《DySAT: Deep Neural Representation Learning on Dynamic Graphs via Self-Attention Networks》

《DySAT：基于自注意网络的动态图的深度神经表示学习》

摘要：

1. **本文的背景：**近年来，许多研究表明图节点嵌入表示学习对于链接预测、节点分类等任务十分有效。但现有的图嵌入表示学习方法主要是针对静态图的，而现实世界中的许多图网络却是会随时间演变的。因此，针对复杂的动态图结构问题，本文旨在提出一种应用于动态图的新型且有效的框架来学习图节点嵌入表示。
2. **本文的贡献：**提出了一种新型的神经网络架构——动态自注意网络（$DySAT$），通过学习节点表征来捕捉动态图的结构演化。
3. **主要创新点：**通过在两个维度 **structural neighborhood** 和 **temporal dynamics** 的自注意力机制 Self-Attention 来学习节点表示，并且使用多头注意力来捕捉多方面的动态性。【GAT + Transformer：（multi-head）】
4. **实验结果：**本文在两种类型的图网络上进行了链接预测实验：通信网络和双方差网络。而实验结果表明，在单步和多步链接预测任务中，$DySAT$的性能都明显优于几种最先进的图嵌入基线方法。

动态图（dynamic graph）定义：

动态图是拓扑和(或)属性随时间变化的图，即动态图中的结构和属性都可能随时间而变化——添加或删除边和(或)节点，其属性也可能因时间而更改。此外，动态图通常又分为**连续时间动态图**和**离散时间动态图**，而本文主要考虑离散时间动态图（Discrete Time Dynamic Graphs，DTDG）。它可以看作由T个静态图组成的序列，如下式所示：

$$G = (\mathcal{G}^1, \mathcal{G}^2, \dots, \mathcal{G}^T)$$

其中，$\mathcal{G}^t = (V, \mathcal{E}^t)$ 是第t个时刻的静态图；所有静态图都共享一个结点集V，但每个静态的图有各自的边集$\mathcal{E}^t$（注意：本论文中的边集是无向加权的）。而由于边集不同，所以每个静态图有各自的邻接矩阵A^t。

动态图表征学习的目标是：对于每个节点$v \in V$，我们旨在学习它在时间t处的嵌入表示$e_v^t \in \mathbb{R}^d$，使得e_v^t能保存节点v的局部结构信息，同时能保存它截止到t时刻的变化行为（比如它在过去增加了边，移除了边的行为）。

$DySAT$整体框架：

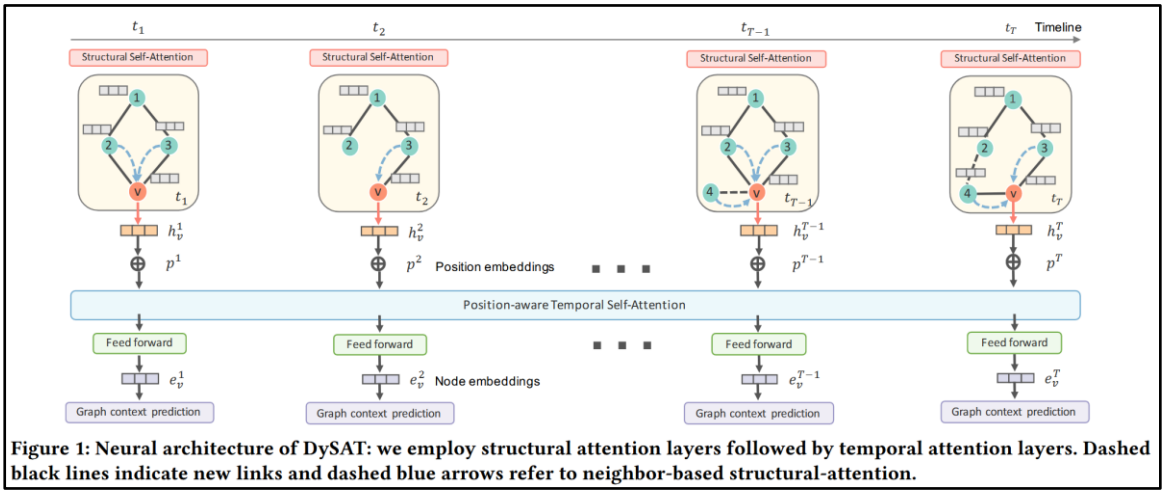

Figure 1: Neural architecture of DySAT: we employ structural attention layers followed by temporal attention layers. Dashed black lines indicate new links and dashed blue arrows refer to neighbor-based structural-attention.

$DySAT$的整体框架如上图所示，可以看到模型框架主要分为两个部分：结构自注意力模块（Structural Self-Attention）和时序自注意力模块（Temporal Self-Attention）。而在结构自注意力模块中，黑色虚线表示基于时序产生的新连接，蓝色虚线表示基于邻居的结构注意力。

结构自注意力（Structural Self-Attention）：

该模块与GAT中的注意力机制类似，相当于一个邻居节点信息汇聚层。对于每一个 snapshot graph，Structural Self-Attention 利用当前时刻各个节点的表征计算注意力再进行加权求和。整个过程的计算公式如下：

（1）将图上的每个节点都用一个D维的$one-hot$向量来表示：

$$x_v \in \mathbb{R}^D, \ \forall v \in V$$

其中，D是节点的总数量。

（2）计算每个节点对应的一个嵌入向量：

$$W^s x_v \in \mathbb{R}^F, \ \forall v \in V$$

其中，$W^s \in \mathbb{R}^{F \times D}$是所有节点共享的嵌入矩阵。

（3）对于节点v，记它的邻居节点们为$\mathcal{N}_v = \{u \in V : (u, v) \in \mathcal{E}\}$，然后计算$v$对每

个邻居u的注意力值：

$$e_{uv} = LeakyRelu(A_{uv} \cdot a^T(W^s x_u || W^s x_v)), \qquad \forall (u,v) \in \mathcal{E}$$

上式通过将节点v的邻居的嵌入向量拼接起来，进行线性变换，再通过边权来缩放。其本质是认为两点之间的注意力取决于他们的嵌入向量以及连接强度(边权)。其中，$a \in \mathbb{R}^{2F}$。

（4）在计算出节点v对它所有邻居的注意力后，对注意力进行归一化：

$$\alpha_{uv} = \frac{\exp(e_{uv})}{\sum_{k \in \mathcal{N}_v} \exp(e_{kv})}$$

（5）将节点v的邻居们的嵌入向量按注意力值来加权聚合，得到节点v的输出向量表示z_v：

$$z_v = \sigma\left(\sum_{u \in \mathcal{N}_v} \alpha_{uv} W^s x_u\right)$$

最终，上述步骤就使得每个节点的 embedding——z_v学习到了图的局部结构。

时序自注意力（Temporal Self-Attention）：

在经过上述的 Structural Self-Attention 后，图上的任一节点在每个时刻的隐藏表示已经包含了该点的局部结构信息。然后，我们在此基础上，按照 time step 进行 concat 拼接，并加上 position embedding，得到时序自注意力模块的输入：

$$X_v = \begin{pmatrix} h_v^1 + p^1 \\ h_v^2 + p^2 \\ \cdots \\ h_v^T + p^T \end{pmatrix} \in \mathbb{R}^{T \times F}$$

其中，p^t代表第t时刻的 position embedding，h_v^t代表节点v在第t时刻的隐藏嵌入表示。

接着，我们利用 Transformer 中的操作，来捕获在图结构上的多个时间步下的时序信息变化。具体过程如下：

（1）计算任一时刻t对所有时刻的注意力值（包括t本身）：

首先，将节点v的所有时刻作为 query 角色映射到F'维的 query 空间，即$X_v W_q \in \mathbb{R}^{T \times F'}$，其中$W_q \in \mathbb{R}^{F \times F'}$；再将节点$v$所有时刻作为 key 角色映射到 key 空间，即

$X_v W_k \in \mathbb{R}^{T\times F'}$，其中$W_k \in \mathbb{R}^{F\times F'}$。然后，将时刻$i$作为 query 查询时刻$j$（作为 key）的注意力值为$\frac{((X_v W_q)(X_v W_k)^T)_{ij}}{\sqrt{F'}}$，其中$\sqrt{F'}$作为缩放因子，目的是避免权值过大。最后，为实现任一时刻只注意到其前的时刻，而无法注意其后时刻，再为注意力值加上一个掩码值M_{ij}：

$$M_{ij} = \begin{cases} 0, & i \le j \\ -\infty, & otherwise \end{cases}$$

因此，掩码后的注意力值为：

$$e_v^{ij} = \left(\frac{\left((X_v W_q)(X_v W_k)^T \right)_{ij}}{\sqrt{F'}} + M_{ij} \right)$$

（2）将求得的注意力值归一化，即时刻i对所有时刻的注意力值归一化（转化为注意力系数）：

$$\beta_v^{ij} = \frac{\exp\left(e_v^{ij}\right)}{\sum_{k=1}^{T} \exp\left(e_v^{ik}\right)}$$

其中，$\beta_v \in \mathbb{R}^{T\times T}$。

（3）以注意力为权重，聚合所有时刻的特征，得到每个时刻的输出特征Z_v：

$$Z_v = \beta_v (X_v W_v)$$

其中，$Z_v \in \mathbb{R}^{T\times F'}$；$W_v \in \mathbb{R}^{F\times F'}$，是节点$v$的所有时刻作为 output 角色映射到$F'$维的 output 空间的共享权重参数。

多头注意力机制（MULTI-HEAD ATTENTION）：

此外，本文还采取了多头注意力的机制（类似CNN中的改变通道数，对节点的每个表示使用多通道叠加的概念），具体操作过程如下：

1）在结构自注意力模块中——

$$h_v = Concat(z_v^1, z_v^2, \dots, z_v^{HEAD}),\ \forall v \in V$$

2）在时序自注意力模块中——

$$H_v = Concat(Z_v^1, Z_v^2, \dots, Z_v^{HEAD}),\ \forall v \in V$$

其中，$h_v \in \mathbb{R}^F$，$H_v \in \mathbb{R}^{T\times F'}$。

损失函数与下游任务：

动态连接预测任务：

single-step link prediction：给定t时刻的信息，预测$t+1$时刻的边存在的概率；

multi-step link prediction：给定t时刻的信息，预测$t+1,\ldots,t+n$时刻的边存在的概率。

转化为二分类任务：

t时刻实际存在的边——1；t时刻负采样的边——0。

$DySAT$模型的损失函数：

$$L=\sum_{t=1}^{T}\sum_{v\in V}\Big(\sum_{u\in\mathcal{N}_{walk}^{t}(v)}-\log\big(\sigma(<e_u^t,e_v^t>)\big)-w_n\cdot\sum_{u'\in p_n^t(v)}\log\Big(1-\sigma\big(<e_{u'}^t,e_v^t>\big)\Big)\Big)$$

其中，w_n为模型的超参数，$\mathcal{N}_{walk}^{t}(v)$代表第$t$时刻的节点$v$通过随机游走得到的上下文节点；$p_n^t(v)$代表第$t$时刻的节点$v$通过负采样得到的样本节点，$e_v^t$代表第$t$时刻的节点$v$的嵌入表示向量。

实验设置及结果：

1）数据集：

	Communication		Rating	
Attribute	**Enron**	**UCI**	**Yelp**	**ML-10M**
# of Nodes	143	1,809	6,569	20,537
# of Links	2,347	16,822	95,361	43,760
# of Time steps	12	13	12	13

Table 1: Summary statistics of four datasets. Link counts include the number of snapshots containing each link, *e.g.*, a link that occurs in three snapshots, is counted thrice.

数据集摘要如上表所示，本次实验共涉及 **2 个通信图数据集：**• Enron：边表示员工之间的电子邮件交互；•UCI：边表示在线社交网络平台上的用户之间发送的消息；和 **2 个评分图数据集：**• Yelp：由用户和企业组成，边表示随时间变化的观察评级；•ML-10M：由用户和标签的交互组成，边表示将用户与他们在某些电影上的标签连接起来。

2）评价指标：

因为每次都是根据t来预测$t+1$，所以每个 time step 都会有一个二分类的统计结果：

- **micro AUC**：所有 time step 边实例的总 AUC；
- **macro AUC**：每个 time step 的 AUC 的平均值。

3）实验结果：

单步预测任务：

Method	Enron		UCI		Yelp		ML-10M	
	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC
node2vec	83.72 ± 0.7	83.05 ± 1.2	79.99 ± 0.4	80.49 ± 0.6	67.86 ± 0.2	65.34 ± 0.2	87.74 ± 0.2	87.52 ± 0.3
G-SAGE	82.48* ± 0.6	81.88* ± 0.5	79.15* ± 0.4	82.89* ± 0.2	60.95† ± 0.1	58.56† ± 0.2	86.19‡ ± 0.3	89.92‡ ± 0.1
G-SAGE + GAT	72.52 ± 0.4	73.34 ± 0.6	74.03 ± 0.4	79.83 ± 0.2	66.15 ± 0.1	65.09 ± 0.2	83.97 ± 0.3	84.93 ± 0.1
GCN-AE	81.55 ± 1.5	81.71 ± 1.5	80.53 ± 0.3	83.50 ± 0.5	66.71 ± 0.2	65.82 ± 0.2	85.49 ± 0.1	85.74 ± 0.1
GAT-AE	75.71 ± 1.1	75.97 ± 1.4	79.98 ± 0.2	81.86 ± 0.3	65.92 ± 0.1	65.37 ± 0.1	87.01 ± 0.2	86.75 ± 0.2
DynamicTriad	80.26 ± 0.8	78.98 ± 0.9	77.59 ± 0.6	80.28 ± 0.5	63.53 ± 0.3	62.69 ± 0.3	88.71 ± 0.2	88.43 ± 0.1
DynGEM	67.83 ± 0.6	69.72 ± 1.3	77.49 ± 0.3	79.82 ± 0.5	66.02 ± 0.2	65.94 ± 0.2	73.69 ± 1.2	85.96 ± 0.3
DynAERNN	72.02 ± 0.7	72.01 ± 0.7	79.95 ± 0.4	83.52 ± 0.4	69.54 ± 0.2	68.91 ± 0.2	87.73 ± 0.2	89.47 ± 0.1
DySAT	**85.71** ± 0.3	**86.60** ± 0.2	**81.03** ± 0.2	**85.81** ± 0.1	**70.15** ± 0.1	**69.87** ± 0.1	**90.82** ± 0.3	**93.68** ± 0.1

Table 2: Single-step link prediction results (micro and macro averaged AUC with std. deviation). We show GraphSAGE (denoted by G-SAGE) results with the best performing aggregators (∗, †, and ‡ represent GCN, LSTM, and max-pooling respectively). DySAT achieves significant performance gains of 4.8% macro-AUC on average across all datasets.

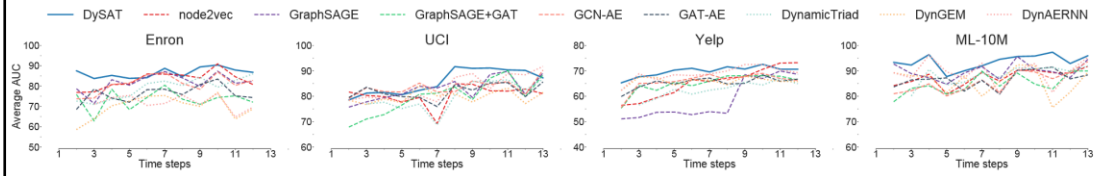

Figure 2: Performance comparison of DySAT with different graph representation learning methods on single-step link prediction: solid line denotes DySAT; dashed and dotted lines denote static and dynamic graph embedding baselines respectively.

如图所示，可以看到在单步预测任务上，$DySAT$相比基线方法，在各个类型的数据集上都取得了最好的结果。

多步预测任务：

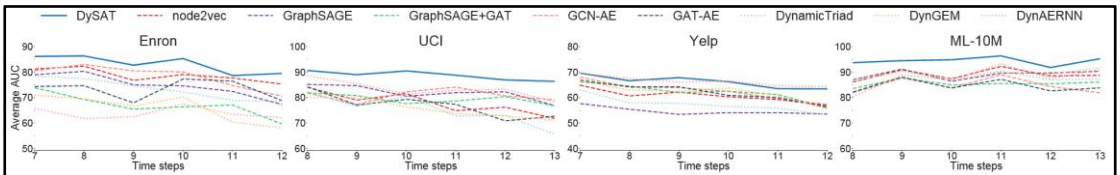

Figure 3: Performance comparison of DySAT with different models on multi-step link prediction for the next 6 time steps. DySAT outperforms competing baselines by a large margin, and maintains consistently high performance over time.

如图所示，可以看到在多步预测任务上，$DySAT$相比基线方法，在各个类型的数据集上也都取得了最好的结果。

单步预测（只评估未出现过的边）：

Method	Enron		UCI		Yelp		ML-10M	
	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC
node2vec	76.92 ± 1.2	75.86 ± 0.5	73.67 ± 0.3	74.76 ± 0.8	67.36 ± 0.2	65.17 ± 0.2	85.22 ± 0.2	84.89 ± 0.1
G-SAGE	73.92* ± 0.7	74.67* ± 0.6	76.69* ± 0.3	79.41* ± 0.1	62.25† ± 0.2	58.81† ± 0.3	85.23‡ ± 0.3	89.14‡ ± 0.2
G-SAGE + GAT	67.02 ± 0.8	68.32 ± 0.7	73.18 ± 0.4	76.79 ± 0.2	66.53 ± 0.2	65.45 ± 0.1	80.84 ± 0.3	82.53 ± 0.1
GCN-AE	74.46 ± 1.1	74.02 ± 1.6	74.76 ± 0.1	76.75 ± 0.6	66.18 ± 0.2	65.77 ± 0.3	82.45 ± 0.3	82.48 ± 0.2
GAT-AE	69.75 ± 2.2	69.25 ± 1.9	72.52 ± 0.4	73.78 ± 0.7	66.07 ± 0.1	65.91 ± 0.2	84.98 ± 0.2	84.51 ± 0.3
DynamicTriad	69.59 ± 1.2	68.77 ± 1.7	67.97 ± 0.7	71.67 ± 0.9	63.76 ± 0.2	62.83 ± 0.3	84.72 ± 0.2	84.32 ± 0.2
DynGEM	60.73 ± 1.1	62.85 ± 1.9	77.49 ± 0.3	79.82 ± 0.5	66.42 ± 0.2	66.84 ± 0.2	73.77 ± 0.7	83.51 ± 0.3
DynAERNN	59.05 ± 2.7	59.63 ± 2.7	77.72 ± 0.5	81.91 ± 0.6	**74.33** ± 0.2	**73.46** ± 0.2	87.42 ± 0.2	88.19 ± 0.2
DySAT	**78.87** ± 0.6	**78.58** ± 0.6	**79.24** ± 0.3	**83.66** ± 0.2	69.46 ± 0.1	69.14 ± 0.1	**89.29** ± 0.2	**92.65** ± 0.1

Table 3: Single-step link prediction results restricted to *new* links (micro and macro averaged AUC with std. deviation). All methods achieve lower AUC scores in comparison to predicting all links; DySAT outperforms baselines on most datasets.

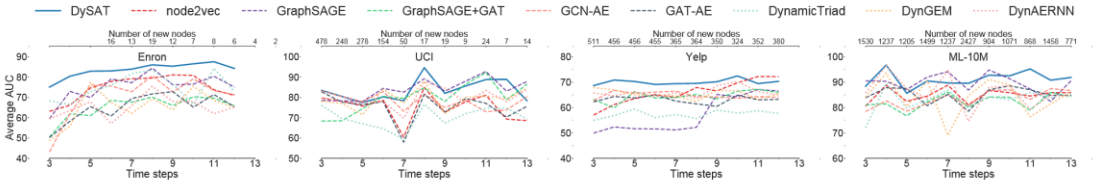

Figure 4: Performance comparison of DySAT with different graph embedding models on single-step link prediction restricted to previously unseen *new* nodes at each time step. Despite the lack of historical context for an unseen node, DySAT outperforms competing baselines due to its ability to factor the temporal evolution of its neighbors to compute its representation.

如图所示，可以看到在单步预测（只评估未出现过的边）任务中，在大多数数据集上，$DySAT$都优于基线方法。

消融实验：

Method	Enron		UCI		Yelp		ML-10M	
	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC	Micro-AUC	Macro-AUC
Original	**85.71** ± 0.3	**86.60** ± 0.2	**81.03** ± 0.2	**85.81** ± 0.1	**70.15** ± 0.1	**69.87** ± 0.1	**90.82** ± 0.3	**93.68** ± 0.1
No Structural	85.21 ± 0.2	86.50 ± 0.3	74.48 ± 0.3	81.24 ± 0.2	70.11 ± 0.1	67.85 ± 0.1	88.56 ± 0.2	90.34 ± 0.2
No Temporal	84.50 ± 0.3	85.68 ± 0.4	76.61 ± 0.2	79.97 ± 0.3	68.34 ± 0.1	67.20 ± 0.3	89.61 ± 0.4	91.10 ± 0.2

Table 4: Ablation study on structural and temporal attention layers (micro and macro AUC with std. deviation). DySAT (joint structural and temporal attention) outperforms the ablation variants by a margin 3% Macro-AUC on average.

实验结果证明了结构自注意力模块和时序自注意力模块的有效性。

结构注意力头和时序注意头数量的敏感性分析：

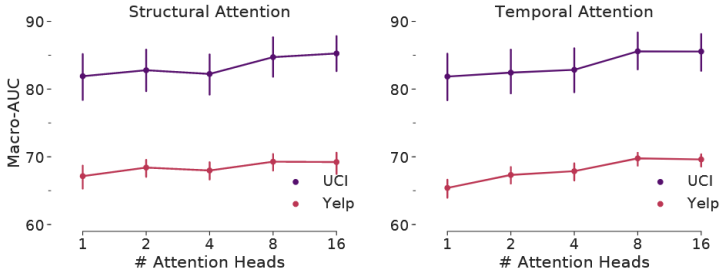

Figure 5: Sensitivity analysis on the number of structural and temporal attention heads in DySAT (UCI and Yelp). Model performance stabilizes with 8 attention heads.

可扩展性比较：

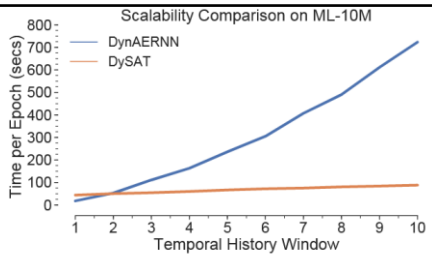

Figure 6: Scalability comparison of DySAT with DynAERNN. DySAT is faster by nearly an order of magnitude.

如图所示，可以看到$DySAT$具有较好的可扩展性，并且$DySAT$的速度快了近一个数量级。

注意力权重热图：

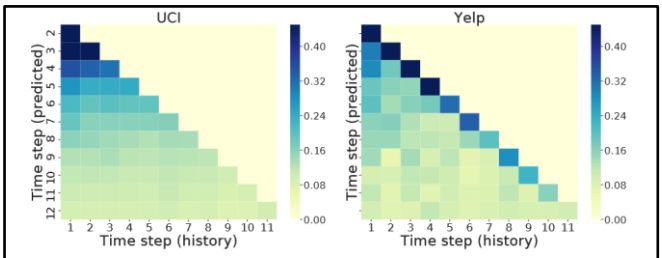

Figure 7: Heatmap of mean temporal attention weights on UCI and Yelp. Attention weights are uniformly distributed in UCI, while Yelp exhibits a bias towards recent snapshots.

可以看到，在不同的数据集中，对注意力权重的分布是有明显区别的。

结论：

本文提出了 structural 和 temporal self-attention 来学习动态图中的结点表征，并且使用多头注意力机制捕捉多方面的动态性，且注意力机制适用于并行。但它的局限性在于只能适用于结点不变化的动态图，且以结点的共现率为损失函数引导模型的训练，这样的损失函数可能重点关注的是图的结构，而对于动态性更丰富的图（如结点的特征的变化）是不够的。

对本文的感悟：

总得来说，$DySAT$是利用GAT来得到图的结构信息（保持局部特征），然后利用含有 position encoding 的$Transformer$来建模时序信息（学习时间演化信息）的。它是一篇很好的结合性文章，而它的创新点主要在于有效结合不同的模型框架以解决更加棘手的问题，这为我们提供了一类很好地科研思路。