A Bird's-Eye View of Attention Architectures for Graph Mining

Kaustubh D. Dhole
kdhole@emory.edu
Emory University
Atlanta, Georgia, USA

ABSTRACT

Graph Neural Networks (GNNs) have shown tremendous strides in performance for graph-structured problems especially in the domains of natural language processing, computer vision and recommender systems. Inspired by the success of the transformer architecture, there has been an ever-growing body of work on attention variants of GNNs attempting to advance the state of the art in many of these problems. Incorporating "attention" into graph mining has been viewed as a way to overcome the noisiness, heterogenity and complexity associated with graph-structured data as well as to encode soft-inductive bias. It is hence crucial and advantageous to study these variants from a bird's-eye view to assess their strengths and weaknesses. We provide a systematic and focused survey centered around attention based GNNs in a hope to benefit researchers dealing with graph-structured problems. Our survey looks at the GNN variants from the point of view of the attention function and iteratively builds the reader's understanding of different attention variants.

KEYWORDS

transformers, graph mining, graph neural networks, attention

ACM Reference Format:
Kaustubh D. Dhole. 2018. A Bird's-Eye View of Attention Architectures for Graph Mining. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The success of deep learning, with the advent of computational power and large swathes of data has bolstered the performance of a variety of tasks, especially in the fields of natural language processing, speech recognition and computer vision. Simultaneously, there has been a proportional rise in graph-structured data in the form of knowledge-graphs, point clouds, protein and molecular data, recommender systems, etc. Unsurprisingly, there has been an increasing recent interest in extending many of the successful deep learning architectures to address the complexities associated with such ubiquitous graphical data.

One set of such architectures referred to as graph neural networks (GNNs) have been the defacto models to cater to a plethora of problems representable in terms of nodes and edges. In Bioinformatics, GNNs have greatly benefit protein interaction prediction [18] by incorporating graphical structure in addition to sequential information of proteins [58]. GNNs have been successfully employed in the computer vision domain converting to and fro between scene graphs and images [68], for processing point clouds [53], etc. In recommender systems, graph-based systems are popular for modelling interactions among users, products, etc. [50] In NLP, GNNs have been explored to interpret tree and graph representations of syntactic and semantic parses like dependency tree structures and abstract meaning representations [51] as well as for modelling knowledge graphs [15, 59]. Other adoptions of GNNs have spanned multifarious domains like music generation [70], mass spectrometry [62], bio-inspired camera denoising [1],molecular property prediction [41], etc. A majority of these GNNs have extended vanilla architectures of recurrent neural networks (RNNs), convolutional neural networks (CNNs), autoencoders and transformer models.

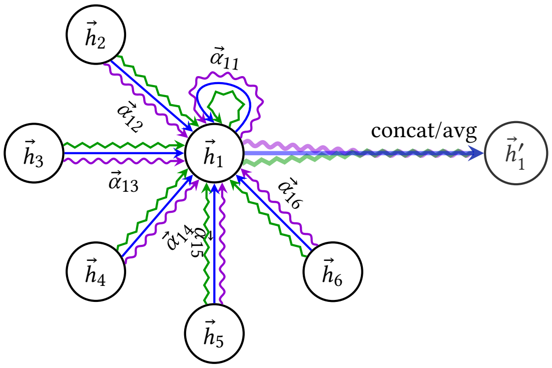

Figure 1: [48]'s illustration of multi-head attention by node 1 on its neighborhood. 3 colors depict $K = 3$ heads computed individually. $\vec{h}'_1$ is obtained by concatenating or averaging features from each head.

With mammoth availability of data over the internet, real-world graphs are generally complicated, are highly heterogenous in nature and worst of all, tend to be noisy and incomplete. Transformer [4] variants of GNNs have been an effective way to deal with such noise by learning to "attend" or "focus" on the relevant nodes or subgraphs while providing an empirical boost on graph tasks by encoding soft-inductive bias.

While the literature of GNNs and graph mining has been reviewed and surveyed a few times in extreme detail, most of the surveys have not significantly covered them in the context of transformers. [55] performed a comprehensive survey by proposing a 4-models taxonomy of GNNs, investigating recurrent GNNs, convolutional GNNs, graph autoencoders, and spatial-temporal GNNs,

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06...$15.00
https://doi.org/XXXXXXX.XXXXXXX

but do not touch on attention/transformer variants. [50] discuss graph-based representations to better recommender systems. [44] discuss some serious shortcomings of GNN evaluation. [69] provide a general design pipeline for GNNs and talk about architectures from a classical point of view. [11] demonstrate a systematic categorization of problems, techniques and applications of graph embeddings for more than 150 papers until 2018. The last related survey focusing on attention networks for graphs was conducted over 3 years before by [34]. A plethora of attention variants have been experimented ever since viz. the GraphFormers [59], GATv2 [8], graph-BERT [35, 63–65], LiteGT [13], Graph Kernel Attention [17], Spectral Attention Network(SAN) [32] etc. It is hence crucial to survey these recent approaches to extract insights on model performance and to gauge where the field is heading.

As discussed earlier, with the rapid popularity and ground-breaking success of attention based models, many attention variants of GNNs have been experimented with by aggregating attention over other nodes of the graph [56] to try to further improve performance on many tasks [5, 8, 17, 21, 27, 35, 59, 63–65]. This focused survey aims to discuss the different architectures of each of these later variants and their performance characteristics on downstream tasks.

Some architectures may perform well on certain graphs, while some many not. There is no universal architecture suitable for every problem, for the selection of the architecture is highly dependent on the traits of the graphs. Our aim is to equip readers with a thorough understanding of these architectures to help contextualise them in their particular problem. While this review is primarily targeted towards graph practitioners, we our optimistic that newbies who are curious about graph neural networks will also be able to make the best use of the same.

2 TAXONOMY OF GRAPH PROBLEMS

We taxonomize attention variants of GNNs according to four intended downstream applications viz node level problems, edge level problems, graph level problems and others which do not fall under the former three. The tasks in the bracket describe the datasets/tasks over which each of the architectures has been evaluated by the original authors or benchmarked in subsequent papers.

2.1 Node Level

Node level tasks majorly include supervised tasks like node classification and node regression as well as unsupervised tasks like node clustering. Knowledge graph problems involving prediction of labels of new nodes would be a popular application. Model performance can be gauged using a plethora of benchmark datasets namely Cora, Citeseer, Pubmed, etc. In this survey, the following node classification architectures would be discussed.

- GAT (Cora, Citeseer, PubMed, PPI)
- GATv2 (Cora, Citeseer, PubMed, PPI)
- GaAN (PPI [23], Reddit [23], METR-LA [36])
- EdgeGAT (Cora, Citeseer, Pubmed, 2 edge sensitive datasets Trade-B and Trade-M)
- HyperbolicGAT (Cora, Citeseer, Pubmed, Amazon Photo)
- HANs (ACM, IMDB, DBLP all requiring metapath information)
- SAN (CLUSTER, PATTERN)
- GraphBERT (Cora, Citeseer, PubMed)
- CAT (Cora, Cite, Pubmed, CoAuthorCS [44], OGB-Arxiv[26])

2.2 Edge Level

Edge level tasks expect edges to be classified, viz link classification [22]. Knowledge graph problems involve tasks to predict missing relations or missing links between existing entities.

- GAT (OGB)
- GATv2 (OGB)
- HyperbolicGAT (realtional reasoning CLEVR, Sort-of-CLEVR)
- GAATs [49]
- SAttLE [3] (FB15k-237, WN18RR)
- HittER[16] (FB15k-237, WN18RR, FreebaseQA, WebQuestionsQA)

2.3 Graph Level

These involve classification and regression tasks at the graph level eg. classifying if the graph of a certain molecule shows properties of inhibiting HIV or not.

- SAN (ZINC regression, MolHIV, MolPCBA classification)

2.4 Others

These are two tasks that GraphBERT uses for pretraining: the node raw attribute reconstruction task focussing on extracting node attribute information and graph structure recovery task focuses on graph connection information.

- SAN (ZINC regression, MolHIV, MolPCBA classification)

3 LIMITATIONS OF MESSAGE PASSING PARADIGMS

The dominant techniques in GNNs incorporate a sparse message-passing process to directly capture graph structure [24] wherein messages are iteratively passed between nodes in the graph. [24] provide a great review of different paradigms. However, this message-passing paradigm has been plagued with several limitations. eg. The expressiveness of message passing seems inescapably limited by the Weisfeiler-Lehman isomorphism hierarchy [32, 38–40]. Besides, message passing paradigms have been victims of *oversquashing* and *oversmoothing*:

Due to the exponential blow-up of computational routes, it is hard for graph neural networks to relay information to distant neighbors. This hardness is referred to as *oversquashing*.

With the addition of more number of layers, GNNs have not shown performance gains. This limitation is referred to as *oversmoothing* [10].

Besides, the message passing paradigm limits the structure of the model's computation graph necessitating the need for approaches which provide the flexibility of soft-inductive bias. The Transformer architecture, for example, eliminates any structural inductive bias by encoding the structure with soft inductive biases like positional encodings [32].

4 ATTENTION ARCHITECTURES IN GRAPH NEURAL NETWORKS

We first formulate and characterize the attention equations popular in sequential problems by [4, 47] and then describe graph variants of the same.

In the transformers architecture, Vaswani et al, [47] define the scaled dot-product attention as follows for query, key and value matrices Q, K and V.

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Sequence-to-sequence models computed context representations c_i for the ith decoder step by attending over all T_x encoder steps indexed by j.

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j$$

where α_{ij} represented the learned attention weights

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x}\exp(e_{ik})}$$

$$e_{ij} = a(s_{i-1}, h_j)$$

A common choice for a has been Bahdanau's attention [4]

$$e_{ij} = v^T tanh(W[s_{i-1}; h_j])$$

The above equations brought about unprecedented success for NLP tasks like machine translation, speech recognition, question answering, etc. and no wonder, the GNN community was motivated to incorporate the same to compute node representations. This was accomplished by "attending" over other nodes in the graph as we will see in subsequent sections.

4.1 Graph Attention Networks (GAT)

Veličković et al [48]'s seminal work established Graph Attention Networks, computing node representations by attending over neighbouring nodes $\mathcal{N}_i$ or nodes one-hop away. For every node i, each neighbouring node j is weighted by a factor α_{ij} computed as:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k\in\mathcal{N}_i}\exp(e_{ik})}.$$

where e_{ij} for nodes i and j can be expressed further in terms of their node features h_i and h_j

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)} \tag{1}$$

where $\|$ represents concatenation as described in their paper [48].Figure 1 describes the neighbourhood attention computation as described in [48].

The final representation of the node was then computed by taking a linear weighted sum of the the neighbours as follows:

$$\vec{h}'_i = \sigma\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}\mathbf{W}\vec{h}_j\right). \tag{2}$$

where σ represents a non-linear activation function.

Taking inspiration from [47], multi-head attention is similarly computed. This multi-head attention is found to stabilize learning [34]. Mathematically, Equation 2 is executed K times (the no. of attention heads), and concatenated (denoted by $\|$), resulting in the following output feature representation:

$$\vec{h}'_i = \Big\|_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right) \tag{3}$$

k is used to denote the k-th attention head.

4.2 Gated Attention Networks (GaAN)

Zhang et al[67] hypothesize that some of the attention heads in 3 might be redundant and could mislead final predictions. They strive to rectify this by introducing small convolutional subnetworks and compute a soft gate (0:low importance to 1:high importance) at each attention head to control the importance of that head:

$$\vec{h}'_i = \mathbf{W}(\vec{h}_i \oplus \Big\|_{k=1}^{K} \sigma\left(g_i^{(k)}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right)) \tag{4}$$

where $g_i^{(k)}$ is the value of the kth head at node i.

$$g_i = [g_i^{(1)},...,g_i^{(K)}] = \psi_g(\mathbf{x}_i, \mathbf{z}_{\mathcal{N}_i}) \tag{5}$$

ψ_g is a convolutional network that takes in the centre node and the neighbouring nodes as input and to compute gate values. The GaAN approach additionally adopted the key-value attention and the dot product attention as compared to GAT which does not compute separate value vectors.

4.3 Edge GATs (EGATs)

While the architectures discussed above only use node features in attention computation, some tasks might benefit with the incorporation of edge information. Eg. popular knowledge graphs like FreeBase and ConceptNet have lots of relations between entities. Tasks like classifying entities (node classification) and predicting links between them would greatly benefit via cues of edge information. Trading networks too generally model send/receive payments as edges between nodes of users. Naturally, modelling relations has been an important facet of GNN research. Relational Graph Convolutional Networks (R-GCNs) [43] introduced modelling relational data in GCNs for two knowledge base completion tasks: link prediction & entity classification. Chen & Chen [14] argue that different graphs may have different preferences for edges and weights and hence introduce Edge GATs. They extend the use of GATs to incorporate edge features in addition to the original node features.

Node representation described in 1 now additionally incorporates edge embeddings e_{ik}.[1]

$$\alpha_{ij} = \frac{\exp(\text{LeakyReLU}(\vec{\mathbf{a}}^T[\vec{h}_i \| \vec{h}_j \| \vec{e}_{ij}]))}{\sum_{k\in\mathcal{N}_i}\exp(\text{LeakyReLU}(\vec{\mathbf{a}}^T[\vec{h}_i \| \vec{h}_k \| \vec{e}_{ik}]))} \tag{6}$$

Node updates are performed similar to GAT's Equation 2. In a multilayer attention architecture wherein multiple node encoders would be stacked, the last encoder attends over $\vec{h}_j \| \vec{e}_{ij}$ instead of $\vec{h}_j$.

[1] W has been omitted to reduce space.

Edge embeddings are then updated in parallel by attending over neighbouring edges. This is achieved by reversing the roles of nodes and edges and defining neighbouring edges as those connected with a common vertex. Effectively, node information is used to update edge representation. Arguably, such an update seems counter intuitive when graphs have independent pre-defined relations. It can be argued that certain edges might be highly prevalent in conjunction with specific subsets of nodes making node information vital for edge representation e.g. in graphs with a large number of relation types.

4.4 Heterogeneous Attention Networks (HANs)

Many real world graphs contain multiple types of nodes and edges as well as crucial information residing in the form of "meta-paths". Widely used in data mining, such heterogeneous graphs, adapted from heterogeneous information networks (HIN) contain more comprehensive information and richer semantics. [45, 52]. Eg. one key feature of HINs is the ability to spread information through various edges among different-typed nodes [45] i.e depending on the meta-paths, the relation between nodes in a heterogeneous graph can have varying semantics [52]. Figure 2 describes how the relation between two movies can be described by two metapaths: Movie-Actor-Movie or Movie-Director-Movie.

[52] introduce hierarchical attention which includes 1) node level attention attending over the meta-path based neighbours and 2) semantic-level attention to learn the importance of different metapaths.

Node level attention for a given neighbouring metapath Φ and for a node i is computed by looping over all the nodes N_i^{Φ} appearing in the metapath.

$$\mathbf{z}_i^{\Phi} = \overset{K}{\underset{k=1}{\Vert}} \sigma\left(\sum_{j \in \mathcal{N}_i^{\Phi}} \alpha_{ij}^{\Phi} \cdot \mathbf{h}'_j \right). \tag{7}$$

Since each metapath from $\{\Phi_1, \ldots, \Phi_P\}$ presents its own semantics, the node representation is computed for each metapath providing P groups of node representations denoted as $\left\{\mathbf{Z}_{\Phi_1}, \ldots, \mathbf{Z}_{\Phi_P}\right\}$. Semantic-level attention [52] refers to attending over the node representations of these metapaths.

$$\mathbf{Z} = \sum_{p=1}^{P} \beta_{\Phi_p} \cdot \mathbf{Z}_{\Phi_p}. \tag{8}$$

where the semantic attention weights β_{Φ_p} are computed as follows

$$w_{\Phi_p} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \mathbf{q}^{\mathrm{T}} \cdot \tanh(\mathbf{W} \cdot \mathbf{z}_i^{\Phi_p} + \mathbf{b})$$

$$\beta_{\Phi_p} = \frac{\exp(w_{\Phi_p})}{\sum_{p=1}^{P} \exp(w_{\Phi_p})},$$

4.5 Hyperbolic GATs

The above networks that we looked at were designed primarily for the Euclidean space. However, some work has pointed out that Euclidean spaces may not provide the perfect geometric environment for learning graph representations as graphs exhibit many non-Euclidean traits[9, 33, 37, 54]. Some graphical properties like hierarchical & power-law structure are naturally reflected in hyperbolic spaces [33, 37] and hence a lot of subsequent work focused on graph neural networks in the hyperbolic space. [tocite]

Since basic algebraic operations like addition & multiplication are not straightforward in the hyperbolic space, [**?**] introduce a hyperbolic proximity based attention mechanism, Hyperbolic Attention Network (HAN) by utilizing gyrovector spaces to featurise the graph. Gyrovector spaces, introduced by [46] provide an elegant formalism for algebraic operations in hyperbolic geometry. The authors notice that HAT performs better in most problems, especially in low dimensional settings.

While there are plenty of GNN architectures proposed in the hyperbolic space, it would be infeasible to discuss and introduce a separate parallel set of notation of hyperbolic geometry within the scope of this review. We hence encourage the reader to read up the article by [9].

4.6 Graph Transformers

Along with attention, positional encodings have been extensively studied for sequential problems where positional information of words is crucial eg. tasks in NLP. Analogously, for fundamental graph tasks, recent studies [20, 20, 60, 61] point out positional information to be key to improve and overcome many of GNNs' failures . [19] replace the sinusoidal embeddings used for sentences or line graphs with the generalised Laplacian positional encodings usable for arbitrary graphs. They also substitute layer normalisation with batch normalisation. Additionally, the architecture attends over neighbouring nodes and takes edge representation too into account.

4.7 Graph BERT

Li et al. attempt to incorporate global information of the graph by extending the attention computation over all the nodes i.e. not just neighbouring nodes. However, such an approach can be costly and does not take advantage of graph sparsity. Besides, GNNs have been known to suffer from problems of "suspended animation" & "oversmoothing". To rectify such isses, [GraphBERT] performs sampling of linkless subgraphs i.e. sample nodes together with their context. While incorporating global information of the graph, it can be crucial to understand the position of nodes within the broader context of the graph. [61] argue that two nodes residing in very different parts of the graph may have an isomorphic topology of neighbourhood but may deserve different representations. They incorporate every node's positional information by computing the distance from a set of random nodes in each forward pass.

Importantly, domain specific & low-resource graphs might not be large enough to train such parameter-heavy graph neural networks and hence learning strategies, successful in machine translation like pre-training + fine-tuning [BERT] can prove beneficial. GraphBERT has been introduced as a graph representation learning on the same lines. The authors pretrained GraphBERT for two tasks, namely node attribute reconstruction and graph structure recovery & fine-tuned it for the task of node classification.

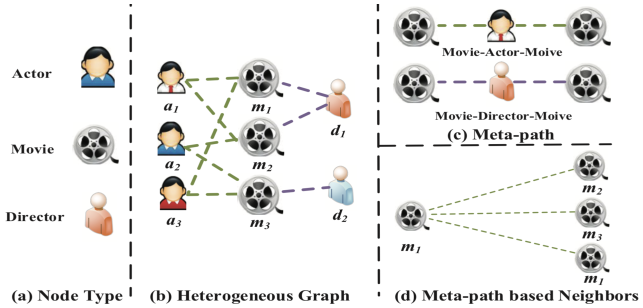

Figure 2: An example of a meta-path as described in Wang et al [52]

A D-layer GraphBERT architecture is summarised below:

$$\begin{cases} \mathbf{H}^{(0)} = [\mathbf{h}_i^{(0)}, \mathbf{h}_{i,1}^{(0)}, \cdots, \mathbf{h}_{i,k}^{(0)}]^\top, \\ \mathbf{H}^{(l)} = \text{G-Transformer}\left(\mathbf{H}^{(l-1)}\right), \forall l \in \{1, 2, \cdots, D\}, \\ \mathbf{z}_i \quad = \text{Fusion}\left(\mathbf{H}^{(D)}\right). \end{cases}$$

where each $\mathbf{h}_i^{(0)}$ is a concatenation of 4 different types of embeddings for a node: 1) the raw feature vector embeddings, (2) Weisfeiler-Lehman absolute role embeddings, (3) intimacy based relative positional embeddings, & (4) hop based relative distance embeddings. And $G - Transformer$ is the usual softmax attention alongwith a residual term G-Res as defined in graph residul networks [66].

$$\begin{aligned} \mathbf{H}^{(l)} &= \text{G-Transformer}\left(\mathbf{H}^{(l-1)}\right) \\ &= \text{softmax}\left(\frac{\mathbf{H^{(l-1)}W_Q^{(l)}H^{(l-1)}W_K^{(l)}}^\top}{\sqrt{d_h}}\right)\mathbf{H^{(l-1)}W_V^{(l)}} \\ &+ \text{G-Res}\left(\mathbf{H}^{(l-1)}, \mathbf{X}_i\right), \end{aligned}$$

Laplacian positional encodings outperform the WL-positional encodings introduced above in capturing positional and structural information as well as in generalization.

4.8 Spectral Attention Network (SAN)

Taking inspiration from spectral graph theory, SAN [32] try to address some of the theoretical limitations of the above Graph Transformer work. They make use of the complete Laplace spectrum for positional encodings. Apart from the benefits offered by [19], SAN incorporate variable number of eigenvectors, the whole spectrum of eigenvalues and are aware of eigenvalue multiplicities. Given appropriate parameters and the utilization of the whole set of eigenfunctions, SAN can approximately differentiate any pair of non-isomorphic graphs, making it more powerful than any WL test. SAN also might be less prone to oversquashing.

The authors argue that unlike the graph transformers discussed earlier which don't exploit eigenvalues and eigenfunctions wholly, SAN is able to model physical interactions better i.e. interactions commonly observed in physics, biology and images. SAN is seen to outperform other attention-based architectures by a large margin on the tasks of ZINC [29], a molecular graph regression dataset, PATTERN & CLUSTER [20], two synthetic benchmarks for node classification & MolPCBA [26], a dataset for molecular graph classification.

4.9 Gated Attention Network v2

[8] argued that the previous well-known GAT design proposed by Veličković et al,[48] and its variants which spread out across different graph domains computed a limited form of attention which was static in nature rather than the actual expressive attention function of Bahadanu et al [4]. [8] showed that the attention function is monotonic with respect to the neighbouring nodes. This monotonicty is shared across all nodes in the graph without being conditioned on the query node. GATv2 instead computes dynmamic attention, with a simple fix by switching the order of internal operations in GAT.

$$\begin{aligned} \text{GAT [48]:} \qquad & e\left(\vec{h}_i, \vec{h}_j\right) = \text{LeakyReLU}\left(\vec{\mathbf{a}}^T [\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right) \\ \text{GATv2[8]:} \qquad & e\left(\vec{h}_i, \vec{h}_j\right) = \vec{\mathbf{a}}^T \text{LeakyReLU}\left([\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right) \end{aligned}$$

The standard GAT scoring function[48] applies the learned layers $\mathbf{W}$ and $\vec{\mathbf{a}}^T$ consecutively, and effectively collapses them into one linear layer. By simply applying the $\vec{\mathbf{a}}^T$ layer after the non-linearity (LeakyReLU) and the $\mathbf{W}$ layer after the concatenation, GATv2 overcomes this issue rendering it with a universal approximator function and making it strictly more expressive than GAT. Besides theoretical superiority, GATv2 has shown empirical advantages over GAT on various tasks which require dynamic selection of nodes.

4.10 Graph Conjoint Attention networks (CATs)

Contextual interventions have been found as helpful external elements that may increase attention and cognitive capacities in cognitive science [30]. Inspired by this finding, [25] describe the concept of conjoint attentions. They incorporate node cluster embedding, and higher-order structural correlations, arguing that such external components can enhance learning and provide more robustness to graph neural networks, e.g. against overfitting.

Here is the formal definition of a structural intervention: $\Psi(\cdot)$ can be any distance function (eg. Euclidean) and $\phi(\cdot)$ can be any operator transforming $\mathbf{C}$ to the same dimensionality of $\mathbf{Y}$, the prior feature matrix. The structural intervention between two nodes can be defined as below:

$$\mathbf{C}_{ij} = \underset{\phi(\mathbf{C})_{ij}}{\arg\min}\, \Psi(\phi(\mathbf{C})_{ij}, \mathbf{Y}_{ij}),$$

$$s_{ij} = \frac{\exp\left(\mathbf{C}_{ij}\right)}{\sum_{k\in\mathcal{N}_i} \exp\left(\mathbf{C}_{ik}\right)}.$$

Given the f_{ij} (Equation 1) and s_{ij}, [25] propose two different strategies to compute the Conjoint Attention scores, aiming at allowing CATs to depend on the structural intervention at different levels. The first mechanism is referred here as *Implicit direction.* Each CA layer introduces two learnable parameters, g_f and g_s. They can be obtained in the following fashion and are used to determine the relative importance of feature and structural correlations:

$$r_f = \frac{\exp\left(g_f\right)}{\exp\left(g_s\right) + \exp\left(g_f\right)}, r_s = \frac{\exp\left(g_s\right)}{\exp\left(g_s\right) + \exp\left(g_f\right)},$$

CAT subsequently computes the attention score based as follows:

$$\alpha_{ij} = \frac{r_f \cdot f_{ij} + r_s \cdot s_{ij}}{\sum_{k\in\mathcal{N}_i} [r_f \cdot f_{ik} + r_s \cdot s_{ik}]} = r_f \cdot f_{ij} + r_s \cdot s_{ij}.$$

The explicit strategy performs a different computation as follows:

$$\alpha_{ij} = \frac{f_{ij} \cdot s_{ij}}{\sum_{k\in\mathcal{N}_i} f_{ik} \cdot s_{ik}}.$$

Eventually, the higher layers are updated as follows, with a learnable parameter $\epsilon \in (0, 1)$ added to introduce expressivity of the CAT approach:

$$\mathbf{h}_i^{l+1} = (\alpha_{ii} + \epsilon \cdot \frac{1}{|\mathcal{N}_i|})\mathbf{W}^l\mathbf{h}_i^l + \sum_{j\in\mathcal{N}_i, j\neq i} \alpha_{ij}\mathbf{W}^l\mathbf{h}_j^l, \qquad (9)$$

The authors showed CAT to be better than GAT on many node classification and clustering tasks but the approach seems to increase both space and time complexity.

4.11 Additional Attention Based GNNs

Additional variants majorly build on top of these fundamental equations. Eg, Graphormers [60] encode node centrality and spatial relations encoded in node pairs and edges alongwith the softmax attention. To improve efficiency, Coarformer [2] performs attention on a courser version of the original graph. [28] give an account of sublinear graph coarsening strategies to reduce the number of nodes by up to a factor of ten without causing a noticeable performance degradation. LiteGT [13] describes $O(nlogn)$ time node sampling strategies resulting in a 100x time reduction and reduced model size to enjoy similar performance.

5 SUMMARY

Now that we've witnessed the mathematical details of each architecture from the point of view of the attention function and downstream application, we can revisit these architectures to address the task at hand. For node classification problems where node positions might not play a big role, it would be worthwhile experimenting with Graph Attention Networks, GAT and GATv2 by attending over neighbouring nodes. While many graphs are sparse in nature while also require making global inferences, it might be necessary to look at the whole graph but might be too costly to do so. In such a case, it is imperative to investigate some of the sampling techniques used to prune the attention candidates as well as encode positional information. eg. as used in GraphBERT alongwith retrieving positional encodings. Having a more exhaustive set of positional encodings and performing better than GraphTransformer, SAN could also be a vital choice in such problems.

For tasks like link prediction or other graph tasks reliant on graphs with vital edge information, it is worthwhile incorporating edge information like EdgeGATs. Besides, one has to be cautious with these higher attention variants as each of these architectures when employed with multiple attention heads would charge a heavy computational cost. Eg, EdgeGATs and CATs, despite showing impressive performance might not be a good choice when computational budgets are constrained. Graphormers would be a great choice for tasks with smaller knowledge graphs since the complexity grows quadratically.

As mentioned earlier, architectures may perform well depending on the graph at hand. There is hardly any universal architecture suitable for every problem. We hope this survey makes readers better informed towards their design choices of GNN architectures.

6 OTHER AVENUES IN GRAPH ML

Despite attention variants being the focus of the current study, there are other sub-fields of graph machine learning eyeing different areas of GNNs which deserve equal attention. Other innovative directions in the field of graph machine learning can be attributed to the works of GFlowNets [6, 7], the study of how GNNs are aligned with dynamic programming [57], GNNs with combinatorial optimisation [12], Satorras, et al [42]'s Equivariant GNNs, Klicpera et al [31]'s GEMnet, etc.

7 CONCLUSION

This survey attempts to delve into multiple attention architectures detailing the strengths and weaknesses of each. We provide the mathematical crux of each of the attention equations in a uniform notation for the benefit of readers. We hope this survey provides guidance to researchers (dealing with graph-structured problems described earlier) so that they can get a high-level overview for their tasks.

REFERENCES

[1] Yusra Alkendi, Rana Azzam, Abdulla Ayyad, Sajid Javed, Lakmal Seneviratne, and Yahya Zweiri. 2021. Neuromorphic Camera Denoising using Graph Neural Network-driven Transformers. arXiv:2112.09685 [cs.CV]

[2] Anonymous. 2022. Coarformer: Transformer for large graph via graph coarsening. In *Submitted to The Tenth International Conference on Learning Representations.* https://openreview.net/forum?id=fkjO_FKVzw under review.

[3] Peyman Baghershahi, Reshad Hosseini, and Hadi Moradi. 2021. Self-attention Presents Low-dimensional Knowledge Graph Embeddings for Link Prediction. *arXiv preprint arXiv:2112.10644* (2021).

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[5] Tian bao Yang, Yujing Wang, Zhi Yue, Yaming Yang, Yunhai Tong, and Jing Bai. 2021. Graph Pointer Neural Networks. *ArXiv* abs/2110.00973 (2021).

[6] Emmanuel Bengio, Moksh Jain, Maksym Korablyov, Doina Precup, and Yoshua Bengio. 2021. Flow Network based Generative Models for Non-Iterative Diverse Candidate Generation. *Advances in Neural Information Processing Systems* 34 (2021).

[7] Yoshua Bengio, Tristan Deleu, Edward J Hu, Salem Lahlou, Mo Tiwari, and Emmanuel Bengio. 2021. GFlowNet Foundations. *arXiv preprint arXiv:2111.09266* (2021).

[8] Shaked Brody, Uri Alon, and Eran Yahav. 2022. How attentive are graph attention networks? *ICLR* (2022).

[9] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. 2017. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine* 34, 4 (2017), 18–42. https://doi.org/10.1109/MSP.2017.2693418

[10] Chen Cai and Yusu Wang. 2020. A Note on Over-Smoothing for Graph Neural Networks.

[11] Hongyun Cai, Vincent W Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering* 30, 9 (2018), 1616–1637.

[12] Quentin Cappart, Didier Chételat, Elias B Khalil, Andrea Lodi, Christopher Morris, and Petar Velickovic. 2021. Combinatorial optimization and reasoning with graph neural networks. (2021).

[13] Cong Chen, Chaofan Tao, and Ngai Wong. 2021. *LiteGT: Efficient and Lightweight Graph Transformers.* Association for Computing Machinery, New York, NY, USA, 161–170. https://doi.org/10.1145/3459637.3482272

[14] Jun Chen and Haopeng Chen. 2021. Edge-featured graph attention network. *arXiv preprint arXiv:2101.07671* (2021).

[15] Sanxing Chen, Xiaodong Liu, Jianfeng Gao, Jian Jiao, Ruofei Zhang, and Yangfeng Ji. 2021. HittER: Hierarchical Transformers for Knowledge Graph Embeddings. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (2021). https://doi.org/10.18653/v1/2021.emnlp-main.812

[16] Sanxing Chen, Xiaodong Liu, Jianfeng Gao, Jian Jiao, Ruofei Zhang, and Yangfeng Ji. 2021. HittER: Hierarchical Transformers for Knowledge Graph Embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 10395–10407.

[17] Krzysztof Choromanski, Han Lin, Haoxian Chen, and Jack Parker-Holder. 2021. Graph Kernel Attention Transformers. arXiv:2107.07999 [cs.LG]

[18] Kaustubh Dhole, Gurdeep Singh, Priyadarshini P Pai, and Sukanta Mondal. 2014. Sequence-based prediction of protein–protein interaction sites with L1-logreg classifier. *Journal of theoretical biology* 348 (2014), 47–54.

[19] Vijay Prakash Dwivedi and Xavier Bresson. 2020. A Generalization of Transformer Networks to Graphs. (2020). arXiv:2012.09699 [cs.LG]

[20] Vijay Prakash Dwivedi, Chaitanya K Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2020. Benchmarking Graph Neural Networks. *arXiv preprint arXiv:2003.00982* (2020).

[21] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2021. Graph Neural Networks with Learnable Structural and Positional Representations. *ArXiv* abs/2110.07875 (2021).

[22] Lise Getoor. 2005. Link-based classification. In *Advanced methods for knowledge discovery from complex data*. Springer, 189–207.

[23] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems* 30 (2017).

[24] William L Hamilton. 2020. Graph representation learning. *Synthesis Lectures on Artifical Intelligence and Machine Learning* 14, 3 (2020), 1–159.

[25] Tiantian He, Yew Ong, and Lu Bai. 2021. Learning Conjoint Attentions for Graph Neural Nets. *Advances in Neural Information Processing Systems* 34 (2021).

[26] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems* 33 (2020), 22118–22133.

[27] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous Graph Transformer. *Proceedings of The Web Conference 2020* (Apr 2020). https://doi.org/10.1145/3366423.3380027

[28] Zengfeng Huang, Shengzhong Zhang, Chong Xi, Tang Liu, and Min Zhou. 2021. *Scaling Up Graph Neural Networks Via Graph Coarsening.* Association for Computing Machinery, New York, NY, USA, 675–684. https://doi.org/10.1145/3447548.3467256

[29] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2018. Junction tree variational autoencoder for molecular graph generation. In *International conference on machine learning*. PMLR, 2323–2332.

[30] Emily A Jones and Edward G Carr. 2004. Joint attention in children with autism: Theory and intervention. *Focus on autism and other developmental disabilities* 19, 1 (2004), 13–26.

[31] Johannes Klicpera, Florian Becker, and Stephan Günnemann. 2021. Gemnet: Universal directional graph neural networks for molecules. *Advances in Neural Information Processing Systems* 34 (2021).

[32] Devin Kreuzer, Dominique Beaini, William L. Hamilton, Vincent Létourneau, and Prudencio Tossou. 2021. Rethinking Graph Transformers with Spectral Attention. In *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (Eds.). https://openreview.net/forum?id=huAdB-Tj4yG

[33] Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. 2010. Hyperbolic geometry of complex networks. *Physical Review E* 82, 3 (2010), 036106.

[34] John Boaz Lee, Ryan A. Rossi, Sungchul Kim, Nesreen K. Ahmed, and Eunyee Koh. 2019. Attention Models in Graphs. *ACM Transactions on Knowledge Discovery from Data* 13, 6 (Dec 2019), 1–25. https://doi.org/10.1145/3363574

[35] Da Li, Ming Yi, and Yukai He. 2022. LP-BERT: Multi-task Pre-training Knowledge Graph BERT for Link Prediction. arXiv:2201.04843 [cs.CL]

[36] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*. https://openreview.net/forum?id=SJiHXGWAZ

[37] Qi Liu, Maximilian Nickel, and Douwe Kiela. 2019. Hyperbolic graph neural networks. *Advances in Neural Information Processing Systems* 32 (2019).

[38] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. 2019. Provably powerful graph networks. *Advances in neural information processing systems* 32 (2019).

[39] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. 2019. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 4602–4609.

[40] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. 2019. Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks.. In *AAAI*. 4602–4609. https://doi.org/10.1609/aaai.v33i01.33014602

[41] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. 2020. Self-supervised graph transformer on large-scale molecular data. *arXiv preprint arXiv:2007.02835* (2020).

[42] Vıctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. 2021. E (n) equivariant graph neural networks. In *International Conference on Machine Learning*. PMLR, 9323–9332.

[43] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*. Springer, 593–607.

[44] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Pitfalls of Graph Neural Network Evaluation. *CoRR* abs/1811.05868 (2018). http://arxiv.org/abs/1811.05868

[45] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and S Yu Philip. 2016. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering* 29, 1 (2016), 17–37.

[46] Abraham Albert Ungar. 2008. A gyrovector space approach to hyperbolic geometry. *Synthesis Lectures on Mathematics and Statistics* 1, 1 (2008), 1–194.

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[48] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[49] Rui Wang, Bicheng Li, Shengwei Hu, Wenqian Du, and Min Zhang. 2019. Knowledge graph embedding via graph attenuated attention networks. *IEEE Access* 8 (2019), 5212–5224.

[50] Shoujin Wang, Liang Hu, Yan Wang, Xiangnan He, Quan Z. Sheng, Mehmet A. Orgun, Longbing Cao, Francesco Ricci, and Philip S. Yu. 2021. Graph Learning based Recommender Systems: A Review. arXiv:2105.06339 [cs.IR]

[51] Tianming Wang, Xiaojun Wan, and Hanqi Jin. 2020. AMR-To-Text Generation with Graph Transformer. *Transactions of the Association for Computational Linguistics* 8 (2020), 19–33. https://doi.org/10.1162/tacl_a_00297

[52] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In *The World Wide Web Conference*. 2022–2032.

[53] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. 2019. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)* 38, 5 (2019), 1–12.

[54] Richard C Wilson, Edwin R Hancock, Elżbieta Pekalska, and Robert PW Duin. 2014. Spherical and hyperbolic embeddings of data. *IEEE transactions on pattern analysis and machine intelligence* 36, 11 (2014), 2255–2269.

[55] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* 32, 1 (2020), 4–24.

[56] Yiqing Xie, Sha Li, Carl Yang, Raymond Chi-Wing Wong, and Jiawei Han. 2020. When Do GNNs Work: Understanding and Improving Neighborhood Aggregation.. In *IJCAI*. 1303–1309.

[57] Keyulu Xu, Jingling Li, Mozhi Zhang, Simon S. Du, Ken ichi Kawarabayashi, and Stefanie Jegelka. 2020. What Can Neural Networks Reason About?. In *International Conference on Learning Representations*. https://openreview.net/forum?id=rJxbJeHFPS

[58] Fang Yang, Kunjie Fan, Dandan Song, and Huakang Lin. 2020. Graph-based prediction of Protein-protein interactions with attributed signed graph embedding.

BMC bioinformatics 21, 1 (2020), 1–16.

[59] Junhan Yang, Zheng Liu, Shitao Xiao, Chaozhuo Li, Defu Lian, Sanjay Agrawal, Amit Singh, Guangzhong Sun, and Xing Xie. 2021. GraphFormers: GNN-nested Transformers for Representation Learning on Textual Graph. *Advances in Neural Information Processing Systems* 34 (2021).

[60] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do Transformers Really Perform Bad for Graph Representation? *arXiv preprint arXiv:2106.05234* (2021).

[61] Jiaxuan You, Rex Ying, and Jure Leskovec. 2019. Position-aware graph neural networks. In *International Conference on Machine Learning*. PMLR, 7134–7143.

[62] Adamo Young, Bo Wang, and Hannes Röst. 2021. MassFormer: Tandem Mass Spectrum Prediction with Graph Transformers. arXiv:2111.04824 [cs.LG]

[63] Jiawei Zhang. 2020. G5: A Universal GRAPH-BERT for Graph-to-Graph Transfer and Apocalypse Learning. arXiv:2006.06183 [cs.LG]

[64] Jiawei Zhang. 2020. Graph Neural Distance Metric Learning with Graph-Bert. arXiv:2002.03427 [cs.LG]

[65] Jiawei Zhang. 2020. Segmented Graph-Bert for Graph Instance Modeling. arXiv:2002.03283 [cs.LG]

[66] Jiawei Zhang and Lin Meng. 2019. Gresnet: Graph residual network for reviving deep gnns from suspended animation. *arXiv preprint arXiv:1909.05729* (2019).

[67] Jiani Zhang, Xingjian Shi, Junyuan Xie, Hao Ma, Irwin King, and Dit-Yan Yeung. 2018. GaAN: Gated Attention Networks for Learning on Large and Spatiotemporal Graphs. arXiv:1803.07294 [cs.LG]

[68] Hao Zhou, Yazhou Yang, Tingjin Luo, Jun Zhang, and Shuohao Li. 2022. A unified deep sparse graph attention network for scene graph generation. *Pattern Recognition* 123 (2022), 108367. https://doi.org/10.1016/j.patcog.2021.108367

[69] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open* 1 (2020), 57–81.

[70] Yi Zou, Pei Zou, Yi Zhao, Kaixiang Zhang, Ran Zhang, and Xiaorui Wang. 2021. MELONS: generating melody with long-term structure using transformers and structure graph. arXiv:2110.05020 [cs.SD]