动态图神经网络综述

动态网络模型在静态网络的基础上增加了时间维度，使其能同时表征复杂系统的结构和时序信息，在生物、医药、社交网络等领域被广泛使用。另外，虽然图神经网络（GNN）在静态复杂网络的数据挖掘中披荆斩棘，但大多工作都不能处理这额外的时间维度。考虑到真实网络大多都是时变的复杂网络，处理这种网络的动态图神经网络（DGNN）架构必会是一个重要的研究方向。最近的一篇综述系统的总结了该方向的成果，可用来了解这个方兴未艾的研究方向。

该综述的作者是悉尼科技大学（UTC）的 Katarzyna Musial-Gabrys 团队，Katarzyna 主要关注复杂网络的动态和演化的分析与网络的结构和性质的建模。最近，她的团队将精力集中在利用机器学习和可预测性模型来研究动态复杂网络。

该综述可以分为以下几部分：

1. 动态网络的分类和表示；
2. 作为动态网络编码器的 DGNN；
3. 动态网络表征的解码与结构预测：以链路预测为例。

一、动态网络的分类和表示

动态（Dynamic）网络与静态（Static）网络的区别在于前者具有：1）动态结构，连边和节点会随时间消失/出现，2）动态属性，节点和连边状态会随时间而改变。我们只考虑具有动态结构的动态网络，并从时间尺度上对动态网络进行区分和讨论，以及动态网络的离散和连续型表示方法。

1.1 动态网络的分类

网络的结构的动态变化体现为网络连边和节点的出现和消失，这种变化的不同决定了动态网络的性质。例如，Email 网络中的新连边（互发邮件）瞬时出现和消失，而文献引用网络中的连边（相互引用）出现 后便永不会消失。根据这种变化的持续时间对动态网络进行分类，可以区分动态网络的演化机理。 首先考虑连边出现到消失的时间（持续时间），按照连边持续时间长短，可以在时间轴上将动态网络分 为如图 4 类：

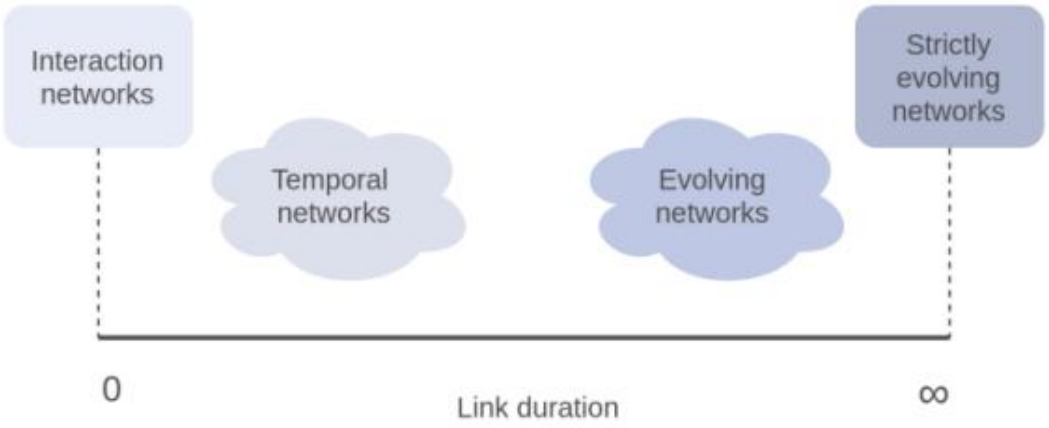

图 1：横轴表示动态网络中连边出现后持续的时间，以此可以将动态网络分为 4 类

其中，交互网络（Interaction networks） 和 严格演化网路（Strictly evolving networks） 分别于上述 email 和 文献引用网络的例子对应，是连边持续时间的两个极端：出现后瞬间消失或者永远持续。这两

种情况是比较容易区分的，在接下来的分类中不做讨论。

但是对处于时间轴中段的动态网络，其区分方式就比较微妙了。这些动态网络的连边都是在出现后持续一段有限时间后消失，根据这“有限时间”的长短，可以将其分为时序（Temporal）和 演化（Evolving）网络。 时序网络具有高度动态的连边，连边持续时间很短，以致于在任一时刻对网络结构进行采样，会发现很难采样出一个完整的网络结构。例如，人际网络中，将人与人开始和结束交谈定义为其连接的出现和消失，虽然每个连接都会持续一段时间（几分钟或者几小时），但是在任意时刻对该网络进行采样，都会发现大部分人都是孤立节点（没有正在与人交谈），也就没有完整的网络结构。

相反，演化网络的连边会持续显著长的时间。对于这种网络，如果在某一时刻对其进行采样，能够得到 比较完整的网络结构。比较典型的例子就是雇佣关系网络，其中的雇佣关系会持续若干年， 对该网络采样，大部分人都会用对应的连边（只要没有失业）。

上述对于连边这种持续时间长短的考虑，有助于我们选择合适的方法来表示（与记录）动态网络数据。

另外一方面，对于网络的节点，本文简单的将其分类为静态和动态两类。

这样，就可以从连边和节点动力学的角度，将动态网络分类为如下 4 类，对应于连边是时序还是演化，节点是静态还是动态：

		Link duration	
		Temporal	Evolving
Node dynamics	Static	Node static temporal	Node static evolving
	Dynamic	Node dynamic temporal	Node dynamic evolving

图 2：按照连边的持续时间和节点的动态将动态网络划分为 4 类

1.2 动态网络的表示

传统的时序数据一般用离散和连续两种方式表示。前者对时间轴等距取窗口，同一个窗口内的数据被聚合成一个时间步，最后用离散化的时间步来表示连续的时序数据。连续表示会记录所有数据点的精确时间，能够完整保存所有的时序信息。类似的，动态网络的表示也主要分为离散和连续两种形式。

动态网络的离散表示本质上是用有序的静态网络序列

$$DG = \{G^1, G^2, \ldots, G^T\},$$

来表示动态网络。该静态网络序列可以看做是对动态网络按照时间步在时间轴上的一系列快照（Snapshot）（也可以表示成多层网络或张量）。这种表示方法简单、直观，并且可以无痛的利用处理静态网络的 GNN 来处理这些快照，从而完成对动态网络数据的处理。但是这种表示方法必然会损失网络的时序信息，而且时间步长的选择还不能兼顾计算效率和精度。

动态网络的连续表示致力于记录所有动态变换（或称 Event）的开始和结束时间，从而保留所有的时序信息。主要分为 3 种形式：

1）演化网络可以表示为一系列四元组的集合

$$EB = \{(u_i, v_i, t_i, \Delta_i); i = 1, 2, \ldots\}$$

，集合中的元素记录着各条动态连边的两个端点、出现时间和持续时间。这种表示方法称为”基于事件的表示”；

2）如果连边持续时间太短或者不重要，还可以根据需要不记录持续时间，这样的序列表示

$$CS = \{(u_i, v_i, t_i); i = 1, 2, \ldots\}$$

称为“接触序列”；

3）当具体时间不重要，只有相对时间重要时，可以简单的用有序连边序列

$$GS = \{e_1, e_2, \ldots\}$$

来表示动态网络，这种表示称为“流图（Graph stream）”。

具体使用哪种表示方法和实际的数据已经研究的侧重点有关。连续化的表示虽然能够最大程度上的保留时序信息，但是却使得数据难以处理，不能简单的挪用静态网络的表征方法。

此外，到底应该将一个实际的动态网络表示成离散的或是连续的，除了上面的考量，还必须该网络本身的动态特性。对于时序网络，用连续表示更合适；对于演化网络，用离散表示更方便。读者可以思考下其中缘由。

二. 动态图神经网络（DGNN）

本文考虑的图神经网络特指具有领域聚合操作的 GNN，属于网络表示学习的一个子方向。读者想了解整个图表示学习领域对于动态网络的研究，可以参考 19 年发表于 Journal of Machine Learning Research 的综述。

动态网络一般可以表示为离散和连续两种形式，那么按照能处理哪些不同形式的动态网络数据，可以将目前的动态图神经网络的工作主要分为离散和连续两大类，其中又主要以处理离散数据的工作居多。下图中还有一个分类，伪动态（Pseudo-dynamic），这类神经网络虽然建模了动态过程，但不会被用来拟合真实数据，所以下文不再考虑。具体工作可参考 Da Xu 等人 19 年的工作。

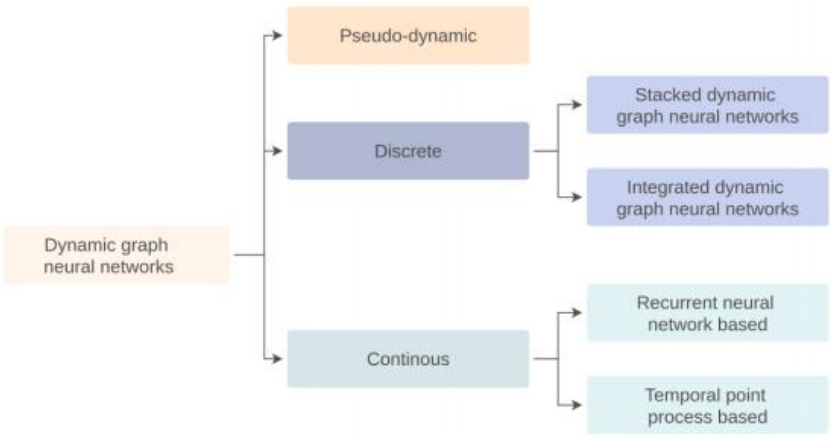

图 3：对于动态图神经网络的分类

2.1 离散的 DGNN

顾名思义，离散 DGNN 是用来处理离散表示的网络数据的框架。该类框架分为两个关键模块：1）对于单个时间步的静态网络，一般直接采用经典 GCN、GAT 等静态 GNN 模型；2）对于离散时序信息的捕获一般采用 RNN 架构。也可以自注意力架构来捕获时序信息。

如上图所示，采用 GNN+RNN 的离散 DGNN 架构的工作主要分为两类，其一称为堆叠动态图网络（Stacked dynamic gnn），其二称为整合动态图网络（Intergrated dynamic gnn）。两者的主要区别在于如何把 GNN 模块和 RNN 的模块相互整合。

2.1.1 堆叠动态图网络

堆叠动态图网络的基本思路是：用 GNN 编码单个时刻的图的信息，再用 RNN （一般使用 LSTM）的门控机制完成图信息在时间轴上的传递和编码。整体结构由 GNN 与 RNN 依次堆叠而成。最简单的形式如下：

$$
\begin{aligned}
z_1^t, \ldots, z_n^t &= \mathrm{GNN}\left(G^t\right) \\
h_j^t &= f\left(h_j^{t-1}, z_j^t\right) \ \text{for } j \in [1, n]
\end{aligned}
$$

其中 GNN 用来处理时刻 t 时刻的静态网络，结合 RNN 的隐状态来更新节点的表示。其中 f()表示 RNN 结构。Youngjoo 等人在 18 年的工作介绍了上述框架最早的一个版本，利用了 GNN 和 peehole LSTM：

$$
\begin{aligned}
z_t &= \mathrm{GNN}\left(X_t\right) \\
i &= \sigma\left(W_i z_t + U_i h_{t-1} + w_i \odot c_{t-1} + b_i\right) \\
f &= \sigma\left(W_f z_t + U_f h_{t-1} + w_f \odot c_{t-1} + b_f\right) \\
c_t &= f_t \odot c_{t-1} \\
&\quad + i_t \odot \tanh\left(W_c z_t + U_c h_{t-1} + b_c\right) \\
o &= \sigma\left(W_o z_t + U_o h_{t-1} + w_o \odot c_t + b_o\right) \\
h_t &= o \odot \tanh\left(c_t\right)
\end{aligned}
$$

当然，在上述基本思路的指导下，堆叠动态图网络可以有很多变体。包括但不限于用不同的 GNN 或 RNN 模型，不同节点之间不共享 RNN 参数，以及用

自注意力机制代替 RNN 等等。有兴趣的读者可以参考综述原文，笔者不再赘述。

2.1.2 整合动态图网络

整合动态图网络的基本思路是将 GNN 模块和 RNN 的相互整合到同一架构中：一方面可以用图卷积替代 RNN 中线性变换模块，或者可以用 RNN 来控制 GNN 在不同时间步的参数。有两个比较典型的工作：

其一是仿照卷积长短期记忆网络 convLSTM，只不过将 LSTM 中的线性变换用图卷积操作代替，如下所示，*表示图卷积。这种 LSTM 的变形天然就可以处理图数据组成的时间序列。

$$
\begin{aligned}
f_t &= \sigma\left(W_f *_{\mathcal{G}} X_t + U_f *_{\mathcal{G}} h_{t-1} + w_f \odot c_{t-1} + b_f\right) \\
i_t &= \sigma\left(W_i *_{\mathcal{G}} X_t + U_i *_{\mathcal{G}} h_{t-1} + w_i \odot c_{t-1} + b_i\right) \\
c_t &= f_t \odot c_{t-1} \\
&\quad + i_t \odot \tanh\left(W_c *_{\mathcal{G}} X_t + U_c *_{\mathcal{G}} h_{t-1} + b_c\right) \\
o_t &= \sigma\left(W_o *_{\mathcal{G}} X_t + U_o *_{\mathcal{G}} H_{t-1} + w_o \odot c_t + b_o\right) \\
h_t &= o \odot \tanh\left(c_t\right)
\end{aligned}
$$

其二是 19 年提出的 EvolveGCN。该工作的出发点很朴素，认为处理时间序列的 GCN 的参数也应该随着时间演化（不同）。他们用 RNN 来控制和更新 GCN 在不同时间步的参数，如下图。

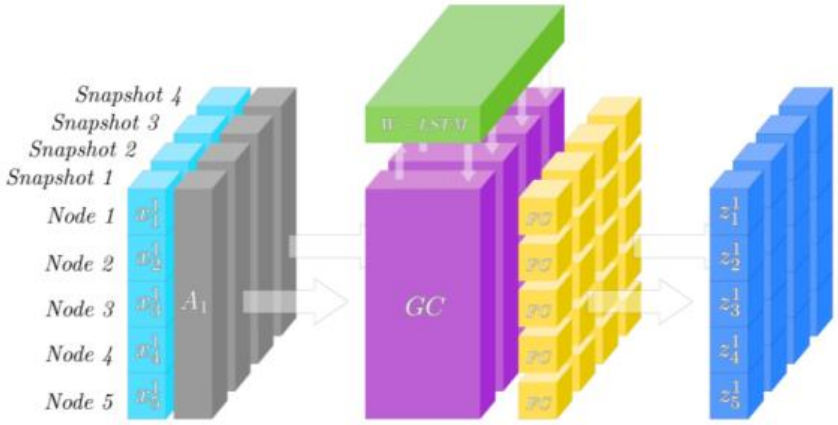

图 4：EvolveGCN 的架构示意图

其中 GCN 的参数可以看作 RNN 的隐状态或者输出，根据这个思路，他们提出了两种变体，分别用 RNN 的隐状态（EGCU-H）和输出（EGCU-O）来控制每一步 GCN 的参数， 如下所示：

EGCU-H layer is given by the following equations, where (l) indicates the neural network layer:

$$\begin{aligned} W_t^{(l)} &= \mathrm{GRU}\left(H_t^{(l)}, W_{t-1}^{(l)}\right) \\ H_t^{(l+1)} &= \mathrm{GNN}\left(A_t, H_t^{(l)}, W_t^{(l)}\right) \end{aligned} \tag{11}$$

And the EGCU-O layer is given by the equations:

$$\begin{aligned} W_t^{(l)} &= \mathrm{LSTM}\left(W_{t-1}^{(l)}\right) \\ H_t^{(l+1)} &= \mathrm{GNN}\left(A_t, H_t^{(l)}, W_t^{(l)}\right) \end{aligned} \tag{12}$$

另外，还有一些利用 Autoencoder 和 VGAE 已经 GAN 来完成离散动态图表征的工作。

2.2 连续的 DGNN

对于连续表示的动态网络，目前的工作也主要集中在两方面：其一是基于 RNN 的框架，节点的表征信息由 RNN 来传递和维持；其二是基于时序点序列（Temporal point process, TTP）的框架，通过用 NN 参数化 TTP 来完成对数据的拟合。

1）基于 RNN 的框架：这类工作的出发点很质朴，认为只要有一个变化 (Or event) 发生， 那么参与变换的节点也需要更新，使得节点的嵌入持续的更新。这种框架下主要有两个工作，其一是 18 年京东的基于流图的工作，他们的框架通过 3 部分:（i) interact uint, (ii) update/propagate unit, (iii) merge unit，完成对一个 event 对节点嵌入的更新。其中核心组件是 update/propagate 部分，他们利用了时间感知 LSTM 完成该部分的建模；第二个工作是斯坦福大学 Jure 在 19 年的工作 JODIE，他们考虑的是顾客 u 和商品 i 的二分图，利用两个 RNN 分别控制顾客和商品节点的时序信息。新的交易发生后，立即更新对应的商品和用户节点特征：

$$u(t) = \sigma\left(W_1^u u\left(\bar{t}\right) + W_2^u i\left(\bar{t}\right) + W_3^u f + W_4^u \Delta_u\right)$$
$$i(t) = \sigma\left(W_1^i i\left(\bar{t}\right) + W_2^i u\left(\bar{t}\right) + W_3^i f + W_4^i \Delta_i\right)$$

2）基于 TTP 的框架：时序点序列（TTP) 是建模异步时间序列的传统模型。而连续动态网络是一种典型的连续时间域上的异步序列。19 年的工作 DyRep 利用 NN 来参数化 TTP，使其通过优化后能够捕获网络结构演化的动态和节点的动态，通过同时建模这两个协同演化的过程，DyRep 能取得更加丰富的网络表征结果。DyRep 还是唯一能够预测事件发生时间的框架（其他只能预测事件时候发生）。近期的工作也有尝试用 NRI 和自注意力来更好的建模 TTP 过程。同时，其他的时序模型也被尝试用来建模动态网络，比如 20 年的工作 GHN 便是使用了 Hawkes 模型。

总的来说，连续 DGNN 目前只能处理特定的几种连续动态网络，比如交互网络和流图网络，更加通用、能够解决更多问题的 DGNN 还有待开发。

三． 动态网络表征的解码与结构预测

综述最后还以动态链路预测这个任务为例，总结了如何设计解码器和损失函数，进而利用上述的 DGNN 框架的表征结果来完成动态链路预测任务。这部分主要就是如何进行负采样和计算模型的约束项，感兴趣的读者可以参看原文。

另外，作者还强调了链路预测任务是一个类别极度不平衡的分类任务，所以在选择性能指标时必须谨慎。作者也总结了部分可用的性能指标，介绍如下：

1）AUC：常见的分类性能度量；

2）AUC 的改进版本 PRAUC：Yang 等人发现用 AUC 作为链路预测的性能指标时，因为链路预测的类别不平衡特点，AUC 不太具有区分度，他们提出可以用 Percision 和 Recall 分别作为横纵坐标，计算其下的面积 PRAUC 来代替传统的 AUC；

3）Precision@K：该指标能够克服 AUC 对于阈值的敏感造成的不稳定，但是 k 值不容易选择；

4）GMAUC：Junuthula 进一步改进 AUC，他们认为动态链路预测任务可以分解两个子任务，其一是预测已经出现过的连边的消失/复现，其二是预测未出现

过的连边。第一个任务没有类别不平衡的问题，可以用 AUC 度量；而后者需要用 PRAUC。