanism to enhance learning performance by measuring information transfer between neurons and modulating feedback connections during training. Experiments on standard datasets show improvements in convergence speed and accuracy compared to conventional feedforward networks.

Chapter 5 extends this approach to Convolutional Neural Networks (CNNs), integrating TE feedback connections into the training process. Results on image classification datasets illustrate accelerated convergence and improved accuracy, albeit with added computational overhead. Chapter 6 applies Transfer Entropy within the Information Bottleneck framework to analyze information flow in neural networks, revealing insights into information compression and its correlation with network performance.

Finally, Chapter 7 presents TE-GGCN, a method that integrates a Transfer Entropy control mechanism into the GGCN algorithm to enhance accuracy and tackle oversmoothing and misclassification issues, especially in heterophilic datasets. The experimental results on various datasets confirm improved accuracy, albeit with increased computational demands.

Throughout the thesis, the integration of Transfer Entropy showcases its potential to optimize information flow, accelerate convergence, and improve accuracy, underscoring its utility as a powerful tool in neural network research and development.

CHAPTER 2

Information Theory Concepts

2.1 Entropy

Entropy, in the context of information theory and statistics, is a measure of the uncertainty or unpredictability of a system's state. It quantifies the amount of information needed to describe the state of a system or the expected value of the information in a message [53].

Entropy can be viewed as a statistical measure of variance and chaos in a system. High entropy indicates a high degree of unpredictability or disorder in the system's state, reflecting a more chaotic system. Conversely, low entropy suggests a more ordered or predictable system. In the context of information theory, entropy represents the minimum number of bits required to encode the transmission of states in a message without loss of information.

To briefly summarize the estimation of entropy-related tools, we provide a simple experiment. Visualizing the entropy of the discrete random variable X, which is part of a standard normal distribution, $X \sim \mathcal{N}(\mu, \sigma^2)$, with $\mu = 0$, $\sigma = 1$, we obtain Figure 2.1. In this plot we have discretized the values of X into bins.

Additional estimation techniques will not be detailed in this section, since we will be dealing with similar methods in the TE and Relative Entropy (KL) sections. Also, the above equations were defined since they have a strong connection with TE via the indirect relation with Mutual Information.

2.2 Mutual Information

Using the same notation as in Section 2.1 we can define the mutual information of the two variables X and Y as the amount of common information contained in both X and Y; we can define MI using Shannon's entropy with the following [84]:

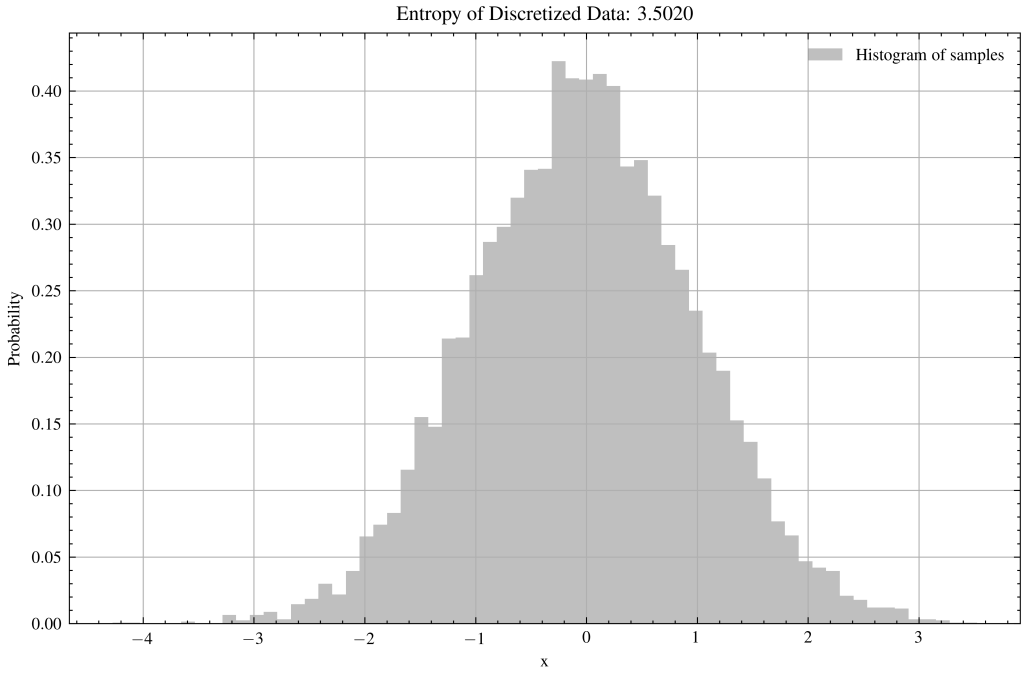

Figure 2.1: Binned representation for a variable drawn from a standard normal distribution. The entropy is calculated algebraically by computing probabilities on each bin.

$$I(X;Y) = H(X) + H(Y) - H(X,Y) = H(Y) - H(Y \mid X) = H(X) - H(X \mid Y) \quad (2.1)$$

MI is a symmetric value; therefore, it does not indicate the direction of information flow which is in contrast to TE ($TE_{X\to Y} \neq TE_{Y\to X}$). Moreover, MI measures general dependencies between X and Y without temporal dynamics, while TE considers the temporal evolution of the variables while focusing on the temporal transitions. However, the directionality inferred by the TE offers no indication of convergence. Also, an important note here is that, when using TE, the direction of the information asymmetry can be ensured only when the TE value is zero [42].

In the course of our experiments involving the calculation of TE and MI, we adopted efficient estimation techniques due to the prohibitive computational demands of algebraic computation. We utilized the Scikit-Learn library, for MI estimation in neural network layer interactions aimed at enhancing accuracy and training efficiency. Among the various MI estimation methods, histogram-based approaches offer simplicity but are sensitive to bin size and suffer from dimensionality curses, while k-nearest neighbor methods, adaptively address data density variations. Variational approaches, have streamlined MI computation for neural networks through backpropagation, excelling with large, high-dimensional datasets but requiring meticulous parameter tuning. Dual density ratio estimation,

avoiding direct density estimation, has shown practical advantages, particularly in high-dimensional scenarios. Despite the plethora of estimators, each encounters challenges related to distribution shapes and input variable relationships, with sparse interactions and long-tailed distributions posing significant biases. High MI variables necessitate substantial sample sizes for precision, with neural estimators demonstrate proficiency in handling high MI values. Our investigations across diverse neural architectures revealed consistent patterns mirroring these estimation complexities.

2.3 Kullback-Leibler Divergence (Relative Entropy)

The Kullback-Leibler (KL) divergence, introduced by Kullback and Leibler in 1951 [52], measures the asymmetrical information loss when approximating distribution Q with P. Essential in AI for optimizing models, particularly in variational inference, Bayesian models, and unsupervised learning, it is pivotal for classification tasks. Defined as $D_{KL}(P\|Q) = \sum_{x\in X} P(x) \log \frac{P(x)}{Q(x)}$, estimating KL divergence in high dimensions or with unknown distributions requires methods like Monte Carlo estimation, histogram-based methods, kernel density estimation, variational methods, and neural network-based techniques such as VAEs and GANs. Comparing KL divergence with cross-entropy, the latter can be seen as the sum of the true entropy $H(y)$ and the KL divergence $D_{KL}(y\|\hat{y})$. In classification tasks, minimizing cross-entropy effectively minimizes the KL divergence, aligning the predicted distribution with the true one, due to $H(y)$ being a constant for one-hot encoded true labels. This statistical foundation supports the widespread use of cross-entropy in classification problems for its effectiveness in quantifying prediction errors relative to ground truth.

CHAPTER 3

Background

3.1 Transfer Entropy

Transfer Entropy (TE), introduced by Thomas Schreiber [81], is a measure for directional information transfer, extending Shannon's entropy to analyze complex systems dynamically without assuming linearity. TE is critical in AI, notably in feature selection for time series forecasting, examining information flow within neural networks, and studying dynamics in computational neuroscience. Unlike Granger causality (GC), which assumes linear and Gaussian relations between time series, TE captures non-linear relationships. GC, initially for econometrics, measures causality through vector autoregression models, distinguishing between unrestricted and restricted forms. Despite initial interest in integrating GC for adjusting weights in neural networks based on correlations between neuron activations, empirical tests showed limited effectiveness, partly due to dataset shuffling and batch training. However, GC showed stability when used as a parameter in gradient descent, albeit requiring more epochs for comparable accuracy.

$$te_{j,i}^{r,n} = \sum_{s_i^{r,n+1},\, s_i^{r,n},\, s_j^{r,n}} p(s_i^{r,n+1},\ s_i^{r,n},\ s_j^{r,n})\ log \frac{p(s_i^{r,n+1},\ s_i^{r,n},\ s_j^{r,n})\ p(s_i^{r,n})}{p(s_i^{r,n+1},\ s_i^{r,n})\ p(s_i^{r,n},\ s_j^{r,n})} \tag{3.1}$$

TE distinguishes itself from mere correlation by incorporating temporal precedence, a crucial criterion for inferring causality [82]. While causality examines the impact of interventions, information transfer measures predictability of state transitions [61]. TE quantifies directional information flow using the Kullback-Leibler distance, measuring deviations from the generalized Markov property. In multilayered neural networks, TE assesses the volume of information transferred between layers during training, where the output of one layer influences the next. The expanded TE formula (3.1) calculates probabilities for neuron activations, reflecting the causal relationship between layers. TE's ability to evaluate dependencies between neurons or groups thereof makes it valuable for

measuring data compression quality in neural networks [67]. Through TE, researchers can gain insights into the intricate dynamics of neural network information processing, enhancing understanding and optimization of AI models.

Estimating Transfer Entropy

Accurate computation of TE for extensive time series data poses significant computational challenges, particularly due to the complexity of entropy-based measure estimations [26]. Three predominant methods address these challenges: k-Nearest Neighbors (k-NN), Discretization, and Kernel Density Estimation (KDE). k-NN estimates probabilities based on distances to nearest neighbors, handling high-dimensional spaces and nonlinear dependencies efficiently but is sensitive to k and distance metrics. Discretization simplifies probability computations by binning data, reducing computational load but risks information loss due to discretization bias. Binarization, a specific form of discretization, proved effective for our studies [65–67], enabling efficient TE computation for neural network applications despite minor accuracy trade-offs.

$$K_h(x) = \frac{1}{h} K\left(\frac{x}{h}\right) \tag{3.2}$$

KDE offers smoother probability estimates, placing kernel functions on data points and summing them to estimate density ($\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right)$). However, KDE faces exponential growth with dimensionality and is sensitive to bandwidth selection (3.2). In our study [68], a KDE-based tool [39] facilitated flexible and robust TE computation, leveraging K-D trees for efficient KDE in high dimensions. K-D trees partition the data space, improving KDE efficiency to $O\left(n \log n + m \cdot n \cdot C_{\text{search}}\right)$ for low-dimensional, short inputs, where n is the number of data points, m is the number of nearest-neighbor searches, and C_{search} is the search cost. We found a lag of 1 optimal for enhancing accuracy in Graph Neural Networks (GNNs).

A comprehensive review of TE computation tools revealed difficulties in achieving consistent outputs across libraries, even with small datasets. Notably, the baseline framework proposed by [59], optimized for binary data, encountered performance issues when interfaced with Python from its original Java implementation. Nonetheless, this framework served as a foundational reference for our binarization strategy, which significantly accelerated TE computation while accommodating the slight error bias introduced by data simplification. The exploration of these methods underscores the ongoing quest for efficient and accurate TE estimation in complex datasets, highlighting the importance of choosing the right estimation method based on the characteristics of the data and the specific requirements of the application.

3.2 Information Bottleneck

The Information Bottleneck (IB) theory, intermittently explored in machine learning and AI, offers insights into neural network behaviors and foundation models. Rooted in Rate Distortion Theory, pioneered by Shannon [85] and refined by others [18, 23], this theory is pivotal in communications and engineering, with applications in machine learning from an information-theoretic viewpoint. Neural networks learn input distributions through estimations, involving lossy processes and quantization steps that introduce distortion. The distortion-rate trade-off is depicted in Figure 3.1, illustrating the compromise between rate (R) and distortion (D), with 0 distortion corresponding to 100% accuracy. Neural network performance, including accuracy and generalization, is inherently limited by architectural constraints affecting distortion metrics.

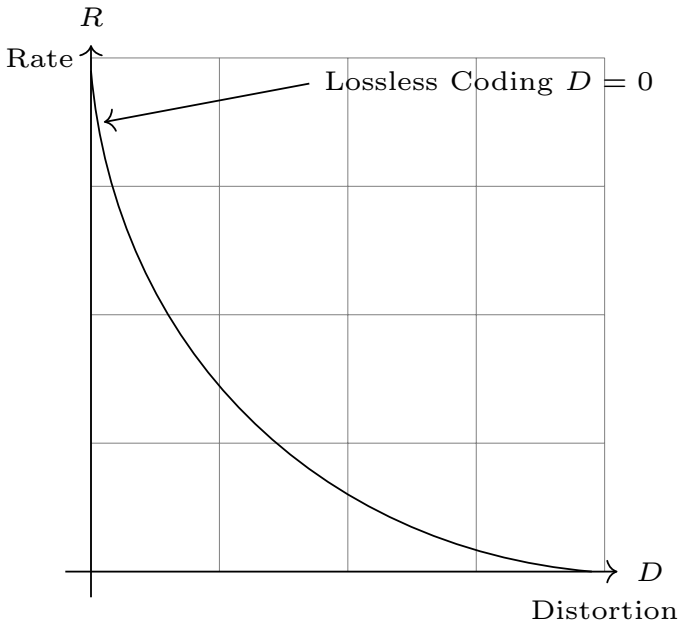

Figure 3.1: This plot has been inspired from Bernd Girod's class on Image and Video Compression handouts: EE368b Image and Video Compression Rate Distortion Theory no. 2

Rate Distortion Theory seeks the minimum rate required for accurate inference through a neural network, given a distortion threshold. Considering $X \sim \mathcal{N}(\mu, , \sigma^2)$, with R bits encoding a symbol from X and $d(x, y) = (x^2 - y^2)$ as the squared error distortion measure, the rate distortion function minimizes Mutual Information (MI) between X and reconstructed Y under the distortion constraint. The optimization targets all conditional distributions $p(x \mid y)$ satisfying $\mathbb{E}[d(X, Y)] \leq D$, aiming to minimize MI while reducing the rate necessary to meet a predefined distortion level. Shannon's lower bound for squared error distortion is given by encapsulating the theory's objective to balance information preservation and transmission efficiency in neural network learning processes.

Information Bottleneck Method

The Information Bottleneck (IB) method, distinct from Rate Distortion Theory, focuses on identifying the most pertinent information from one variable for predicting another. This is achieved by introducing a bottleneck variable T, representing the compressed representation of X, to store as much information as possible about Y (3.3). The parameter β balances the compression and prediction.

$$min_{P_{T|X}}(I(X;T) - \beta I(Y;T)) \tag{3.3}$$

Extensive research ([2, 14, 29, 31, 43, 79, 83, 89, 90, 94, 95]) has explored IB in neural networks, constructing information planes from layer activations to reveal fitting and compression phases, demonstrating the importance of maintaining a balance between compression and prediction for improved accuracy.

Studies have directly utilized IB in various mechanisms, from model selection to improving training time, accuracy, and generalization performance ([14, 79]). Deeper architectures are found to offer better trade-offs for image classification tasks, attributed to their capacity to preserve more relevant information. The Blahut-Arimoto algorithm ([4, 7]) facilitates the computation of IB for variables X and Y with positive $I(X;Y)$. Clustering algorithms modified with IB achieve higher compression levels ([90]), while complexity-performance trade-offs are optimized ([27]). Deep learning breakthroughs ([95]) reveal distinct training phases tied to network performance, with stochastic gradient descent's random diffusion behavior aiding compression.

IB's role in compression and prediction is nuanced, with activation functions significantly impacting compression capabilities ([79]). Compression is often observed only in classification layers, with higher layer indices exhibiting lower MI variance. IB as a layer-wise loss function ([22]) and precise $I(X;T)$ estimation ([29]) enhance understanding of class clustering and compression mechanisms. Multi-view unsupervised learning ([25]) extends IB's utility, while generalization errors correlate with IB degree ([43]). Advancements in generative models ([88]) and linear regularized DNNs on Gaussian classification tasks ([34]) further elucidate IB's multifaceted role in deep learning architectures and algorithms.

3.3 Graph Neural Networks

Graph Neural Networks (GNNs), a frontier in deep learning, excel in handling relational and graph-related data, addressing tasks from node classification to link prediction and graph generation ([5, 6, 13, 15, 20, 21, 32, 33, 37, 40, 41, 44, 58, 73, 74, 78, 80, 91, 98, 105, 114, 115, 118, 119]). Architecturally, GNNs encompass a spectrum from Graph Convolutional

Networks (GCNs) to Graph Attention Networks (GATs), Graph Autoencoders (GAEs), Graph Generative Adversarial Networks (GGANs), Graph Recurrent Neural Networks (GRNNs), Graph Transformers, Graph Isomorphism Networks (GINs), and specialized GNNs utilizing edge or node features ([47, 48, 96, 99, 106, 111, 113]). Methodologically, they range from spectral-based to spatial-based, message-passing neural networks (MPNNs), deep learning, reinforcement learning, transfer learning, inductive and transductive learning methods, adapting to various graph complexities.

Sperduti *et al.* [92] initiated graph-like neural networks, evolving to Gori *et al.*'s Graph Neural Network [30]. Scarselli *et al.* [80] refined the framework, integrating topology and node features. Bruna *et al.* [9] introduced spectral networks, paving the way for localized spectral filtering by Defferrard *et al.* [17], culminating in Kipf *et al.*'s seminal work [48] that simplified architectures for semi-supervised classification, setting the foundation for contemporary GCNs. Innovations continue, with studies enhancing spectral convolutions and signal processing in graphs ([17, 50, 54, 87]). GNNs' versatility and scalability across interdisciplinary applications have surged research interest, making them indispensable for complex, interconnected data tasks. The forthcoming section delves into graph convolutional networks, their challenges, and solutions, guided by Kipf *et al.*'s contributions [48] to Yan *et al.'s* recent advancements [108].

3.3.1 Graph Convolutional Neural Networks

Summarizing graph components: $G = (V, E)$ denotes a graph with node set V and edge set E; adjacency matrix A reflects node connections; degree of node v_i, d_i, is its edge sum; degree matrix D has diagonal elements $D_{ii} = \sum_j A_{ij}$; incidence matrix K shows vertex-edge connections; $X \in \mathbb{R}^{n \times o}$ represents node feature matrix with o as feature dimension; $\mathbf{x}_v$ signifies node (v)'s feature vector; $X^{(l)}$ and $\mathbf{x}^{(l)}$ denote layer l feature matrices.

Graph Laplacian Matrix

The graph Laplacian matrix, pivotal in graph networks and signal processing, encapsulates node connectivity and signal smoothness. Derived as $L = D - A$, it mirrors the second derivative of a function, assessing value changes between connected vertices.

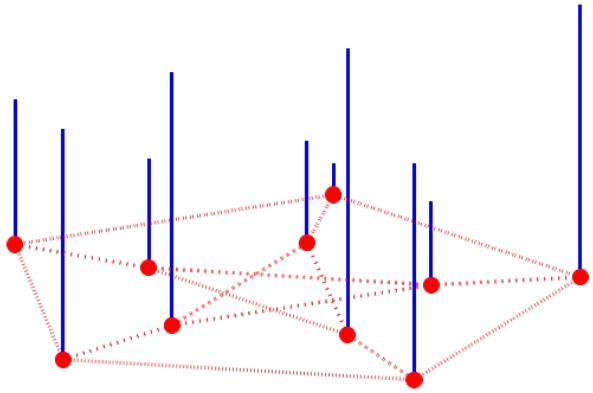

Figure 3.2: A positive graph signal on top of a Petersen graph (from [87]). Each node has an associated signal that is proportional with the blue bar height.

Visualized as a 'flexible cover' responsive to vertex signals, as illustrated in Figure 3.2, the Laplacian's eigenvectors vary with frequency across the graph ([87]). Beyond measuring connectivity and diffusion, it's crucial for cluster identification in graph analysis.

Graph Convolutions

Graph convolutions, unlike traditional CNNs, grapple with the irregular structure of graph data, lacking predefined node order or consistent neighbor counts [56, 57, 87, 112]. Transforming graph structure into the spectral domain facilitates convolutional operations, offering several advantages: decoupling inputs into frequencies for tailored processing, computational efficiency via graph Laplacian diagonalization, invariance to node ordering, global structure capture, and design flexibility for frequency-specific filters. However, spatial locality is less exploited spectrally, and dynamic graphs necessitate frequent spectral decomposition recalculations. Spectral graph convolutions involve transforming signals into the frequency domain, applying filters, and reverting to the spatial domain, utilizing the graph Fourier transform $\hat{x} = U^\top x$ and its inverse $x = U\hat{x}$. Convolution in the vertex domain corresponds to spectral domain multiplication $\hat{y} = \hat{g} \odot \hat{x}$, with spectral filters approximated using truncated Chebyshev polynomials for computational efficiency ($g_\theta(\Lambda) \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{\Lambda})$, $g_\theta \star x \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{L})x$). This approximation allows direct spatial domain computation, avoiding costly eigendecomposition, bridging the gap between graph Laplacians and convolutions, and enabling scalable Graph Convolutional Networks (GCNs) [17, 48].

In essence, the graph Laplacian's eigendecomposition underpins the graph Fourier transform, facilitating spectral representation and processing of graph signals. Efficient convolution and filtering are achieved through spectral domain multiplication, with polynomial approximations streamlining computations for large-scale graph analysis, thus

enhancing GCN scalability and applicability.

Multi-layer GCNs

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{3.4}$$

Multi-layer Graph Convolutional Networks (GCNs), pioneered by Kipf *et al.* [48], apply layer-wise convolutions to aggregate node features from local neighborhoods, bridging spectral and convolutional approaches without direct spectral methods. Simplifications include limiting Chebyshev polynomials to order (K=1) and employing a specific parameterization to avoid oversmoothing [8, 77, 108], introducing a renormalization trick to tackle exploding/vanishing gradients [46]. The convolution propagation rule (3.4) balances node influence on neighbors, using self-linked adjacency matrices and learnable weights. Filters, represented as $\Theta \in \mathbb{R}^{C\times F}$, process input signals $X \in \mathbb{R}^{N\times C}$ via normalized aggregation and parameter multiplication ($Z = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}X\Theta$). Multi-layer GCNs generalize this process, incorporating activation functions, as illustrated in Figure 3.3, depicting node feature aggregation through convolutional layers, where node A updates its properties using aggregated features from its neighborhood, showcasing message passing in GNNs.

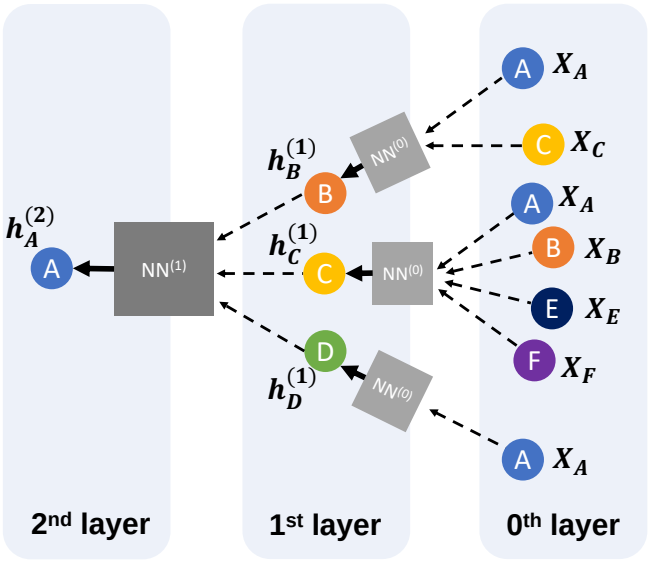

Figure 3.3: Node feature aggregation using a two layers convolution(graphic from [110]). Node A updates is own properties using aggregated features from all the other nodes.

Oversmoothing, Heterophily and Homophily

In [68] we tackled some of the well-known and still open problems in the GNN world: oversmoothing on the architecture and optimization end, and heterophily and homophily

on the graph's connectivity properties on the other end. The latter two are attributes of the components of a graph such as nodes or subgraphs. All three have a critical impact on a GCN discriminative capabilities.

Oversmoothing

Oversmoothing is a phenomenon observed in GCNs where node representations become increasingly similar as the number of layers increases. It is not solely a product of the dataset, such as densely connected graphs, but it is also a result of the GCN architecture. This convergence to a non-informative limit hinders the performance of deep GCNs, especially on heterophilic graphs where nodes with different labels are connected [12, 56, 77]. Although oversmoothing can benefit regression and classification tasks in small amounts, excessive smoothing can be detrimental [77]. In other words, aggregation of neighborhood information can lead to nodes in close proximity being represented too similarly, even if they belong to different classes. As layers increase, the receptive field expands, potentially averaging out unique node features [72].

Mathematically, the oversmoothing problem can be understood through the eigenvalue spectrum of the graph Laplacian. In deep GCNs, the eigenvalues associated with the graph Laplacian can become too large, causing the feature vectors to converge to a constant vector [56]. This convergence is exacerbated by the depth of the network, as deeper layers amplify the smoothing effect.

Heterophily

Heterophily refers to the characteristic of a graph in which connected nodes are likely to have different labels or features [100, 116]. This property presents a challenge to traditional GCNs that perform well on homophilic graphs, where connected nodes tend to share similar attributes [100].

Traditional GCNs, designed with an implicit assumption of homophily, struggle to handle heterophilic graphs due to their dependence on neighbor aggregation for label prediction[64, 100], [63, 116]. The design of traditional GCN models can be considered ill-posed for heterophilic graphs, as their implicit use of homophily can exacerbate both oversmoothing and the negative effects of heterophily [100, 108, 116]. When applied to heterophilic graphs, these GCNs can experience performance degradation because the aggregation process mixes information from nodes with different labels, leading to less informative representations [63, 64, 101].

Homophily

Homophily is the tendency for nodes with similar characteristics, such as labels or features, to be connected in a graph [1, 63, 64, 116]. Many GNNs implicitly rely on this assumption, leading to limitations in the handling of heterophilic graphs [63, 64].

And probably the obvious question by now is if these properties are caused by GCN design, inherent to datasets, or both? The occurrence of oversmoothing, heterophily, and homophily stems from both the inherent properties of graph datasets and the design of GCN models. Both *heterophily and homophily are dataset characteristics*. Real-world graphs often exhibit varying degrees of heterophily, depending on the nature of the relationships between nodes [63], [101]. Homophily is prevalent in social networks, citation networks, and other domains where similar entities tend to connect [64], [63], [75].

Mitigations

From the theoretical bounds of the heterophily and homophily, these are not inherently bounded; yet they are bounded by the GCN design and the characteristics of the dataset. These are still active research areas. For example [69] suggests that the choice of kernel in spectral graph convolutions can influence the degree of oversmoothing. Similarly, the concept of "mixing time" in random walks on graphs can provide insights into how quickly heterophily or homophily can manifest in GCN predictions [11]. [12] showed that designing deeper GCNs with fewer layers can help reduce oversmoothing, while [96] introduced graph attention mechanisms to selectively focus on important neighbors and mitigate oversmoothing [55].

However, the impact of these properties can be mitigated through specialized GNN models and techniques. These include adaptive aggregation schemes [75, 100], considering edge directions and node dissimilarities, incorporating high-order neighbor information [100, 104, 120], employing techniques like ego-embedding and neighbor-embedding separation [75], graph attention [96], graph sampling [75, 100, 104] have shown important improvements. GCNs can leverage homophily for better performance on homophilic graphs by emphasizing local neighborhood aggregation. However, excessive expectation of homophily presence can be detrimental to heterophilic graphs [120]. Oversmoothing can be bounded by limiting the number of GCN layers, adding residual connections and dilated convolutions, employing residual connections, using skip links, implementing new normalization strategies, or incorporating edge dropout [108], or consider models that can use both local and global contexts [120]. These techniques help preserve node feature diversity and prevent convergence to a constant value [107, 108].

Although beneficial in homophilic settings, excessive homophily expectations should be avoided. Techniques like adaptive channel mixing can help balance between aggregation,

diversification, and identity channels to address different homophily situations [62, 63].

In conclusion, understanding and addressing oversmoothing, heterophily, and homophily is crucial for developing effective GCN models. While oversmoothing can be mitigated through architectural modifications, handling heterophily and leveraging homophily effectively require specialized techniques and adaptive approaches. Future research should focus on developing more robust and generalizable GCN models that can effectively learn from graphs with varying levels of homophily and heterophily.

3.3.2 Graph Regularization Techniques

Addressing oversmoothing, heterophily, and homophily in Graph Convolutional Networks (GCNs) is pivotal for enhancing their discriminative capabilities. Oversmoothing, characterized by increasingly indistinguishable node representations in deeper layers, undermines GCN performance, especially in heterophilic graphs where nodes of differing labels are connected [12, 56, 77]. Mathematically, it arises from the graph Laplacian's eigenvalue spectrum, where excessively large eigenvalues lead to feature vector convergence [56]. Heterophily, the tendency for connected nodes to have dissimilar labels, challenges traditional GCNs designed under homophily assumptions, leading to performance degradation [64, 100]. Conversely, homophily, where similar nodes are connected, is often implicitly relied upon by GCNs, posing limitations in heterophilic graph handling [63, 64]. Mitigation strategies include adaptive aggregation schemes, edge-aware mechanisms, high-order neighbor consideration, and techniques like ego-embedding and graph attention [75, 96, 100]. Regularization techniques, particularly Laplacian-based and non-Laplacian-based methods, play a crucial role in bounding oversmoothing, with Laplacian regularization promoting similarity among neighbors [3, 109] and PairNorm [117] maintaining pairwise feature distances across layers. DropEdge [76], a non-Laplacian technique, introduces edge removal for data augmentation, enhancing robustness and preventing overfitting in deep GCNs.

The interplay between oversmoothing, heterophily, and homophily in GCNs is multifaceted, influenced by both dataset properties and GCN architecture. Oversmoothing's manifestation can be exacerbated by the GCN design, especially in densely connected graphs, leading to a loss of node feature distinctiveness [72]. Heterophily and homophily, intrinsic dataset characteristics, challenge GCN assumptions, necessitating adaptive approaches for effective performance. Laplacian-based regularization techniques, like the one proposed by Ando *et al.* [3], aim to maintain label similarity among neighbors, offering benefits in feature preservation but with limited impact on GNNs already capturing structural information. PairNorm, a non-Laplacian regularization method, ensures consistent pairwise feature distances, preventing feature mixing across clusters without altering the network architecture [117]. DropEdge, by inducing edge removal, augments data diversity and combats oversmoothing in deep GCNs, indirectly modifying the Laplacian to maintain

feature diversity [76].

In summary, the challenges posed by oversmoothing, heterophily, and homophily in GCNs underscore the need for adaptive and regularization strategies. Laplacian-based techniques, such as the graph Laplacian regularizer, aim to preserve graph structure by encouraging feature similarity among adjacent nodes [102, 121]. PairNorm's node-level and edge-level penalties promote local and global graph structure preservation, ensuring feature vectors remain smooth across the graph. DropEdge's edge removal mechanism introduces randomness and diversity, preventing overfitting and oversmoothing in deep GCN architectures. Future research directions should focus on developing more sophisticated regularization and adaptive aggregation schemes to enhance GCN robustness and generalizability across diverse graph properties.

CHAPTER 4

Transfer Entropy with Shallow Neural Networks

This chapter introduces an innovative training algorithm, *FF+FB*, for feedforward neural networks that harnesses causal relationships through Transfer Entropy (TE) feedback, aiming to enhance learning efficiency [65]. TE, traditionally used to quantify effective connectivity between neurons [24, 60, 86, 97], is repurposed to measure information transfer between adjacent layers, amplifying connection relevance. Unlike previous applications of TE in neural networks [35, 71], our approach integrates TE directly into the backpropagation weight update process, refining the standard algorithm by incorporating a feedback mechanism that considers the TE between neuron pairs.

FF+FB is structured in two stages: Stage (I) computes TE values during training, storing them for all neuron pairs; Stage (II) retrains the network using these stored values, modifying the gradient descent to incorporate the TE feedback (Eq. 4.1). This adaptation accelerates the learning process and improves accuracy, as evidenced by the XOR problem, where *FF+FB* achieves 100% training accuracy in significantly fewer epochs than *FF* (7-10 times less, on average 62.2 epochs vs. 349.9 epochs). On ten UCI datasets [19], *FF+FB* also demonstrates superiority, reaching target accuracies faster and achieving higher test set accuracy in most cases.

$$\Delta w_{ij}^{l} = -\eta \frac{\partial C}{\partial w_{ij}^{l}} (1 - te_{j,i}^{l}) \tag{4.1}$$

Hyperparameters play a crucial role in *FF+FB*'s performance. The learning rate η and binning threshold g require careful tuning; *FF+FB* often benefits from smaller η values, indicating a more targeted learning approach. However, small η and g can trap the network in local minima, necessitating a judicious choice of g via grid search. Control experiments, including modifications to the te values and their weighting, confirm the

algorithm's robustness and highlight its adaptability in compensating for suboptimal η selections.

FF+FB's computational overhead, primarily in Stage (I), is offset by its superior training efficiency and accuracy gains. The algorithm's performance on the *car* and *glass* datasets, despite initial challenges, validates its competitiveness with established packages like Weka. Practical considerations suggest that the increased computational cost during training stage (I) is inconsequential for inference tasks, as trained weights can be stored, encapsulating the te values. This makes *FF+FB* viable for real-world applications, even with large datasets, as te values computed in Stage (I) can be reused in Stage (II) without additional overhead. Alternative TE estimation techniques might alleviate computational burdens in such scenarios [10].

In conclusion, *FF+FB* represents a significant advancement in neural network training algorithms, leveraging TE to quantify and enhance causal relationships between neurons. It not only reduces the number of epochs required for training and improves accuracy but also offers stability and resilience to local minima, as shown in the plots from the full version of the thesis. Optimizing the (g) threshold can mitigate the impact of other hyperparameters, suggesting a pathway to more efficient model tuning. The potential for *FF+FB* to facilitate knowledge extraction and explanations from trained networks, albeit left as an open problem, underscores its broader implications for understanding neural network decision-making processes.

CHAPTER 5

Transfer Entropy in Convolutional Neural Networks

Our work extends the application of Transfer Entropy (TE) to Convolutional Neural Networks (CNNs), aiming to enhance training mechanisms and interpretability [66]. TE, a measure of directional information transfer, quantifies relationships between neuron outputs in adjacent layers, acting as a smoothing factor that stabilizes the learning process and accelerates convergence. Unlike symmetric mutual information, TE's asymmetry aligns with the causal nature of neural network layers. Inspired by [35], our novel approach integrates TE directly into backpropagation, updating weights according to the TE between neuron pairs. This differs from Herzog *et al.*'s method, which used TE for structuring feedback connections post-training [36]. Our experiments on CNNs with TE feedback demonstrate improved performance and stability, particularly in the last two fully connected layers, akin to fine-tuning classification mechanisms (Figure 5.1).

TE computation in CNNs, particularly for large timeseries, is computationally intensive [26]. We optimize the TE integration by limiting timeseries lengths and using a sliding window technique over batches, which maintains accuracy while managing overhead. The window length s, which ideally matches the batch size, facilitates smoother TE values and favorable accuracy trends. Our focus on the last two fully connected layers, as opposed to convolutional layers, reduces computational demands while significantly impacting accuracy, akin to dropout's effect but with a performance enhancement focus. Experimentally determined parameters and the adaptive nature of TE as a meta-parameter contribute to the algorithm's robustness and stability.

The experimental results which can be read on the complete thesis, showcase the efficiency of TE in accelerating CNN training to reach target accuracies with fewer epochs. We observed that using TE feedback for an additional layer pair improves performance but at a cost of exponentially increasing computational overhead. The optimal performance-

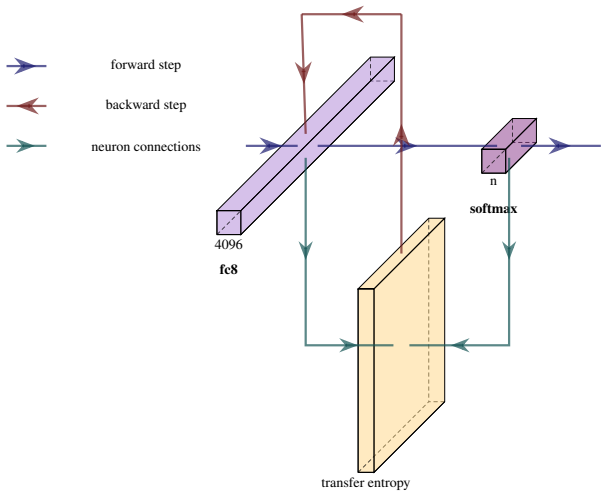

Figure 5.1: During the feedforward step, we compute timeseries I and J, and the **te** matrix, as shown by the green arrows. When the backward step propagates the errors, the **te** matrix is used in the weight updates.

overhead trade-off is application-dependent, requiring careful consideration of TE's role as a slowly changing meta-parameter. Our experiments with pre-trained networks, where only the last two layers undergo TE correction, yield inconsistent results, indicating the importance of TE's synchronized integration with the backpropagation process.

In conclusion, our study confirms TE's utility in enhancing CNN training, particularly in the final layers, mirroring its effectiveness in simple feedforward networks [65]. The computational overhead is mitigated by focusing on a subset of neuron pairs, aligning with biological neural systems' feedback structures [28, 93]. TE's role as a smoothing factor and its periodic activation contribute to the stability and generalization of the learning algorithm, similar to the hierarchy of parameters in learning neural causal models [45]. Our findings suggest that TE's integration could have evolutionary parallels in real neural systems, optimizing relevance in feedforward pathways [36]. Future research could explore TE's impact on deeper network architectures and its potential for enhancing interpretability in CNNs.

The experiments carried out on a range of well-established datasets (CIFAR-10 [51], FashionMNIST [103], STL-10 [16], SVHN [70], and USPS [38]) using a standardized CNN architecture and hyperparameters underscore TE's effectiveness. The positive influence of TE feedback on training stability and accuracy is evident, with notable gains even when using a fraction of neuron pairs (10%) from the last two fully connected layers. This efficiency hints at the potential for TE to serve as a tool to understand and optimize the dynamics of neural networks, akin to the insights gained from the study of biological neural systems. The trade-off between TE's benefits and computational overhead is a key consideration, guiding the optimal use of TE in real-world applications.

CHAPTER 6

Transfer Entropy in Information Bottleneck

MI and TE offer distinct insights into neural network dynamics, and TE uniquely captures directional and temporal information flow [67]. Our study pioneers the use of TE to quantify information transfer between neural layers, revealing its potential to enhance training efficiency and shed light on the compression-generalization relationship. By measuring TE across adjacent layers, we observe a fitting-compression pattern akin to the Information Bottleneck (IB) principle, with TE values peaking early and diminishing as training advances. This trend supports the hypothesis that initial epochs focus on fitting, followed by a compression phase where the network refines and retains generic features. TE's dynamic nature as a metric, particularly its sensitivity to network architecture and efficiency, aligns with the notion that optimized architectures facilitate better compression [22, 29].

Experimentally, we employed shallow feedforward networks and CNNs optimized for various datasets, including UCI's glass, ionosphere, seeds, divorce, liver disorders, and Iris, alongside FashionMNIST, STL-10, SVHN, and USPS for CNNs ([65, 66]). TE was computed for all adjacent layers and neurons, with the first 5% of each epoch excluded to stabilize neuron activations. Dynamic thresholding, based on a 95th percentile of activation values, ensured TE's relevance throughout training. The observed trend of decreasing TE over epochs and the confirmation of higher TE in final layers for shallow networks parallel the compression phases identified in IP analyses [79, 89]. In CNNs, focusing on the last two fully connected layers (including softmax), we replicated these findings, observing steep but smooth TE trajectories for larger datasets. We recommend observing the depicted training evolution in the plots presented in the full version of this thesis.

The direct correlation between network performance metrics—accuracy and loss—and TE fluctuations underscores TE's potential as a diagnostic tool. During the fitting phase,

TE rapidly declines, mirroring loss reduction and inversely tracking accuracy, before stabilizing in a compression phase with minimal variance. This pattern, consistent across datasets and architectures, supports the notion that TE reflects a network's learning dynamics and compression capabilities. In efficient architectures, TE exhibits smoother lines with distinctive evolution patterns, suggesting that optimized structures achieve better interlayer compression [22, 29].

TE's sensitivity to network architecture and its evolution during training offer insights into the training process's nuances, beyond accuracy and parameter count. It emerges as a promising adaptive parameter, potentially reducing the number of training epochs needed. However, the computational overhead associated with TE calculation, particularly for larger datasets and deeper networks, necessitates strategic selection of layers for analysis. Our findings on TE's utility in diagnosing training hurdles and its inverse relation to loss and accuracy align with IB's theoretical framework, validating TE as a viable alternative for IP analysis.

In conclusion, our study demonstrates a strong connection between TE evolution and network performance metrics, suggesting its role in diagnosing training dynamics and optimizing compression. While TE complements the IB principle by offering a dynamic, layer-specific perspective, its practical integration as a training enhancement or diagnostic metric requires careful consideration of computational costs. Future research could explore TE's potential in guiding network architecture design and its role in developing more interpretable and efficient neural models. The observed trends in TE, alongside accuracy and loss, provide a richer understanding of how neural networks learn and compress information, potentially informing the development of more robust training algorithms and cost functions [2, 49].

CHAPTER 7

Transfer Entropy in Graph Convolutional Networks

Our study [68] investigates Graph Convolutional Networks (GCNs) from a practical standpoint, focusing on generalization performance and addressing challenges such as oversmoothing and heterophily. We propose the TE-GGCN method, which integrates TE as a postconvolution control mechanism to improve node feature discrimination and classification accuracy. Unlike homogeneous GCN enhancements, TE-GGCN selects nodes with high heterophily and degree, computing TE to adjust their features, thus boosting discriminative capabilities without overhauling the convolutional process. This strategy, while computationally demanding, particularly for high-degree nodes, demonstrates effectiveness in mitigating oversmoothing and improving accuracy across various GCN models.

The GGCN method [108] re-calibrates edge weights based on node degrees and adjusts edge features for heterophilous and homophilous relationships. Our TE-GGCN builds upon this by computing node heterophily rates (using $\mathcal{H}_v = \frac{1}{|N(v)|} \sum_{u \in N(v)} 1\left(l_u \neq l_v\right)$) and selecting the top 5% heterophilic nodes, further narrowing to the highest 10% degree nodes among them. TE is calculated for these nodes using the usual TE equation, influencing weight updates using the $\mathbf{H}_{i,j} = \mathbf{H}_{i,j} + max(TE_{Y_j \to X_i})$ post-convolution. This selective approach ensures computational feasibility while maximizing accuracy gains, as TE values amplify classification precision for nodes transitioning between classes.

In our experiments, TE-GGCN was evaluated on a diverse set of real-world and synthetic citation network datasets, showcasing a range of homophily and heterophily levels. Our implementation, based on PyTorch and Torch Geometric, achieved competitive or superior accuracy compared to the original GGCN model, particularly on low-homophilic datasets like Texas, Wisconsin, and Cornell. However, the computational overhead varied significantly, with PubMed and Squirrel requiring up to five times more training time due to their high-degree nodes. Computing TE within each convolutional layer offered

higher accuracy but was impractical due to prohibitive computational costs. For validation accuracy results we recommend examining the full version of this thesis.

TE-GGCN's performance hinges on its ability to identify and correct high node variances, applying the highest calculated TE value as a feature adjustment post-convolution. This method, consistent with our prior research [66, 67], improves existing GCN models without requiring complex modifications. The trade-off between accuracy and computational overhead is manageable, offering a practical avenue to improve GCN performance on classification tasks.

In conclusion, TE-GGCN showcases the potential of TE as a sensitive metric for identifying similar connectivity patterns and distributions among node features. By incorporating TE values alongside heterophily and degree metrics, we refine GGCN's classification capabilities, particularly for nodes prone to misclassification. This straightforward enhancement to GCN models, while computationally intensive for dense graphs, offers a flexible method to boost accuracy without disrupting established GCN mechanisms. Future directions could explore computational optimizations to extend the applicability of TE-GGCN to larger, more complex datasets while maintaining its performance benefits.

CHAPTER 8

Final Remarks

This chapter summarizes the key findings and contributions of our research across multiple studies. Our work has focused on improving neural network training algorithms, particularly through the use of Transfer Entropy (TE) feedback and its applications in various neural network architectures.

We introduced the neural training algorithm *FF+FB*, which utilizes TE to quantify the relationships between neurons and uses it as feedback to improve certain neural connections. This method has demonstrated several advantages: it generally requires fewer training epochs while achieving higher accuracy compared to standard feedforward (FF) networks; exhibits more stable behavior during the training process; it is less susceptible to local minima. In addition, the use of TE feedback has shown potential in reducing the effort required to optimize hyperparameters: the threshold parameter g can decrease the importance of other hyperparameters, such as the learning rate η and the number of hidden neurons. This approach may facilitate easier design and training of network architecture.

Following *FF+FB* findings, our research was extended to Convolutional Neural Networks (CNNs), where we found:

- It is efficient to consider only the interneural information transfer of a random subset of neuron pairs from the last two fully connected layers
- Information transfer within these layers has the most significant impact on the learning process
- Many inter-neural information transfer connections appear redundant, allowing the use of only a fraction of them

We observed that TE acts as a smoothing factor in our models since it becomes active periodically, not after each input sample is processed. Also TE can be considered a slowly

changing meta-parameter, relating to the hierarchy of quickly-changing vs. slowly-changing parameters in learning neural causal models.

Our research also extended to Graph Convolutional Network (GCN) method through our *TE-GGCN* algorithm, where:

- We demonstrated improvements by leveraging node heterophily, degree metrics, and TE values
- Utilized TE as a measure of high node variances, applying the highest TE value calculated in a forward pass as an adjustment to node features, post-convolution
- This TE-based correction, applied prior to the softmax classification layer, offers a versatile and easy way to improve existing GCN implementations

Our approach could potentially facilitate the extraction of knowledge and explanations from trained networks using the causality paradigm, although this remains an open problem for future research. The observations regarding information transfer in neural networks, particularly in CNNs, can be further discussed from a neuroscientific perspective, drawing parallels to structures in the vertebrate brain. Our findings align with speculations that evaluation of the relevance of different feedforward pathways could have been a phylo- or ontogenetic driving force for the design of feedback structures in real neural systems.

Although TE feedback accelerates the training process by reducing the number of required epochs, it adds computational overhead to each epoch. The optimal balance between these factors is application-dependent. The addition of TE in the learning mechanism generates new hyper-parameters, raising questions about potential overfitting and generalization performance. However, our experiments suggest that TE's role as a slowly changing meta-parameter may mitigate these concerns. While our work primarily focused on TE, future research could explore alternative cost functions, as some researchers have suggested that Information Bottleneck (IB) loss may not always behave optimally.

References

[1] S. Abu-El-Haija, B. Perozzi, A. Kapoor, N. Alipourfard, K. Lerman, H. Harutyunyan, G. V. Steeg, and A. Galstyan. MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 21–29. PMLR, 09–15 Jun 2019.

[2] R. Amjad and B. C. Geiger. Learning representations for neural network-based classification using the information bottleneck principle. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(09):2225–2239, sep 2020.

[3] R. Ando and T. Zhang. Learning on graph with laplacian regularization. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

[4] S. Arimoto. An algorithm for computing the capacity of arbitrary discrete memoryless channels. *IEEE Transactions on Information Theory*, 18(1):14–20, 1972.

[5] J. Bastings, I. Titov, W. Aziz, D. Marcheggiani, and K. Sima'an. Graph convolutional encoders for syntax-aware neural machine translation. *arXiv preprint arXiv:1704.04675*, 2017.

[6] R. v. d. Berg, T. N. Kipf, and M. Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.

[7] R. Blahut. Computation of channel capacity and rate-distortion functions. *IEEE Transactions on Information Theory*, 18(4):460–473, 1972.

[8] C. Bodnar, F. Di Giovanni, B. Chamberlain, P. Lió, and M. Bronstein. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 18527–18541. Curran Associates, Inc., 2022.

[9] J. Bruna, W. Zaremba, A. Szlam, and Y. Lecun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*, 2014.

[10] A. Caţaron and R. Andonie. Transfer information energy: A quantitative indicator of information transfer between time series. *Entropy*, 20(5), 2018.

[11] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view, 2019.

[12] M. Chen, Z. Wei, Z. Huang, B. Ding, and Y. Li. Simple and deep graph convolutional networks. In H. D. III and A. Singh, editors, *37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1725–1735. PMLR, 13–18 Jul 2020.

[13] Z. Chen, X. Li, and J. Bruna. Supervised community detection with line graph neural networks. *arXiv preprint arXiv:1705.08415*, 2017.

[14] H. Cheng, D. Lian, S. Gao, and Y. Geng. Utilizing information bottleneck to evaluate the capability of deep neural networks for image classification. *Entropy*, 21(5), 2019.

[15] W.-L. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C.-J. Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 257–266, New York, NY, USA, 2019. Association for Computing Machinery.

[16] A. Coates, A. Ng, and H. Lee. An analysis of single-layer networks in unsupervised feature learning. *Journal of Machine Learning Research - Proceedings Track*, 15:215–223, 01 2011.

[17] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[18] A. G. Dimitrov and J. P. Miller. Neural coding and decoding: communication channels and quantization. *Network: Computation in Neural Systems*, 12(4):441, aug 2001.

[19] D. Dua and C. Graff. UCI machine learning repository, 2017.

[20] D. K. Duvenaud, D. Maclaurin, J. Iparraguirre, R. Bombarell, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015.

[21] A. Eetemadi and I. Tagkopoulos. Genetic Neural Networks: an artificial neural network architecture for capturing gene expression relationships. *Bioinformatics*, 35(13):2226–2234, 11 2018.

[22] A. Elad, D. Haviv, Y. Blau, and T. Michaeli. Direct validation of the information bottleneck principle for deep nets. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 758–762, 2019.

[23] W. Equitz and T. Cover. Successive refinement of information. *IEEE Transactions on Information Theory*, 37(2):269–275, 1991.

[24] H. Fang, V. Wang, and M. Yamaguchi. Dissecting deep learning networks—visualizing mutual information. *Entropy*, 20(11), 2018.

[25] M. Federici, A. Dutta, P. Forré, N. Kushman, and Z. Akata. Learning robust representations via multi-view information bottleneck. *arXiv preprint arXiv:2002.07017*, 2020.

[26] D. Gencaga, K. H. Knuth, and W. B. Rossow. A recipe for the estimation of information flow in a dynamical system. *Entropy*, 17(1):438–470, 2015.

[27] R. Gilad-Bachrach, A. Navot, and N. Tishby. An information theoretic tradeoff between complexity and accuracy. In *Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003, Washington, DC, USA, August 24-27, 2003. Proceedings*, pages 595–609. Springer, 2003.

[28] C. D. Gilbert and W. Li. Top-down influences on visual processing. *Nature Reviews Neuroscience*, 14(5):350–363, 2013.

[29] Z. Goldfeld, E. Van Den Berg, K. Greenewald, I. Melnyk, N. Nguyen, B. Kingsbury, and Y. Polyanskiy. Estimating information flow in deep neural networks. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2299–2308. PMLR, 09–15 Jun 2019.

[30] M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE international joint conference on neural networks, 2005.*, volume 2, pages 729–734. IEEE, 2005.

[31] H. Hafez-Kolahi and S. Kasaei. Information bottleneck and its applications in deep learning. *arXiv preprint arXiv:1904.03743*, 2019.

[32] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[33] M. Hanik, M. A. Demirtaş, M. A. Gharsallaoui, and I. Rekik. Predicting cognitive scores with graph neural networks through sample selection learning. *Brain Imaging and Behavior*, 16(3):1123–1138, Jun 2022.

[34] H. He, C. L. Yu, and Z. Goldfeld. Information-theoretic generalization bounds for deep neural networks. pages 1–25, 2024.

[35] S. Herzog, C. Tetzlaff, and F. Wörgötter. Transfer entropy-based feedback improves performance in artificial neural networks. *CoRR*, abs/1706.04265, 2017.

[36] S. Herzog, C. Tetzlaff, and F. Wörgötter. Evolving artificial neural networks with feedback. *Neural Networks*, 123:153 – 162, 2020.

[37] B. Huang and K. M. Carley. Syntax-aware aspect level sentiment classification with graph attention networks. *arXiv preprint arXiv:1909.02606*, 2019.

[38] J. J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

[39] K. M. Ikegwu, J. Trauger, J. McMullin, and R. J. Brunner. Pyif: A fast and light weight implementation to estimate bivariate transfer entropy for big data. In *SoutheastCon*, pages 1–6, 2020.

[40] D. D. Johnson. Learning graphical state transitions. In *International conference on learning representations*, 2022.

[41] J. Johnson, A. Gupta, and L. Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018.

[42] A. Kaiser and T. Schreiber. Information transfer in continuous processes. *Physica D: Nonlinear Phenomena*, 166(1):43 – 62, 2002.

[43] K. Kawaguchi, Z. Deng, X. Ji, and J. Huang. How does information bottleneck help deep learning? In *International Conference on Machine Learning*, pages 16049–16096. PMLR, 2023.

[44] J. Kawahara, C. J. Brown, S. P. Miller, B. G. Booth, V. Chau, R. E. Grunau, J. G. Zwicker, and G. Hamarneh. Brainnetcnn: Convolutional neural networks for brain networks; towards predicting neurodevelopment. *NeuroImage*, 146:1038–1049, 2017.

[45] N. R. Ke, O. Bilaniuk, A. Goyal, S. Bauer, H. Larochelle, C. Pal, and Y. Bengio. Learning neural causal models from unknown interventions, 2019.

[46] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[47] T. N. Kipf and M. Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

[48] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[49] A. Kolchinsky, B. D. Tracey, and S. V. Kuyk. Caveats for information bottleneck in deterministic scenarios. In *International Conference on Learning Representations*, 2019.

[50] D. Kreuzer, D. Beaini, W. Hamilton, V. Létourneau, and P. Tossou. Rethinking graph transformers with spectral attention. *Advances in Neural Information Processing Systems*, 34:21618–21629, 2021.

[51] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. Technical report, 2009.

[52] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79 – 86, 1951.

[53] K. Kumar and V. Prasad. Few generalized entropic relations related to rydberg atoms. *Scientific Reports*, 12(1):7496, May 2022.

[54] R. Levie, F. Monti, X. Bresson, and M. M. Bronstein. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Transactions on Signal Processing*, 67(1):97–109, 2018.

[55] G. Li, M. Müller, A. Thabet, and B. Ghanem. Deepgcns: Can gcns go as deep as cnns? In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9266–9275, 2019.

[56] Q. Li, Z. Han, and X.-M. Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[57] R. Li, S. Wang, F. Zhu, and J. Huang. Adaptive graph convolutional neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[58] Y. Li, R. Yu, C. Shahabi, and Y. Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

[59] J. T. Lizier. Jidt: An information-theoretic toolkit for studying the dynamics of complex systems. *Frontiers in Robotics and AI*, 1, Dec. 2014.

[60] J. T. Lizier, J. Heinzle, A. Horstmann, J.-D. Haynes, and M. Prokopenko. Multivariate information-theoretic measures reveal directed information structure and task relevant changes in fmri connectivity. *Journal of Computational Neuroscience*, 30(1):85–107, Feb 2011.

[61] J. T. Lizier and M. Prokopenko. Differentiating information transfer and causal effect. *The European Physical Journal B*, 73:605–615, 2010.

[62] S. Luan, C. Hua, Q. Lu, J. Zhu, M. Zhao, S. Zhang, X.-W. Chang, and D. Precup. Is heterophily a real nightmare for graph neural networks to do node classification?, 2021.

[63] S. Luan, C. Hua, Q. Lu, J. Zhu, M. Zhao, S. Zhang, X.-W. Chang, and D. Precup. Revisiting heterophily for graph neural networks. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 1362–1375. Curran Associates, Inc., 2022.

[64] Y. Ma, X. Liu, N. Shah, and J. Tang. Is homophily a necessity for graph neural networks? *arXiv preprint arXiv:2106.06134*, 2021.

[65] A. Moldovan, A. Caţaron, and R. Andonie. Learning in feedforward neural networks accelerated by transfer entropy. *Entropy*, 22(1):102, 2020.

[66] A. Moldovan, A. Caţaron, and R. Andonie. Learning in convolutional neural networks accelerated by transfer entropy. *Entropy*, 23(9), 2021.

[67] A. Moldovan, A. Caţaron, and R. Andonie. Information plane analysis visualization in deep learning via transfer entropy. In *2023 27th International Conference Information Visualisation (IV)*, pages 278–285, 2023.

[68] A. Moldovan, A. Caţaron, and R. Andonie. Transfer entropy in graph convolutional neural networks. In *2024 28th International Conference Information Visualisation (IV)*, pages 278–285, 2024.

[69] F. Monti, M. Bronstein, and X. Bresson. Geometric matrix completion with recurrent multi-graph neural networks. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[70] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Ng. Reading digits in natural images with unsupervised feature learning. *NIPS*, 01 2011.

[71] O. Obst, J. Boedecker, and M. Asada. Improving recurrent neural network performance using transfer entropy. In *Proceedings of the 17th International Conference on Neural Information Processing: Models and Applications - Volume Part II*, ICONIP'10, pages 193–200, Berlin, Heidelberg, 2010. Springer-Verlag.

[72] K. Oono and T. Suzuki. Graph neural networks exponentially lose expressive power for node classification. *ICLR2020*, 8, 2020.

[73] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford infolab, 1999.

[74] G. Panagopoulos and F. D. Malliaros. Influence learning and maximization. In *International Conference on Web Engineering*, pages 547–550. Springer, 2021.

[75] O. Platonov, D. Kuznedelev, M. Diskin, A. Babenko, and L. Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: are we really making progress?, 2023.

[76] Y. Rong, W. Huang, T. Xu, and J. Huang. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*, 2019.

[77] T. K. Rusch, M. M. Bronstein, and S. Mishra. A survey on oversmoothing in graph neural networks, 2023.

[78] K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, and A. Cabellos-Aparicio. Routenet: Leveraging graph neural networks for network modeling and optimization in sdn. *IEEE Journal on Selected Areas in Communications*, 38(10):2260–2270, 2020.

[79] A. M. Saxe, Y. Bansal, J. Dapello, M. Advani, A. Kolchinsky, B. D. Tracey, and D. D. Cox. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, dec 2019.

[80] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[81] T. Schreiber. Measuring information transfer. *Phys. Rev. Lett.*, 85:461–464, Jul 2000.

[82] W. Shadish, T. Cook, and D. Campbell. *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. Boston: Houghton Mifflin, 2001.

[83] O. Shamir, S. Sabato, and N. Tishby. Learning and generalization with the information bottleneck. *Theoretical Computer Science*, 411(29-30):2696–2711, 2010.

[84] C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[85] C. E. Shannon. Coding theorems for a discrete source with a fidelity criterion (international convention record), vol. 7. *New York, NY, USA: Institute of Radio Engineers*, 1959.

[86] M. Shimono and J. M. Beggs. Functional clusters, hubs, and communities in the cortical microconnectome. *Cerebral cortex (New York, N.Y. : 1991)*, 25(10):3743–3757, Oct 2015. 25336598[pmid].

[87] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

[88] R. Shwartz Ziv and Y. LeCun. To compress or not to compress—self-supervised learning and information theory: A review. *Entropy*, 26(3), 2024.

[89] R. Shwartz-Ziv and N. Tishby. Opening the black box of deep neural networks via information. *CoRR*, abs/1703.00810, 2017.

[90] N. Slonim and N. Tishby. Agglomerative information bottleneck. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

[91] L. Song, Y. Zhang, Z. Wang, and D. Gildea. A graph-to-sequence model for amr-to-text generation. *arXiv preprint arXiv:1805.02473*, 2018.

[92] A. Sperduti and A. Starita. Supervised neural networks for the classification of structures. *IEEE transactions on neural networks*, 8(3):714–735, 1997.

[93] L. Spillmann, B. Dresp-Langley, and C.-H. Tseng. Beyond the classical receptive field: the effect of contextual stimuli. *Journal of Vision*, 15(9):7–7, 2015.

[94] N. Tishby, F. C. Pereira, and W. Bialek. The information bottleneck method, 1999.

[95] N. Tishby and N. Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5, 2015.

[96] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, et al. Graph attention networks. *stat*, 1050(20):10–48550, 2017.

[97] R. Vicente, M. Wibral, M. Lindner, and G. Pipa. Transfer entropy—a model-free measure of effective connectivity for the neurosciences. *Journal of Computational Neuroscience*, 30(1):45–67, Feb 2011.

[98] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt. Graph kernels. *The Journal of Machine Learning Research*, 11:1201–1242, 2010.

[99] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo. Graphgan: Graph representation learning with generative adversarial nets. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[100] J. Wang, Y. Guo, L. Yang, and Y. Wang. Understanding heterophily for graph neural networks. *arXiv preprint arXiv:2401.09125*, 2024.

[101] T. Wang, D. Jin, R. Wang, D. He, and Y. Huang. Powerful graph convolutional networks with adaptive propagation mechanism for homophily and heterophily. In *AAAI Conference on Artificial Intelligence*, volume 36, pages 4210–4218, Jun. 2022.

[102] J. Weston, F. Ratle, and R. Collobert. Deep learning via semi-supervised embedding. In *Proceedings of the 25th international conference on Machine learning*, pages 1168–1175, 2008.

[103] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[104] C. Xie, J. Zhou, S. Gong, J. Wan, J. Qian, S. Yu, Q. Xuan, and X. Yang. Pathmlp: Smooth path towards high-order homophily, 2023.

[105] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5410–5419, 2017.

[106] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

[107] Y. Yan, Y. Chen, H. Chen, M. Xu, M. Das, H. Yang, and H. Tong. From trainable negative depth to edge heterophily in graphs. *Advances in Neural Information Processing Systems*, 36, 2024.

[108] Y. Yan, M. Hashemi, K. Swersky, Y. Yang, and D. Koutra. Two sides of the same coin: Heterophily and oversmoothing in graph convolutional neural networks. In *IEEE International Conference on Data Mining (ICDM)*, pages 1287–1292, Los Alamitos, CA, USA, dec 2022. IEEE Computer Society.

[109] H. Yang, K. Ma, and J. Cheng. Rethinking graph regularization for graph neural networks. In *Proceedings of the aaai conference on artificial intelligence*, volume 35, pages 4573–4581, 2021.

[110] M. Yoon. Introduction to graph neural networks, 2022.

[111] J. You, R. Ying, X. Ren, W. L. Hamilton, and J. Leskovec. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International Conference on Machine Learning*, 2018.

[112] W. Yu, C. Zheng, W. Cheng, C. C. Aggarwal, D. Song, B. Zong, H. Chen, and W. Wang. Learning deep network representations with adversarially regularized autoencoders. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2663–2671, 2018.

[113] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim. Graph transformer networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[114] C. Zhang, Q. Li, and D. Song. Aspect-based sentiment classification with aspect-specific graph convolutional networks. *arXiv preprint arXiv:1909.03477*, 2019.

[115] M. Zhang and Y. Chen. Link prediction based on graph neural networks. *Advances in neural information processing systems*, 31, 2018.

[116] K. Zhao, Q. Kang, Y. Song, R. She, S. Wang, and W. P. Tay. Graph neural convection-diffusion with heterophily, 2023.

[117] L. Zhao and L. Akoglu. Pairnorm: Tackling oversmoothing in gnns. *arXiv preprint arXiv:1909.12223*, 2019.

[118] E. Zheleva and L. Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th*

International Conference on World Wide Web, WWW '09, page 531–540, New York, NY, USA, 2009. Association for Computing Machinery.

[119] T. Zhou, L. Lü, and Y.-C. Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71:623–630, 2009.

[120] J. Zhu, Y. Yan, L. Zhao, M. Heimann, L. Akoglu, and D. Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7793–7804. Curran Associates, Inc., 2020.

[121] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003.