1 绪论

1.1 选题背景及意义

社交媒体机器人(Social Media Bots)，简称为社交机器人，是一种运作于社交媒体平台上的机器人，他们作为社交网络中的一种实体，可以自动和别的用户产生互动，发布信息，提出观点。根据世界领先的网络安全公司 Imperva 研究显示，大约有 9~15%的 Twitter 用户是机器人。它们不仅能够假装是某位博主的关注者（粉丝），也能创造一个虚假的账号来吸引关注者。由于它们在社交媒体中的行为和真实人类用户非常相似，所以很难在庞大的社交媒体用户群中被发现。这些机器人渗透于普通人类用户中，针对某些社会热议的话题发表特定的言论。然而并没有特定的法规对社交媒体机器人的活动和发言进行限制，导致这种机器人成为了影响社交网络的一把双刃剑。良好的社交媒体机器人可以即时发送某个特定领域的重要新闻，让关注它的粉丝们得到感兴趣的信息；恶意的社交媒体机器人会散播虚假的信息和负面消极的言论，误导舆论走向，造成社会的恐慌。因此，为了加强社交平台的安全性，研究人员需要对机器人在社交网络上的行为和影响力进行跟踪和预测，了解信息从机器人到人类用户的传播过程（即，用户之间的点赞、评论、转发等事件）。为了预防恶意机器人制造有害信息并利用关注者的转发等行为大肆宣传，还需要对未来信息的可能传播途径进行预测。这种研究机器人用户和人类用户之间在未来是否会发生信息传播事件的任务可以被抽象地称为链接预测(Link Prediction)。它作为网络理论的一个重要分支，拥有广泛的应用前景，因此成为近年来的研究热点之一。

然而，当代互联网的高效性导致了基于计算机网络的社交媒体能够几乎不受地理位置的限制而传递多种类别的信息，社交网络的用户规模和结构复杂度也因此进一步提升。基础的统计学习方法通常只能处理欧几里得空间中的数据，若对于非欧式空间的网络图结构进行链接预测，则需要耗费大量的计算时间；且社交网络的变化日新月异，网络中可能会出现新的用户，用户的爱好和职业等特征会随着时间而改变，也会不断形成新的好友关系（例如，他们可能关注新的好友，取消关注不再感兴趣的好友）和信息传递事件，致使社交网络几乎在每一个时刻都有着不同的形态。由于网络结构高度的动态性，传统的基于静态图数据的机器学习方法无法适用于实时变化的动态社交网络，所以其学习得出的链接预测结果失去了参考价值。而构建一个准确且有效的动态链接预测模型是具有挑战性的，它需要提取非欧几里得空间数据的特征并分析两个节点单元之间的关系，学习网络的拓扑结构和时间变化特征，并根据得到的信息做出符合当前时刻网络特征的链接预测。并且，目前多数的链接预测研究将应用重心放在商品和好友推荐任务上，而少有在信息传播预测上的应用。因此，为了更有效地研究动态社交网络、预防恶意社交机器人，本研究需要寻找能够学习社交网络动态特征和结构特征信息的链接预测方法，以此推理网络中社交机器人和用户节点间的信息传播倾

向和传播时间，如图 1-1 所示。

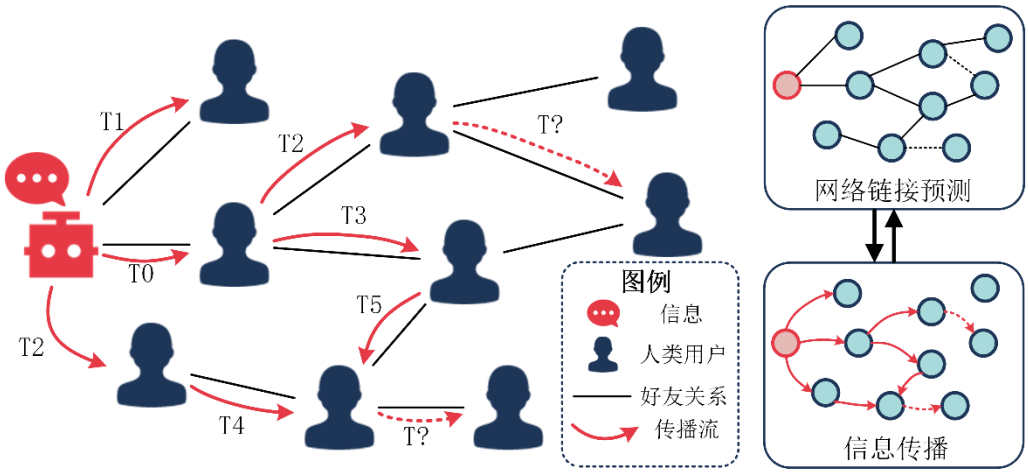


图 1-1 动态网络中社交机器人的信息传播过程研究图示

1.2 研究现状

目前，已有大量链接预测方法被应用于社交网络相关的分析研究。由于深度学习是近年来计算机研究领域的热点，近期涌现的有关链接预测的方法也多与深度学习有关。根据所用原理不同，链接预测模型具体可以被分为两个类别：基于传统数学方法的链接预测模型和基于深度学习方法的链接预测模型。

1.2.1 基于传统数学方法的链接预测模型

传统的数学方法通常从节点相似度、概率论原理判断网络中节点间产生链接的可能性。

节点相似度方法通常认为越相似的节点间越有可能产生链接。Newman et al.[1] 最早提出将网络中两个节点的相似度定义为共同邻居（Common Neighbors，CN），即两个节点邻居的交集。因此两个节点共同的邻居越多，它们之间将越有可能产生联系。在此基础上，Kossinets & Watts[2][3]证明了该方法在大型社交网络中的适用性。虽然在 CN 的基础上后续研究给出了很多类似的算法模型，但它们在庞大的社交网络中的预测表现和效率不如原始的 CN 算法。

基于概率理论的链接预测模型使用条件概率描述节点间发生互动的可能性。Holland et al.[4] 首先提出了随机分块模型（Stochastic Block Model，SBM），将网络中的节点定义到不同的分块（Block）中并赋予不同的特征，两个节点产生链接的概率取决于节点所属的分块。Guimera et al.[5] 在此基础上，对 SBM 模型进行优化，使得模型可以用于预测网络中本该出现但未出现、实际不该存在但存在的链接。SBM 提取了富有价值的网络特征，因此在链接预测任务上比节点相似度方法拥有更高的准确率。

但需要注意，传统数学方法普遍存在的问题是：算法使用的前提是网络中的节点属性和边属性，这些属性信息通常很难从复杂的网络中提取出来。而且，调整模型的参数也将耗费大量的时间。

1.2.2 基于深度学习方法的链接预测模型

相比传统的数学方法，深度学习方法可以有效解决网络图信息提取的问题。深度学习下的链接预测任务通常转化为根据模型训练得出的低维节点嵌入表示向量，预测某个特定时间点下某两个节点间是否会产生链接。Perozzi et al[6]首次将基于自然语言处理(Natural Language Processing , NLP)嵌入表示的 Word2Vec[7]模型转化到图结构上，提出了 Deep Walk 模型，通过随机游走将富有图结构信息节点表示映射到一个低维向量之中。随后 Grover & Leskovec[8]对 Deep Walk 模型进行改进，提出了 Node2Vec 模型，可以学习到准确性更高的能捕捉多种邻居信息的节点表示，然而由于随机游走的随机性，模型的可控性无法保证。

图神经网络（Graph Neural Networks, GNNs）是作用于图域上的一种深度学习方法[9]。由于社交网络中的社会实体可以被抽象地看做节点，而实体间的互动可以被抽象为边，所以图神经网络能够很顺利地用于社交网络链接预测研究。现有的大多数图神经网络的链接预测方法都集中在静态图上。Bruna et al.[10] 最先使用图卷积神经网络(GCN)学习节点的嵌入表示，并由 Kipf & Welling[11] 对其进行改进，将节点嵌入表示成 K-步之内的邻居节点的聚合。在 GCN 的基础上，Veličković et al.[12]提出了图注意力网络(GAT)，根据邻居节点各自的结构特点对目标节点和相邻节点连接的边分配不同的聚合权重。动态图网络链接预测模型多为在静态图表示学习模型的基础上加入时间维度。例如，基于 Deep Walk 模型：Nguyen et al.[13]提出了时序游走策略，令随机游走中的每一步遵循时间的升序排列，因此生成的节点序列中就包含了时间；Wang et al.[14]结合匿名游走策略和因果推理提出了因果匿名游走（CAW）对时序网络图进行归纳式的学习。基于 GAT 模型，Qu et al.[15] 用时间差代替节点的聚合权重计算公式中的指数部分，因此聚合得到的节点表示中也涵盖了时间的信息。基于图神经网络的链接预测模型多用于社交网络中的好友推荐[16][17]，而对社交机器人和信息传播的研究尚处于起始状态。

1.3 主要工作内容

本文将使用基于时序点过程和注意力机制的图神经网络模型[18]，对 Twitter 上的用户间转发、关注的互动数据进行链接预测，并将预测结果应用于机器人信息传播研究。主要的工作内容分为以下三个方面：

1）基于动态图，构建一个用于动态链接预测的图神经网络模型。

2）使用该模型对两种现实动态社交网络进行链接预测，并验证模型的正确性和有效性。

3）对模型的预测结果进行可视化，模拟来自社交媒体机器人的信息在社交网络中的传播过程。

1.4 论文组织结构

本文由五个章节组成。其中：

第一章：绪论。该章节讲述了动态链接预测对社交媒体机器人信息传播监测的重要性，以及图神经网络解决信息传播问题的适用性。

第二章：相关技术理论。该章节列出了本研究所涉及的理论基础。包括时序点过程理论、图注意力网络模型、动态图表示学习模型框架。

第三章：面向信息传播分析的动态预测模型。该部分详细阐述了本研究搭建的面向信息传播分析的动态预测模型。模型实现了节点表示学习和两种链接预测任务。

第四章：实验结果与分析。此章节首先介绍 Social Evolution 和 TwiBot-20 数据集和处理方法，并说明搭建网络模型所需的软硬件环境和模型的参数设置。然后对两个数据集进行实验，对所得结果进行展示和分析。最终可视化模拟来自社交媒体机器人的信息在社交网络中不断扩散的过程。

第五章：结论与展望。此章节总结了本文的全部工作，并对未来实验的可能方向进行了展望。

2 相关技术理论

该章节是本研究涉及理论的简要概述。其中，时序点过程描述社交网络动态性，是本研究的数学基础；图注意力网络捕捉社交网络的空间信息，是本文动态链接预测模型的算法依据；动态网络图的节点表示学习理论结合网络的时间信息和空间信息，是本文进行准确有效的动态链接预测的前提条件。注意：本文后续使用的数学符号中，所有小写字母符号，如e_{ij}，代表标量；粗体的小写字母符号，如$\boldsymbol{h}$，代表向量；粗体的大写字母符号，如$\boldsymbol{A}$，代表矩阵。

2.1 时序点过程

时序点过程(Temporal point process, TPPs)[19]是在连续时间域上建模的经典数学工具。在社交网络模型中，时序点过程可以看做连续时间域上的一系列事件构成的随机过程。时序点过程等价于一个计数过程，可以被形式化为在时刻$\{t_1, t_2, t_3, \dots\}$发生的事件，其中$t_i \le T$，T代表了该过程中的时间跨度。区分不同时序点过程的核心是条件强度函数(conditional intenstiy function)，简称为强度函数，用$\lambda(t)$表示。在给定了历史的事件发生的时刻$t_1, t_2, t_3, \dots, t_n$，以及在时间$t \in (t_n, T]$内没有事件发生的条件下，事件在区间$[t, t+dt]$内发生的可能性为$\lambda(t)dt$。时序点过程的幸存函数(Survival Function)表示在时间段$[t_n, t)$中没有事件发生的概率[20]，计算方式为$S(t) = exp(-\int_{t_n}^{t} \lambda(\tau)d\tau)$。而时序点过程的条件密度函数(conditional density function)$f(t)$定义为幸存函数和条件强度函数的乘积：$f(t) = \lambda(t)S(t)$，表示事件在时间t发生的概率大小。下一个事件发生的时间可以通过计算$f(t)$的期望得到。在本文中所使用的链接预测方法就是基于时序点过程的原理，计算图在每个变化时刻的条件强度函数，进而得出条件密度函数的值，以此确定在某个时间点，既定的两个节点之间的是否会产生连接。通过计算条件密度函数的数学期望来预测既定的两个节点之间什么时候会产生下一次的链接。

2.2 图注意力网络

图注意力网络是对图卷积神经网络进行改进得到的模型。传统的图卷积神经网络本质是学习一个卷积核来输出节点表示。其计算公式为：

$$H = \widetilde{\boldsymbol{D}}^{-1/2}\widetilde{\boldsymbol{A}}\,\widetilde{\boldsymbol{D}}^{-1/2}X\boldsymbol{\theta}\ , \qquad (2\text{-}1)$$

其中，H代表卷积计算的结果，也是节点的隐藏表示向量；$\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_n$表示考虑自环的图邻接矩阵；$\widetilde{\boldsymbol{D}} = \sum_j \widetilde{\boldsymbol{A}}_{i,j} = \boldsymbol{D} + \boldsymbol{I}_N$表示图的度矩阵；$X$为节点的特征向量；$\boldsymbol{\theta}$为卷积核，在模型中作为训练参数随着梯度下降法进行更新。

然而这种卷积模型依赖于特定的图结构，因此当该模型在一个图上训练完毕后，不能通用于其他结构不同的图。然而图注意力网络使用了注意力机制，专注于获取重

要性更大的邻居节点信息，可以处理不同结构的图网络，因此模型的应用范围更加广泛。图注意力网络的输入是节点隐藏表示向量$\vec{z_i}$，输出是带有结构特征的节点隐藏表示向量$\boldsymbol{h}_i$。为了得到更具有表达力的节点表示，模型引入了注意力系数（Attention Coefficient）：

$$e_{ij} = s(\boldsymbol{W}\boldsymbol{z}_i, \boldsymbol{W}\boldsymbol{z}_j), \tag{2-2}$$

其中，$\boldsymbol{W}$是权重矩阵；s是表现自注意力机制的参数。该注意力系数被用于生成源节点i和其邻居节点j的边的权重：

$$q_{ij} = \mathrm{softmax}_j(e_{ij}) = \frac{exp(e_{ij})}{\sum_{k\in\mathcal{N}_i} exp(e_{ik})} \tag{2-3}$$

根据式（2-2），源节点和其邻居间边的权重会根据两个节点间的相关性大小分配相应的权重。源节点的节点隐藏表示向量可以计算为源节点所有邻居节点的加权聚合：

$$\boldsymbol{h}_i = \sigma\left(\sum_{j\in\mathcal{N}_i} q_{ij}\boldsymbol{W}\boldsymbol{h}_j\right) \tag{2-4}$$

该计算方式克服了图中不同节点拥有不同邻居数量的问题，并只选取 1 阶邻居矩阵进行计算，大幅提高了节点表示学习的效率。上述每个图注意力层中的计算过程可以表示为图 2-1。

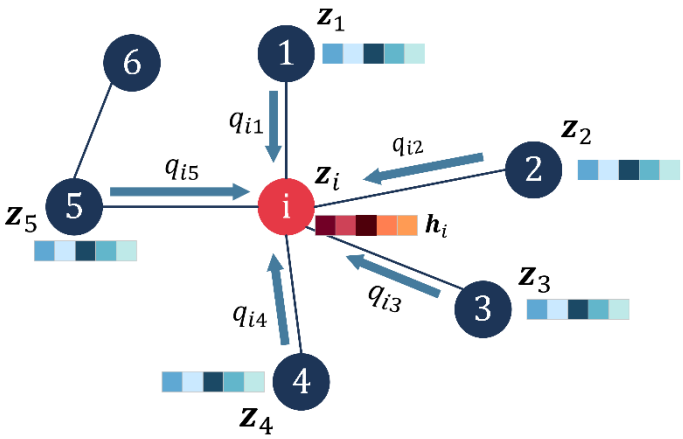


图 2-1 图注意力网络节点表示计算过程

2.3 动态网络图的节点表示学习

动态网络图通常有两种表现形式：离散表示(Discrete Representation)和连续表示(Continuous Representation)。离散表示使用一系列静态图的快照来表示动态图，因此模糊了时间的概念。而本文使用的连续表示是唯一在图中包含了时间信息的图表示法[21]，它最接近现实意义上的动态图结构，但是也由于它精细的时间粒度导致了方法的复杂化[18]。社交网络媒体通常拥有较高的更新速率。根据 Twitter 在 2021 年发表的致股东函显示[22]，Twitter 在 2020 年第四季度已有日均 192,000,000 名活跃用户，每天约有 500,000,000 条新推文被发送到平台上，基本每秒就有 5.787 条推文被发送。因此只有

连续表示的动态网络图才能较好地符合该类社交网络的细粒度特征。动态图的细粒度区别体现在图中节点和边的变化不具有一样的动态特征：在社交网络中，两个用户间“互相关注形成的好友关系”和“点赞评论转发形成的通信关系”在时间维度上是不同的，前者产生的联系是较为持久、稳定的，而后者产生的联系是瞬时的。因此，使用连续表示方法的动态图网络模型需要能够区分两种不同时间规模的交互事件：关联事件和交流事件[23][24]。关联事件代表图的动态性，会导致图的结构发生改变，这种改变是长时间持续的；交流事件代表图中元素的动态性，指代图中两个可能并不相连的节点间发生的临时的信息流动，这类事件不会导致图的拓扑结构发生改变。

从数学的角度，这种图上的动态性可以使用时序点过程进行抽象：图上的动态过程表示为一串事件序列$\mathcal{O} = \{(u, v, t)_p\}_{p=1}^{P}$，每一个元组$e = (u, v, t)$代表在时间$t$下发生的一个事件。为了捕捉每个节点之间的依赖关系，可以把图中的节点作为时序点过程的维度，边代表维度之间的互动。为了体现关联事件和交流事件之间的区别，在每一个事件元组中增加一个维度 k 表示不同动态特征的事件。因此，本研究所使用的深度归纳式节点表示学习模型使用两种时间规模的时序点过程来捕捉连续时间动态网络图上的关联事件和交流事件，并将时序点过程中的条件强度函数作为该模型的参数。节点的表示由受到条件强度函数影响的时序注意力机制对邻居节点进行加权聚合得到。这种计算方式可以将空间结构和时间维度结合在一起，得出具有较强时空表现力的节点表示。

3 面向信息传播分析的动态预测模型

此章详细描述了本研究使用的模型的基本框架、节点表示学习算法和动态链接预测问题的求解过程。展示了面向信息传播分析的动态预测模型提取社交网络时空信息、生成节点表示、训练优化和完成链接预测任务的过程。

3.1 问题定义

本文使用$\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t)$表示在时间点$t$时的社交网络图$\mathcal{G}$，其中$\mathcal{V}_t$是图$\mathcal{G}_t$中边的集合。$\boldsymbol{A}(t) \in \mathbb{R}^{n\times n}$是社交网络图$\mathcal{G}_t$在时间点$t$时刻的邻接矩阵。网络图中的不同节点代表不同的用户。节点之间的边代表对应的两个用户之间在时间t发生的事件，本文用元组$e = (u, v, t, k)$表示，其中u, v是该事件中涉及的两个用户节点，t是事件发生的时间，$k \in \{0,1\}$代表事件的类型。若$k = 0$，则发生的是关联事件；若$k = 1$，则发生的是交流事件。因此，在时间窗口$[0, T]$内发生的P个事件能够被表示为$\mathcal{O} = \{(u, v, t, k)_p\}_{p=1}^{P}$，其中事件之间是按照时间升序排列的，且对每个事件的时间$t_p \in \mathbb{R}^+$，有 $0 \le t_p \le T$。对于社交网络中的用户节点v，由于社交网络图$\mathcal{G}$随着时间不断变化，所以将$\boldsymbol{z}^v(t) \in \mathbb{R}^d$作为事件$e = (u, v, t, k)$发生之后节点$v$的表示，将$\boldsymbol{z}^v(\bar{t})$作为事件$e = (u, v, t, k)$发生前一时刻的节点$v$的表示。

3.2 模型基本架构

模型基于循环神经网络(RNN)，但对其进行了结构上的调整，使其适用于图网络的训练。循环神经网络模型可以保存研究对象的历史状态，因此适用于动态网络的分析研究。本模型的输入和输出并不相互独立：对模型进行事件输入的前提是模型已经处理好前一个时间点发生的事件，并更新相关参数。更具体的，本模型进行用户节点表示学习的过程为：1）事件元组按照发生顺序从先到后依次进行训练，模型在一个时间点只处理一条元组。2）用户节点的表示向量随着事件的发生并使用注意力机制进行实时更新。3）记录图结构信息的矩阵也根据时序点过程的原理随着每一个事件的发生进行更新。4）每一次模型在时间t的更新都基于$\bar{t}$时间及以前的事件，并影响链接预测的计算结果。本实验模型的输入和输出数据间的依赖关系如图 3-1 所示。

图 3-1 中，Ω_n代表了t_n时刻的模型参数；$e(t_n)$代表在t_n时刻发生的事件；$Z(t_n)$表示图中所有用户节点的表示向量；$\lambda(t_n)$和$S(t_n)$表示 $e(t_n)$中两个用户节点间的条件强度值和幸存概率。模型每一次训练的输入除了当前时间t_n发生的事件以外，还有前一次训练结束后的模型参数Ω_{n-1}、最新的用户节点表示向量$Z(t_{n-1})$。当前训练完成后，输出$\lambda(t_n)$和$S(t_n)$，并将改变了的模型参数Ω_n和更新后的$Z(t_n)$送入下一时刻的训练过程。

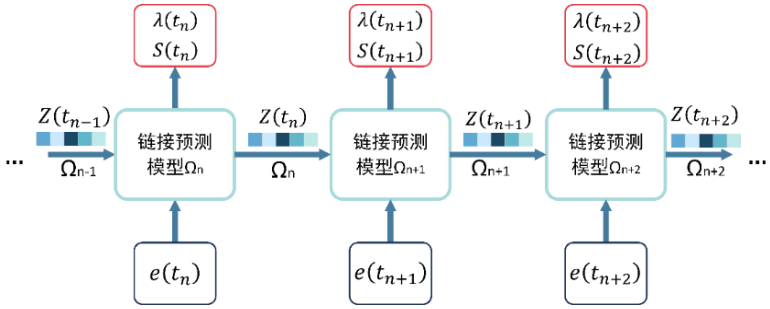


图 3-1 模型的输入输出依赖关系

3.3 动态社交网络图中的用户节点表示

进行可靠的链接预测的前提往往是令模型学习具有高表现力和分辨力的节点表示，具体体现在节点表示中涵盖的时间信息和空间结构信息。本部分将阐述模型计算节点表示的原理和过程，以及相关参数的更新算法。

3.3.1 节点表示的计算方法

由于社交网络的动态性，每一个参与在事件中的用户节点表示也会随着时间发生变化。为了尽可能保存历史信息，节点表示的计算使用了一种循环结构。同时，为了更好地学习不同规模上的信息，计算公式由三部分组成：局部嵌入传导、自传导和外部影响。即，在事件$e_p = (u, v, t_p, k)$发生后，节点v的表示被定义为：

$$\boldsymbol{z}^v(t_p) = \sigma\left(\underbrace{\boldsymbol{W}^{struct}\boldsymbol{h}^u_{struct}(\bar{t_p})}_{\text{局部嵌入传导}} + \underbrace{\boldsymbol{W}^{rec}\boldsymbol{z}^v(\bar{t_p^v})}_{\text{自传导}} + \underbrace{\boldsymbol{W}^t(t_p - \bar{t_p^v})}_{\text{外部影响}} \right), \tag{3-1}$$

其中，$\boldsymbol{z}^v(t_p) \in \mathbb{R}^d$为目标用户节点$v$的表示；$t_p$为当前时间点；σ为激活函数；$\boldsymbol{W}^{struct} \in \mathbb{R}^{d \times d}$为负责聚合局部嵌入的全连接的神经网络；$\boldsymbol{h}^u_{struct} \in \mathbb{R}^d$是聚合了用户节点$u$的邻居（好友）节点信息的隐藏向量；$\bar{t_p}$表示了当前时间点前一刻的时间点；$\boldsymbol{W}^{rec} \in \mathbb{R}^{d \times d}$为负责聚合目标用户节点当前表示的全连接的神经网络；$\bar{t_p^v}$表示了节点v在时间点t_p之前最近一次发生事件的时间；$\boldsymbol{W}^t \in \mathbb{R}^d$为负责聚合外部影响因素的全连接的神经网络。

为了计算式（3-1），需要先求$\boldsymbol{h}^u_{struct}$。该部分的计算基于时序注意力机制，即对于目标节点$u$和它的邻居节点$i$，其对应的注意力权重系数$q_{ui}(t)$可以被定义为：

$$q_{ui}(t) = \frac{exp(\mathcal{S}_{ui}(\bar{t}))}{\sum_{i' \in \mathcal{N}_u(t)} exp(\mathcal{S}_{ui'}(\bar{t}))}, \tag{3-2}$$

其中，$\mathcal{S}_{ui}(\bar{t})$是一个随机的选择矩阵，表示在$\bar{t}$时刻的用户节点$u$和用户节点$i$之间的强度，具体的运算细节将在 3.1.2 节介绍。

则节点u的邻居信息隐藏向量$\boldsymbol{h}^u_{struct}$可以由式（3-3）计算：

$$\boldsymbol{h}^u_{struct}(\bar{t}) = max\left(\{\sigma\left(q_{ui}(t) \cdot \boldsymbol{h}^i(\bar{t})\right), \forall i \in \mathcal{N}_u(\bar{t})\}\right), \tag{3-3}$$

其中，$\boldsymbol{h}^i(\bar{t}) = \boldsymbol{W}^h\boldsymbol{z}^i(\bar{t}) + \boldsymbol{b}^h$表示用户节点$u$在时间$\bar{t}$时的隐藏向量；$\mathcal{N}_u(\bar{t})$表示用户节点$u$在时间$\bar{t}$时的邻居集合；$\boldsymbol{W}^h \in \mathbb{R}^{d\times d}$和$\boldsymbol{b}^h \in \mathbb{R}^d$是主导节点特征向量传播的可训练向量。

因此，节点表示的计算方法可以总结为图 3-2 所示。

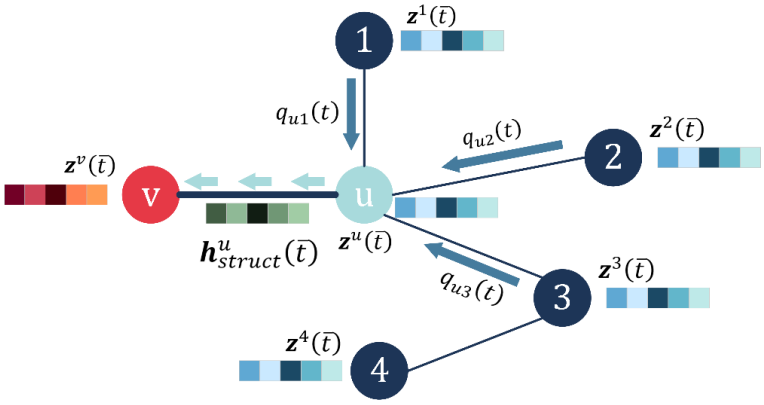


图 3-2 节点表示的计算方法

除此之外需要注意，新的节点可能会加入网络。以下将分别分析当一个事件(u, v, t, k)中存在一个或两个新节点时，节点表示向量和隐藏向量的计算方法：

1）一个事件中，只有一个新节点u：对新点u，$\boldsymbol{h}^u_{struct}$就是新点$u$的向量表示，$\boldsymbol{z}^u$则是随机生成的；对于已存在的节点$v$，$\boldsymbol{h}^v_{struct}$可以直接按照式（3-3）通过已存在的节点$v$的邻居计算得出，$\boldsymbol{z}^v$就是它最新的节点嵌入。

2）一个事件中，两个节点都是新点：对于时间中的每个节点i，$\boldsymbol{h}^i_{struct}$是另一个节点的特征向量，$\boldsymbol{z}^i$是节点自身的特征向量

3.3.2 随机选择矩阵$\mathcal{S}(t)$的更新

由于模型需要保存网络图在过去时间点的特征，需要设定一个储存动态图里复杂时序信息的矩阵。本研究中，将这个矩阵称为随机选择矩阵$\mathcal{S}(t)$。随机选择矩阵$\mathcal{S}_{uv}$(t)捕捉时间t时刻用户u和用户v之间联系的强度，而该强度由两个用户之间事件的发生情况决定，因此$\mathcal{S}_{uv}$(t)应当随着节点u和节点v间发生的事件同步进行更新，并使用条件强度函数$\lambda_k^{u,v}(t)$作为主导$\mathcal{S}_{ui}(t)$更新的因素。对于已观察到的事件$p = (u, v, t, k)$，条件强度函数$\lambda_k^{u,v}(t)$定义一个时序点过程，用以表示在时间t下用户节点u和v之间发生事件p的概率，即：

$$\lambda_k^{u,v}(t) = \psi_k \log\left(1 + \exp\left(\frac{\boldsymbol{\omega}_k^T \cdot [\boldsymbol{z}^u(\bar{t}); \boldsymbol{z}^v(\bar{t})]}{\psi_k}\right)\right), \tag{3-4}$$

其中，ψ_k是一个动态的向量，用于捕捉不同规模的事件演变过程；[;]表示向量的拼接；$\boldsymbol{\omega}_k \in \mathbb{R}^{2d}$是模型中的可学习参数，与$[\boldsymbol{z}^u(\bar{t}); \boldsymbol{z}^v(\bar{t})]$组合在一起以体现用户节点$u$的表示向量$\boldsymbol{z}^u(\bar{t})$和用户节点$v$的表示向量$\boldsymbol{z}^v(\bar{t})$之间在时间规模上的兼容度。

计算得到了强度函数$\lambda_k^{u,v}(t)$之后，随机矩阵的更新规则可以概括为：在初始的时间点$t = t_0$,从初始邻接矩阵$\boldsymbol{A}(t_0)$直接建立$\mathcal{S}(t_0)$。对于一个给定的节点v,初始化随机选择

矩阵的规则为：若$v = u$ 或$\boldsymbol{A}_{vu}(t_0) = 0$，则$\mathcal{S}_{\mathrm{uv}}(t_0) = 0$; 若$\mathcal{N}_v(t_0) = \{u: \mathbf{A}_{vu}(t_0) = 1\}$, 则$\mathcal{S}_{vu}(t_0) = \frac{1}{|\mathcal{N}_v(t_0)|}$。初始化后，$\mathcal{S}(t)$和邻接矩阵$\boldsymbol{A}(t)$随着每一个事件$e = (u, v, t, k)$发生而进行更新。对于邻接矩阵$\mathbf{A}(t)$：只有当发生关联事件时，$\mathbf{A}(t)$才会更新。对于随机选择矩阵$\mathcal{S}(t)$：只有当节点$u$和$v$之间本身存在边且发生了交流事件，或节点$u$和$v$之间发生了关联事件时，$\mathcal{S}(t)$。若节点$u$和$v$之间没有边但发生了交流事件，即当$A_{vu} = 1$ 且$k = 1$时，$\mathcal{S}(t)$不会改变。$\mathcal{S}(t)$更新的具体方法为：1）当节点u和v间存在边且发生了交流事件后，$\mathcal{S}_{vu}(t)$需增加条件强度值$\lambda_k^{u,v}(t)$。2）当节点u和v之间发生关联事件，$\mathcal{S}$更新较复杂，需要体现邻居的更新变化情况。即：先将$\mathcal{S}_{vu}(t)$增加条件强度值$\lambda_k^{u,v}(t)$，而由于关联事件会造成新邻居的产生，原先节点的聚合权重值就会减少，减去$\mathrm{b} - \mathrm{b}'$值，b对应事件发生后的基础注意力(Background Attention)，b'表示发生前的基础注意力。b和b'是基于邻居数量统一分布的。

根据以上计算规则，可以得到随机选择矩阵$\mathcal{S}(t)$的更新算法：

算法 3-1 随机选择矩阵$\mathcal{S}(t)$的更新算法

输入：当前事件$o = (u, v, t, k)$，事件强度$\lambda_k^{u,v}(t)$，前一时刻的邻接矩阵$\boldsymbol{A}(\bar{t})$和前一时刻的随机选择矩阵$\mathcal{S}(\bar{t})$

输出：$\mathcal{S}(t)$

令$\boldsymbol{A}(t) = \boldsymbol{A}(\bar{t})$。
if $k = 0$ **then** $\boldsymbol{A}_{uv}(t) = \boldsymbol{A}_{vu}(t) = 1$
令$\mathcal{S}(t) = \mathcal{S}(\bar{t})$
if $k = 1$ and $\boldsymbol{A}_{uv}(t) = 0$ **return** $\mathcal{S}(\boldsymbol{t})$
for $j \in \{u, v\}$ **do**
  $b = \frac{1}{|\mathcal{N}_j(t)|}$，其中$|\mathcal{N}_j(t)|$是$\mathcal{N}_j(t) = \{i: \boldsymbol{A}_{ij}(t) = 1\}$中的元素个数
  $y \leftarrow \mathcal{S}_j(t)$
  if $k = 1$ and $\boldsymbol{A}_{uv}(t) = 1$ **then**
    $\boldsymbol{y}_i = b + \lambda_k^{ji}(t)$，其中$i$是参与在该事件中的另一个节点
  else if $k = 0$ and $\boldsymbol{A}_{uv}(t) = 0$ **then**
    $b = \frac{1}{|\mathcal{N}_j(\bar{t})|}$，其中$|\mathcal{N}_j(\bar{t})|$是$\mathcal{N}_j(\bar{t}) = \{i: \boldsymbol{A}_{ij}(t) = 0\}$中的元素个数
    $x = b' - b$
    $\boldsymbol{y}_i = b + \lambda_k^{ji}(t)$，其中$i$是参与在该事件中的另一个节点
    $\forall w \neq i \; and \; \boldsymbol{y}_w \neq 0, \; \boldsymbol{y}_w = \boldsymbol{y}_w - x$
  end if
  归一化$\boldsymbol{y}$，$\mathcal{S}_j(t) \leftarrow \boldsymbol{y}$
end for
return $\mathcal{S}(t)$

3.4 模型求解

该部分阐述了模型在训练阶段为了提高准确率使用的损失函数和进行链接预测任

务的计算方法。损失函数将在训练过程中作为模型优化的依据，通过最小化损失函数的值提升后续链接预测的表现。而动态链接预测问题的求解法则将沿用时序点过程中的理论。

3.4.1 损失函数的计算

将负对数似然函数作为损失函数，即对于P个已经观察到的事件，损失函数可以表示为：

$$\mathcal{L} = -\sum_{p=1}^{P} log\left(\lambda_p(t)\right) + \int_0^T \mathbf{\Lambda}(\tau)d\tau, \tag{3-5}$$

其中，$\lambda_p(t) = \lambda_{k_p}^{u_p,v_p}(t)$；$\mathbf{\Lambda}(\tau) = \sum_{u=1}^{n}\sum_{v=1}^{n}\sum_{k\in 0,1}\lambda_k^{u,v}(\tau)$为幸存概率，即事件没有发生的概率。由于幸存概率的积分难以计算，研究中采用蒙特卡洛[25]采样法对损失函数中的积分项结果$\mathcal{L}_{survive} = \int_0^T \mathbf{\Lambda}(\tau)d\tau$进行估计，具体算法如下所示：

算法 3-2 蒙特卡洛采样法计算$\mathcal{L}_{survive}$

输入： 小批量数据子集$e_\text{minibatch} = \{e_q = (u,v,t,k)_q\}_{q=1}^{|\ell|}$，数据子集中的节点列表$I$，样本大小$N$

输出： $\mathcal{L}_{survive}$

令$\mathcal{L}_{survive} = 0$
for $q = 0$ **to** $\ell - 1$ **do:**
  $t_{curr} = e_q \to t;\ u_{curr} = e_q \to u;\ v_{curr} = e_q \to v\ ;\ u_{survive} = 0; v_{survive} = 0$
  for N 个幸存样本 **do:**
    从节点列表I中随机选择一个节点u_{other}和节点v_{other}，且$u_{other} \notin \{u_{curr}, v_{curr}\}$, $v_{other} \notin \{u_{curr}, v_{curr}\}$
    for $k \in \{0,1\}$ **do:**
      $u_{survive} \mathrel{+}= \lambda_k^{u_{curr},v_{other}}(t_{curr})$
      $v_{survive} \mathrel{+}= \lambda_k^{u_{other},v_{curr}}(t_{curr})$
    end for
  end for
  $\mathcal{L}_{survive} \mathrel{+}= (u_{survive} + v_{survive})/N$
end for

其中$u_{survive}$和$v_{survive}$的计算由抽样得到的负样本(non events)上的条件强度函数求和得到。即选取$2 \times N$个不包含当前节点u_{curr}和v_{curr}的其他节点，组成实际上不存在的边，计算这些“假边”的条件强度函数，每一组抽样的方法如图 3-3 所示。

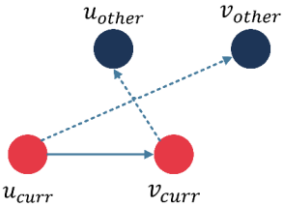


图 3-3 负样本选取

为了提升模型训练的效率，训练过程采用批处理技术，将训练数据分成多组，最终以组为单位统一进行时间反向传播训练，该方法被称为全局时间反向传播。完整的训练方法如下所示：

算法 3-3 全局时间反向传播算法

输入：全局事件序列$\mathcal{O}$，步长l，停止条件$iter_{max}$，当前数据索引$index_{cur}=0$，开始时间$t_{begin}=0$。

for $iter=0$ **to** $iter_{max}$ **do**
  if $index_{cur}>0$ **then** $t_{begin}=\mathcal{O}[index_{cur}-1]\rightarrow t$
  $e\_minibatch=\mathcal{O}[index_{cur}:index_{cur}+l]$
  针对$e_minibatch$建立一个训练网络。将l个连续时间点作为网络的输入。
  计算l步上总损失$\mathcal{L}=-\sum_{p=1}^{l}log\left(\lambda_p(t)\right)+\int_0^T\mathbf{\Lambda}(\tau)d\tau$
  反向传播l步内的误差并更新所有的权重
  if $index_{cur}+l>\mathcal{O}$ **then** $index_{cur}=0$
  else $index_{cur}=index_{cur}+l$
  end if
end for

3.4.2 动态链接预测

为了观察社交媒体机器人对社交平台产生的影响，本文从两种观察角度开展研究：

1）预测机器人在某个时间点是否会和其他用户产生交互事件（关联事件或交流事件）。该问题可以用于估计机器人可能会影响到的用户群体范围。

2）预测在什么时刻，机器人会和特定的用户产生交互事件（关联事件或交流事件）。该问题可以用于模拟信息从机器人向各个用户传播的过程，便于更精确地分析用户在受到机器人影响时的时间信息。

第一种预测任务被称为以时间为条件的预测(Time-Conditioned Prediction)[21]，第二种预测任务被称为事件时间的预测(Event Time Prediction)[18]。两者都属于动态链接预测的范畴，但是所使用的预测方法有细微的差别。

对于以时间为条件的预测，多数研究使用边聚合函数将两个节点合并成一个边向量[9][26][27]，再使用全连接神经网络对该向量进行训练，令训练的结果作为在给定时间下这两个节点之间产生边的概率，并令值趋近于 1。鉴于本文使用时序点过程作为事件的分析方法，两个节点间的条件强度函数可以用于计算在t时刻链接发生的可能性大小。因此，本文以时间为条件的预测可以不直接使用节点的表示向量。其预测过程为：在t时刻，给定一个节点u，计算该节点和图中其他所有的节点产生联系的概率大小，并对这些节点按照计算得出的概率进行从大到小的排序，排名位于前十的节点便是会和节点u产生关联的节点。对于节点u和节点v，其条件密度可以由下式计算得出：

$$f_k^{u,v}(t)=\lambda_k^{u,v}(t)exp\left(-\int_{\bar{t}}^{t}\lambda(s)ds\right), \tag{3-6}$$

其中，$\bar{t}$是节点u和节点v前一次发生事件的时间。

对于事件时间的预测，可以沿用式（3-6）得到的条件强度函数，计算未来时间中最有可能产生链接的时刻，公式为：

$$\hat{t} = \int_t^{\infty} t\, f_k^{u,v}(t)dt, \qquad (3\text{-}7)$$

由于式中的积分项较难计算，可以使用蒙特卡洛抽样法对其进行估计。计算的结果就代表了节点u和节点v之间下一次产生事件联系的时间点。

3.5 本章小结

本章对面向信息传播分析的动态预测模型的运作原理进行了详细的介绍。首先，本模型的基本架构是循环神经网络，模型在处理当前时刻的事件所需的输入依赖于模型处理历史事件时的输出。然后，为了学习包含时间信息和空间信息的节点表示，模型将局部嵌入传导信息、自传导信息和外部影响信息进行聚合，生成具有高表现力的节点表示向量。为了优化模型参数，在训练阶段使用基于时序点过程条件强度值的损失函数。在测试阶段，同样借助时序点过程的条件强度函数进行链接预测任务。

4 实验结果与分析

本章节将使用前文构建的动态预测模型，对带有时间属性的动态网络数据进行实验分析，以完成以下任务：1）验证该模型在动态图网络结构上进行预测的有效性。2）对存在社交媒体机器人的社交网络进行动态链接预测。3）根据链接预测的结果进行信息传播可视化分析。下文将首先介绍实验所需的数据集和环境等基本设置，然后对动态链接预测模型在社交网络中的实验结果进行展示和分析。

4.1 数据集

本研究使用两种数据集对模型进行实验：Social Evolution[28]和 TwiBot-20 数据集①。

Social Evolution 是由麻省理工学院通过跟踪一个宿舍楼内本科生的日常生活收集到的公开数据集。收集的信息包括他们的在 2008 年 1 月到 2009 年 6 月的住址、年级、好友圈和手机电话、短信、蓝牙上的通信信号。好友信息由问卷调查的形式进行收集，记录了存在好友关系的两个人的 id 号和好友关系开始的时间点，因此可以作为动态网络中的关联事件。通信信号由于带有信号的发起人和接收人以及信号产生的时间，所以可以作为动态网络中的交流事件。为了构建初始的图结构，本研究在该数据集中选取了 2008 年 1 月至 2008 年 9 月 10 日间发生的关联事件作为模型训练开始之前图中存在的边。2008 年 9 月 11 日至 2009 年 4 月 30 日期间发生的所有事件作为训练数据，2009 年 5 月 1 日至 2009 年 6 月 30 日期间发生的所有事件作为测试数据。并对测试数据以十天为单位划分成 6 个测试时间段。有关该数据集更详细的统计数据如表 4-1 所示。

TwiBot-20 是本实验分析的重心。它是由西安交通大学冯尚彬等人从 Twitter 上爬取的包含多种综合信息的数据集，包含了几千位用户的个人信息、关注列表（Following）、粉丝列表（Follower）和该用户在 2020 年 7 月至 8 月间发送的推文正文信息。该数据集中的用户包括社交机器人和真实的人类用户，真实地采样了当代社交媒体构成的社交网络，符合了本研究针对社交媒体机器人在社交网络中的行为和影响力进行检测的目标。所有的用户根据其兴趣爱好被分为四个聚类：政治（Politics）、商业（Business）、娱乐（Entertainment）和体育（Sports）。由于 Twitter 的社交平台系统设置，用户 A 在推文中手动“@”用户 B 或者“转发”用户 B 的推文都会对用户 B 发送信息提示，导致信息传播，本文将其统称为用户 A 对用户 B 的“提到”行为。因此，和 Social Evolution 相似，每一个关联事件表示目标用户关注了另一个用户（成为了另一个用户的粉丝），每一个交流事件中的两个节点由目标用户和其推文中“提到”的用户组成。本文在数据集中选取 2020 年 6 月 20 日至 2008 年 6 月 30 日间发生的关联事件作为模型训练开始

① https://github.com/GabrielHam/TwiBot-20

之前图中存在的边。2020 年 7 月 1 日至 2020 年 8 月 20 日期间发生的所有事件作为训练数据，2020 年 8 月 21 日至 2020 年 8 月 31 日期间发生的所有事件作为测试数据。并对测试数据以两天为单位划分成 6 个测试时间段。有关该数据集更详细的统计数据如表 4-1 所示。

表 4-1 Social Evolution 和 TwiBot-20 的统计数据

数据集	聚类名	节点数量	初始关联事件数量	训练阶段事件数量	测试阶段事件数量	聚类系数
Social Evolution	无	83	376	55948	11440	0.548
TwiBot-20	Politics	2723	1396	53687	11939	0.137
	Business	3016	1433	55073	11338	0.134
	Entertainment	3095	1467	56835	11947	0.141
	Sports	3295	1325	69203	15039	0.148

处理数据集时，值得注意的是：1）Social Evolution 中的关联事件（$k=0$）由带有 CloseFriend 标签的数据组成；交流事件（$k=1$）由带有 Calls、SMS 和 Proximity 标签的数据组成。但由于 Proximity 数据中存在大量的噪声数据影响实验结果的准确性，本研究通过筛选去除了事件出现频率低于 0.8 的事件。2）由于 TwiBot-20 提供的数据中包含了大量的原始推文文本，因此需要对该数据集进行信息提取，令最终形成的事件序列$\mathcal{O}$中的每一个事件由格式为(u,v,t,k)的四元组组成。数据集中的 Profile 文件包含了用户的字符串昵称和用户 id，可以构建一个以字符串昵称为键、用户 id 号为键值的字典。Tweet 文件包含了用户 id 和两个月的推文文本，将文本中“提到”的用户名提取后放入字典中查询，可以得出与当前用户产生互动的用户 id，组成的交流事件格式为 $e=(u,\ v,\ t,\ 1)$。Neighbor 文件包含了用户 id 和关注列表，从中提取的关联事件格式为$e=(u,\ v,\ t,\ 0)$。数据的处理示意图如图 4-1 所示。

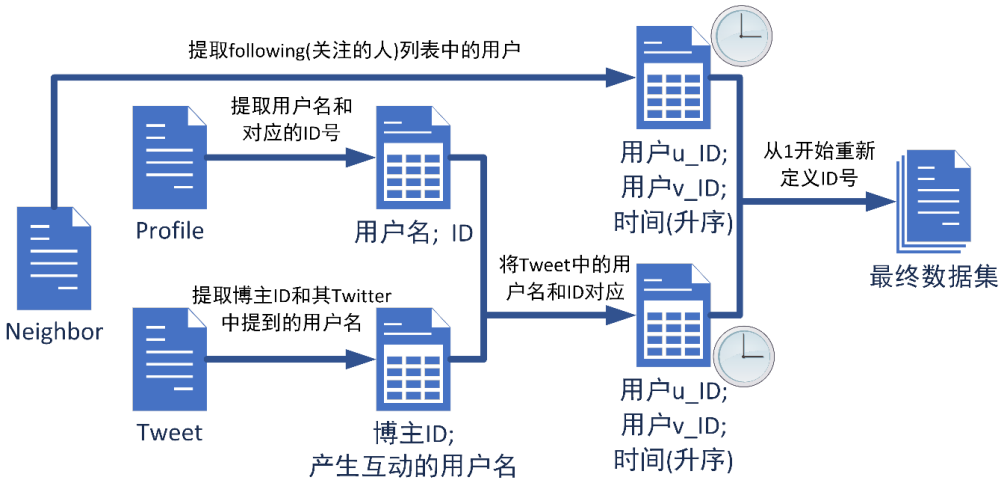


图 4-1 TwiBot-20 数据处理方法

4.2 实验设置

4.2.1 实验硬件和环境

本研究全部的实验是在 Ubuntu 16.04.6 LTS 操作系统的服务器上运行并得出结果的。该服务器的基本配置为：Intel(R) Core(TM) i9-7900X 3.30GHz 的 CPU，Nvidia Titan Xp 12GB 的 GPU 和 125.5GB 的内存。

动态链接预测模型的搭建过程为：以 Python 作为编程语言，使用 pytorch 深度学习框架对模型的基本架构进行实现。为了令实验的过程和结果更具有可理解性，本实验选取了一些重要的模型特征和实验测试结果进行了可视化。使用的可视化库工具包括 Matplotlib 和 Plotly。其中 Matplotlib 工具主要用于可视化以下几个过程：训练阶段的损失函数的下降过程；测试阶段的链接预测结果误差值和精确度的变化过程；在 Social Evolution 数据集上进行实验时，节点信息的学习过程。而 Plotly 工具主要用于动态可视化在 TwiBot-20 数据集上进行实验的测试阶段，节点间信息的动态交互过程。

4.2.2 实验细节

对于节点的初始特征向量，本研究在 Social Evolution 数据集上将用户的年级类型和宿舍号信息作为可提取的特征信息；在 TwiBot-20 数据集上随机生成初始特征信息。

在训练阶段使用 Adam 优化器作为模型的优化方法，并设置学习率=0.0002，权重衰减=0.0005。为了计算幸存函数的蒙特卡洛估计值，设置抽样的时间样本数量为 200。

模型中针对两个数据集所设定的超参数说明如下：

1）对 Social Evolution：节点数量=100，动态过程种类=2，节点特征维度=32，隐藏单元维度=32，批处理大小（步长）=200，幸存样本数=5，梯度阈值=100。

2）对 TwiBot-20：节点数量={2723，3016，3095，3295}，动态过程种类=2，节点特征维度=128，隐藏单元维度=128，批处理大小（步长）=300，幸存样本数=5，梯度阈值=100。

4.3 评价指标

为了验证本研究所使用的链接预测的有效性，使用平均排序值（Mean Average Ranking，MAR）和排序前十命中率 HITS@10 两项标准作为时间为条件的链接预测任务的结果评估指标。使用平均绝对误差（Mean Absolute Error，MAE）和时间预测命中率 TIME@10 的作为事件时间预测任务的结果评价指标。并可视化模型对节点表示的学习和聚类过程。

关于上述的四项指标的计算过程说明如下（令每一个批处理区间内事件的数量为N）：

1）MAR：针对每一个批处理区间，生成涉及到的节点列表。针对第i个测试事件$e_i = (u, v, t, k)$，不仅需要计算用户节点u和真实将会产生链接的用户节点v在时间t时刻发生联系的概率P_{truth}（即，条件密度$f_k^{u,v}(t)$），还需要计算用户节点u和节点列表里所

有其他的节点发生联系的概率大小。然后将这些概率从高到低排序，找出P_{truth}的名次R_i。对每一个批处理中的事件的名次进行求和后取平均便得到了 MAR。

$$MAR = \frac{\sum e_i}{N} \tag{4-1}$$

2）HITS@10：在 MAR 计算的基础上，统计名次R_i进入前十的事件数量，并将其除以总共在批处理中的事件数量，得到 HITS@10。

$$HITS@10 = \frac{名次R_i进入前十的事件数量}{N} \tag{4-2}$$

3）MAE：针对批处理区间内每一个事件$e = (u, v, t, k)$，计算得出其下一次最有可能产生链接的时间$\hat{t}$，并计算该预测的时间和真实情况中下一次链接发生的时间点t_{truth}的误差大小并求平均。

$$MAE = \frac{|t_{truth} - \hat{t}|}{N} \tag{4-3}$$

4）TIME@10：由表 4-1 可以发现，TwiBot-20 数据集中的用户节点数量较多，而用户间的事件数量密度较稀疏，聚类系数相比 Social Evolution 数据集也较低。因此在该社交网络中进行学习的难度较大，对链接预测的结果也需要更多的衡量标准以了解链接预测的效果好坏。本研究提出了类似于 HITS@10 准确性标准的 TIME@10。它在 MAE 计算的基础上，统计$|t_{truth} - \hat{t}| < 10$的事件数量，并将其除以总共在批处理中的事件数量，得到 TIME@10。计算方式如式（4-4）所示。

$$TIME@10 = \frac{|t_{truth} - \hat{t}|<10 \text{ 的事件数量}}{N} \tag{4-4}$$

4.4 实验结果及分析处理

本部分将展示模型在 Social Evolution 和 TwiBot-20 数据集上进行实验的结果。本研究使用的模型基于 Trivedi 等人提出的节点表示学习模型——DyRep。该模型同样在 Social Evolution 上进行实验，但该文献中并没有提供模型开源代码，因此本实验首先改进了 DyRep 模型。为了证明本人构建的模型具有正确性和有效性，模型在展示 Social Evolution 数据集上的测试结果的同时，也将其和原论文模型性能进行对比。在此基础上，TwiBot-20 数据集的实验结果将被用于模拟社交网络中的信息传播预测。

4.4.1 Social Evolution 数据集实验

实验对训练阶段的损失函数（Loss Function）值进行了跟踪。由于使用了批处理技术，本研究记录了模型在处理完每一个批数据之后的损失函数结果。在经过了 280 个批处理迭代过程后，损失函数值的变化趋势如图 4-2 所示。

通过分析图中结果可以发现，随着迭代次数的增加，损失函数总体呈下降趋势，证明了该神经网络模型训练的有效性。图中有四处出现了损失值异常的偏大情况但随后又恢复正常。经分析可能的原因为训练数据没有进行完全的打散（Shuffle）操作。由于

数据需要依照时间顺序排列，因此数据集中可能存在某些用户节点对在某个特定的时间段中发生了大量集中的联系。模型在接收这些类似数据进行学习时，容易造成过拟合，最终导致在处理其他正常分布的数据时损失函数的暴增。

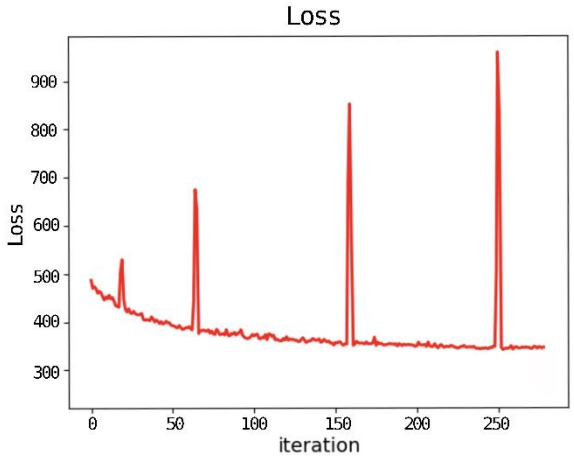
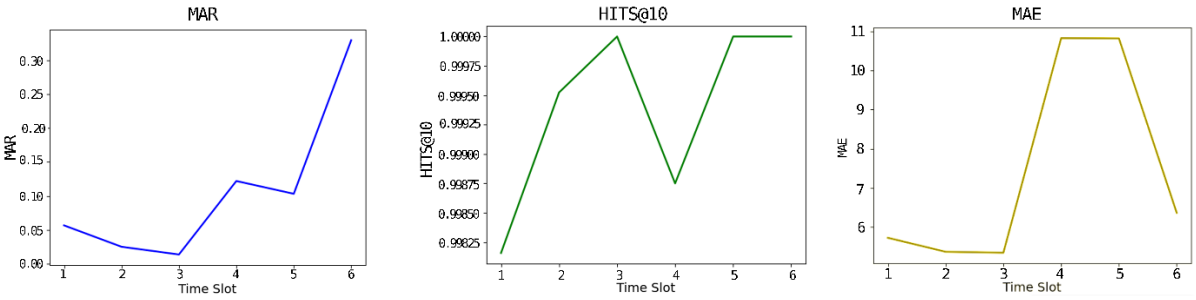


图 4-2 Social Evolution 数据集上链接预测实验损失函数变化趋势

在实验的测试阶段，测试样本被分为 6 个时间段（Time Slot），每个时间段包含 10 天的时间跨度。对于每一个时间段计算模型测试的 MAR、HITS@10 和 MAE 结果。总结如图 4-3 所示。



（a）MAR　　（b）HITS@10　　（c）MAE

图 4-3 Social Evolution 数据集上的链接预测实验测试结果

图 4-3 中的（a）（b）图展示了以时间为条件的链接预测实验结果，通过分析可以发现：大多数预测样本的 MAR 值位于 0.3 以下，证明了对于多数预测样本，源用户节点u和事实上与其产生事件联系的用户节点v_{truth}之间的条件密度函数值比所有节点u其他节点的条件密度函数值大；而观察到每一个时间段 HITS@10 的数值都大于 0.99，可以推断几乎每一个的v_{truth}的条件密度值在测试样本给定的时间条件下都排在前十的位置。由此可以得出结论，模型计算得出的源节点u最有可能产生事件的节点$v_{predict}$和真实产生事件联系的节点v_{truth}在多数情况下是重合的。

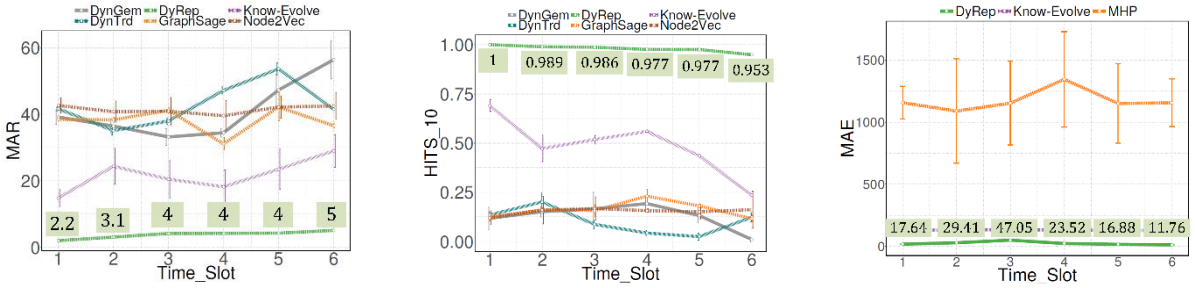
图 4-3 中的（c）图展示了时间预测的实验结果，通过观察发现，在前三个测试时间段内，模型的预测误差较小，而后三个时间段的模型预测误差较大。鉴于此，实验对六个时间段中的事件数量进行统计分析。表 4-2 显示了测试阶段的数据分布情况。

表 4-2 Social Evolution 数据集在测试阶段的分布情况

时间段（Time Slot）	时间范围	事件数量
Slot 1	2009 年 5 月 1 日-2009 年 5 月 10 日	3713
Slot 2	2009 年 5 月 11 日-2009 年 5 月 20 日	4087
Slot 3	2009 年 5 月 21 日-2009 年 5 月 31 日	2633
Slot 4	2009 年 6 月 1 日-2009 年 6 月 10 日	669
Slot 5	2009 年 6 月 11 日-2009 年 6 月 20 日	247
Slot 6	2009 年 6 月 21 日-2009 年 6 月 30 日	91

分布结果可以发现，在 Slot 4~Slot 6 范围中，事件数量明显降低。表明事件的分布变得稀疏。因此可以推断，该链接预测模型更加适用于事件发生较为密集的社交网络。而对于事件发生较为稀疏的网络，会由于数据量的缺失导致预测的精度也随之下降。

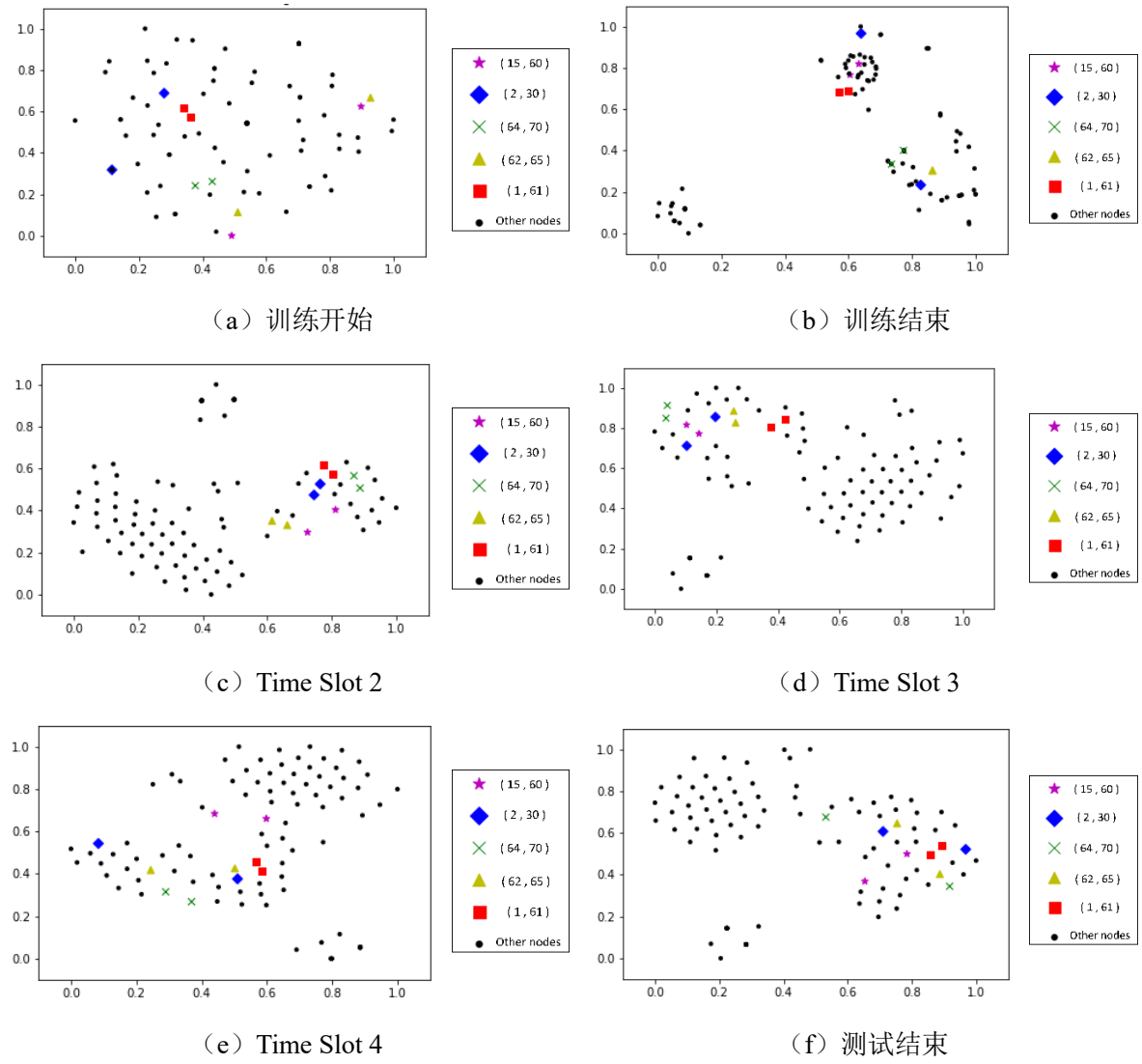
为了更好地体现本研究搭建的模型的有效性和准确性，图 4-4 展示了 DyRep 模型原论文[18]中对 Social Evolution 数据集的链接预测结果的 MAR、HITS@10、MAE 数值。由图中可以发现，绿色的折线所代表的 DyRep 模型预测效果明显高于其他的几类链接预测模型。而经过比对可以发现，本研究经过改进后的模型的链接预测结果相较于原文展示的结果有更高的精度。



（a）MAR　　（b）HITS@10　　（c）MAE

图 4-4 模型原文的实验测试结果

由于基于时序点过程和图注意力机制的神经网络模型可以同时捕捉时间和空间上的信息，因此该模型不仅在链接预测方面有较好的表现，而且有更强的节点表示学习能力。该能力体现在训练和测试阶段，节点表示向量能够随着每一个事件的发生而不断更新。本研究使用 t-SNE（t-分布式随机近邻嵌入）技术对高维的节点表示向量进行维度的削减，映射到低维度的空间中。Python 第三方机器学习库 sklearn 提供了现成的 TSNE 函数。本实验对该函数设置的参数值为：perplexity=30, n_components=2, init='pca', n_iter=3000。其中，将初始化方法设定为“pca”可以令降维的过程更加稳定。实验过程中所有用户节点表示向量的变化过程如图 4-5 所示。可以发现，随着事件序列的发生，节点表示向量从初始杂乱无章的分布逐渐发展成可以观察出聚类的分布。此外，实验跟踪观察了 5 对交互较多的节点，并分别给予它们不同的图标以便区分。如图 4-5 所示，每一个节点对中两个节点之间的距离也并不固定。



（a）训练开始　　（b）训练结束

（c）Time Slot 2　　（d）Time Slot 3

（e）Time Slot 4　　（f）测试结束

图 4-5 节点表示学习过程

为了更直观地体现节点间发生的事件节点表示所造成的影响，我们计算了目标节点间的余弦距离（Cosine Distance）。余弦距离的大小代表了两个向量之间的相似度，余弦距离越小，则向量之间的相似度越大。令节点$\boldsymbol{u}=(u_1,u_2,\dots,u_n)$，节点$\boldsymbol{v}=(v_1,v_2,\dots,v_n)$，余弦距离$D(u,v)$的计算方法如下：

$$D(u,v)=1-\frac{\sum_{i=1}^{n}u_i\times v_i}{\sqrt{\sum_{k=1}^{n}u_{1k}^2}\sqrt{\sum_{k=1}^{n}v_{1k}^2}} \tag{4-5}$$

根据式（4-5），表 4-3 统计了从训练开始至测试结束的 7 个时刻下，5 对节点间的余弦距离。结合数据集的实际事件发生情况，对表 4-3 的数据分析如下：

1）在训练阶段，节点 62 和节点 65 之间产生的事件多达 2067 个。而节点 30 和节点 2 之间产生的事件只有 142 个。因此，$D(15,60)$在训练结束后相比其他节点对具有更小的节点间距离。而$D(30,2)$相比其他的节点对的余弦距离更大。

2）在测试阶段 Slot 2 中，由于节点对(70,64)和节点对(62,65)之间的事件较多，余

弦距离$D(70,64)$和$D(62,65)$相比其他节点对的余弦距离较小。

3）在测试阶段 Slot 3 中，由于节点对$(2,30)$之间没有事件发生，因此在 Slot 3 的数据中$D(2,30)$明显相较其他的余弦距离偏大。

4）而在测试阶段的 Slot 4 之后，表格中所统计的节点对之间发生的事件数量都较少。因此他们的余弦距离也偏大。

表 4-3 节点间余弦距离随着事件序列的变化过程

节点对	训练开始	训练结束	Slot 1	Slot 2	Slot 3	Slot 4	测试结束
D(1,61)	0	0.00012	0.00032	0.00037	0.00023	0.00041	0.00019
D(62,65)	0	1.4305e-06	0.00032	5.900e-06	5.900e-06	0.05341	0.06176
D(70,64)	0	0.00322	0.04208	2.568e-05	2.384e-05	0.01659	0.09481
D(30,2)	1.0	0.03735	0.01392	5.960e-07	0.04074	0.06937	0.04684
D(15,60)	1.0	4.5895e-06	0.05141	0.01889	8.940e-07	0.01119	0.03411

综合上述实验结果，得出的结论为：本研究使用的神经网络模型不仅可以有效地学习时序上的信息，并将其用于动态链接预测任务，得出具有高准确率的预测结果。该模型同样也能捕捉节点独特的并且可能瞬时变化的结构属性，学习出具有高区分度的节点表示。

4.4.2 TwiBot-20 数据集实验结果

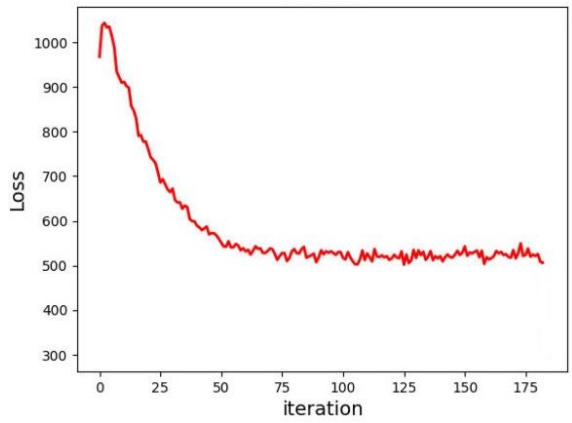
TwiBot-20 数据集的规模较庞大、数据较稀疏，若对整个数据集进行链接预测将导致结果过于繁杂，因此本实验将对数据集中的四个聚类（Business、Entertainment、Politics 和 Sports）分别进行实验，以此缩短模型进行预测任务的耗时，且便于对比分析实验结果。

在训练阶段，模型跟踪了四个类别中的损失函数随着迭代次数增加的变化过程。分析发现模型在 TwiBot-20 数据集中训练时损失函数的下降速度更快且更为稳定，没有出现损失值的暴增或骤减。说明模型中的参数较为符合 TwiBot-20 数据集的分布特征。对比每一个聚类的结果能够发现，从训练开始至训练结束，损失函数值降幅都达到 500 左右。此结果相比模型在 Social Evolution 数据集上训练时损失函数 150 左右的降幅有了成倍的提升。详细的损失函数下降过程如图 4-6 所示。

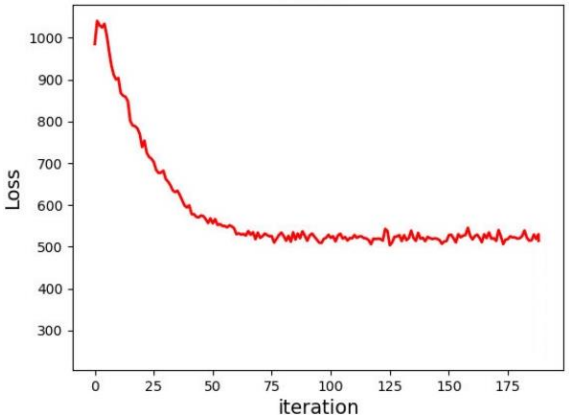
测试阶段，测试样本同样被分为 6 个时间段，但此时每个时间段仅包含 2 天的时间跨度。同样对每一个时间段统计 MAR、HITS@10 和 MAE 的结果。实验中四个聚类完整的测试结果如图 4-7 所示。

根据图 4-7 所示的结果，发现模型的准确率虽然不及 Social Evolution 的准确率效果好，但是与原始模型在相似聚类系数的 Github 数据集上的表现[18]相比，本实验的模型能够达到预期的预测效果。并且随着测试次数的增加，误差指标 MAR 和 MAE 指标总体下降，且准确率指标 HITS@10 和 TIME@10 指标总体上升。因此该实验能够反映

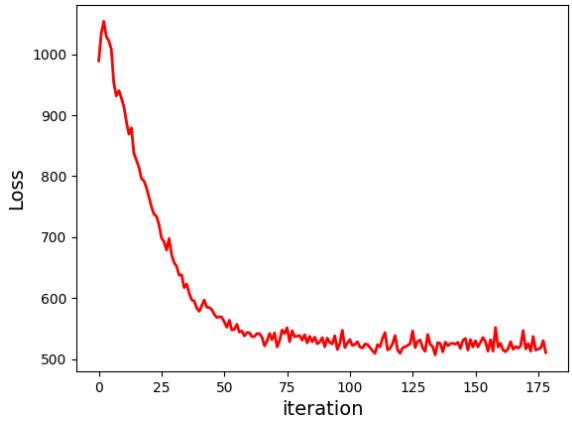
模型在测试过程中不断迭代更新和学习。



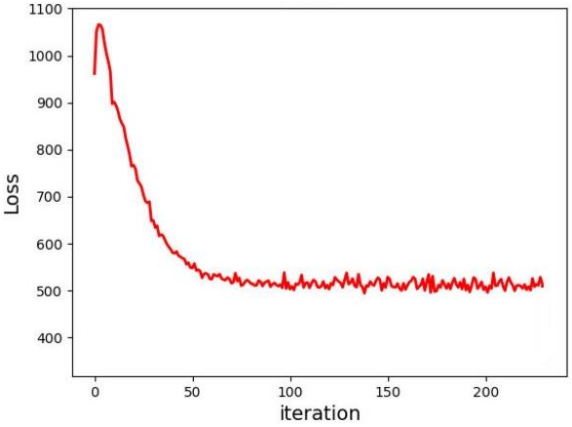
（a）Business



（b）Entertainment

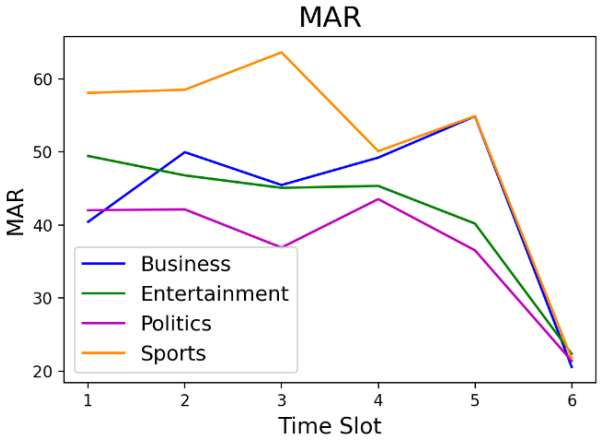


（c）Politics

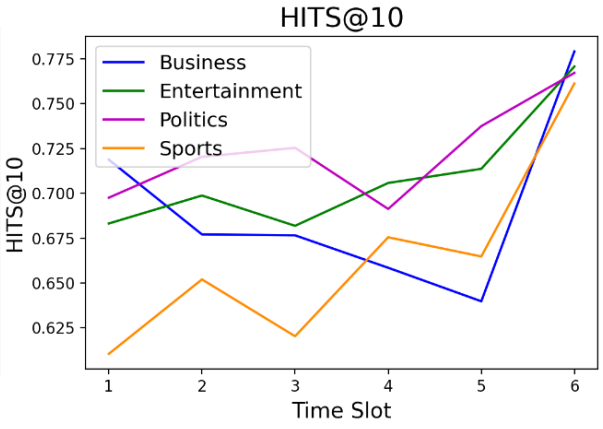


（d）Sports

图 4-6 TwiBot-20 数据集上链接预测实验损失函数变化趋势

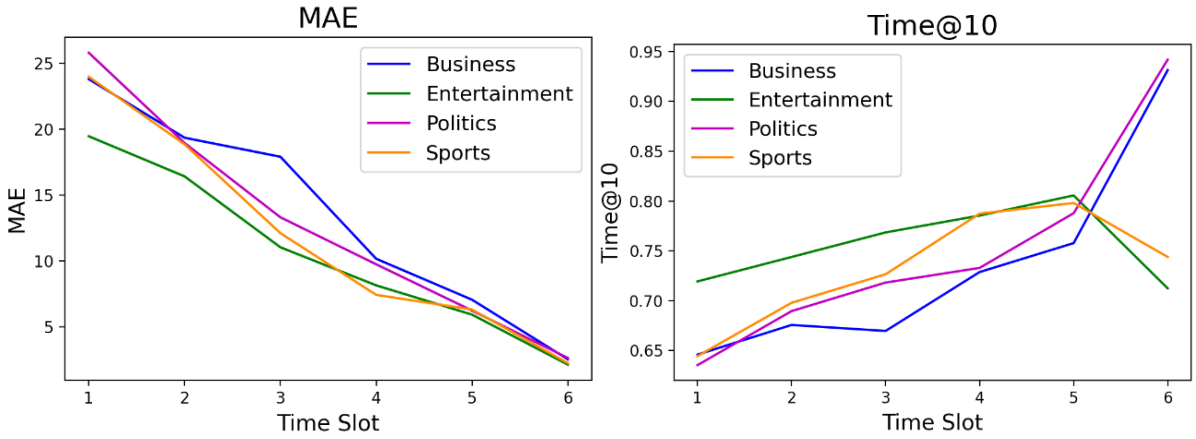


（a）MAR



（b）HITS@10

图 4-7 TwiBot-20 数据集 Sports 聚类的链接预测实验测试结果



（c）MAE　　（d）TIME@10

图 4-7 （续）

4.4.3 基于 TwiBot-20 的社交网络信息传播可视化

上述实验展示了模型在 TwiBot-20 数据集上将链接预测应用于分析在某个特定时间点下哪些用户之间会形成联系，以及两个用户节点之间在什么时候会产生联系。然而社交媒体机器人的跟踪检测不仅需要预测他在某个将来时间点可能产生的联系的用户，也需要从较为宏观的角度观察用户在一段连续时间内和各种用户的信息交流过程。基于这个需求，本研究将预测时间的链接预测任务进行了扩展，对模型的测试结果进行了进一步处理，最终模拟出在一个时间段内，一条信息从一个原始用户（可能是机器人或真实人类用户）传播到另一个用户，最后逐渐扩散到一个用户群的过程。

实验集成开发环境为基于浏览器的 web 应用——Jupyter Notebook，进行动态可视化的工具为 Plotly 图形库。该图形库为生成的图片提供了一个可交互界面，实验者可以放大观察图中的细节，例如节点的邻居、节点间的连线；也可以通过鼠标悬停在图中某个对象来获得位置和属性信息，例如节点的坐标、身份。图 4-8 展示了使用 Plotly 制作的 Business 聚类下的完整网络图，以及使用放大功能和鼠标悬停可以观察到的节点信息。

根据本研究对社交网络图的定义可以发现，只有连通图才能证明信息在所有节点间已互通，因此为了更清晰地展示一条信息的传播过程，本实验在模型测试时统计整个网络的连通图数量。每一个连通图都代表了某种信息在用户间的传播。图 4-9 展示了 Politics 聚类中某个信息的动态传播过程。图 4-10 展示了 Business 聚类中某个信息的动态传播过程。每两帧画面的间隔为 1 个小时。进度条每一个单元也代表 1 小时。

图 4-9 和图 4-10 中的红色节点表示 Twitter 上的机器人用户，而蓝黑色节点代表真实的人类用户。被绿色的空心三角标记的节点表示每一个交互事件中发出信息的那个节点。黑色节点外的绿色的空心三角代表了该节点曾作为信息传递者，帮助机器人将信息传播给更远的用户。

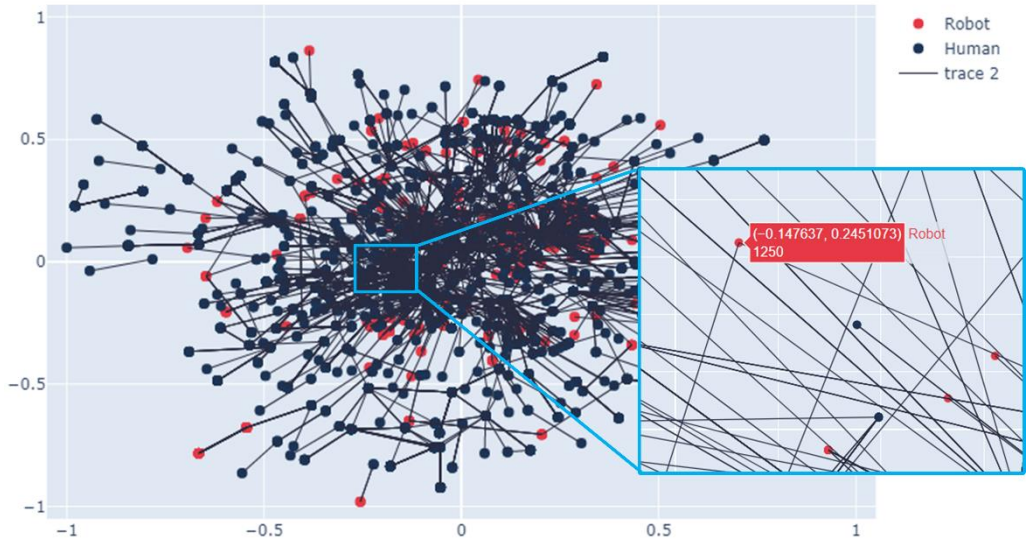
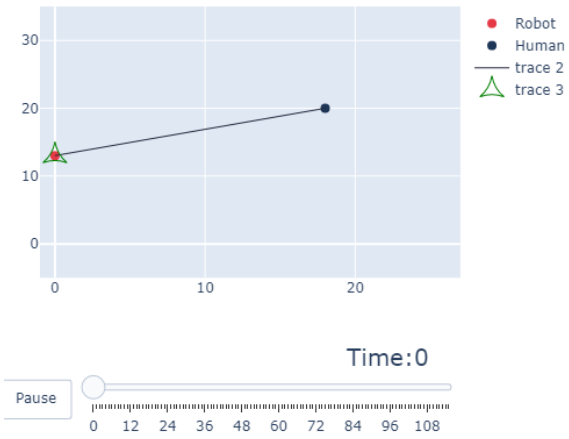
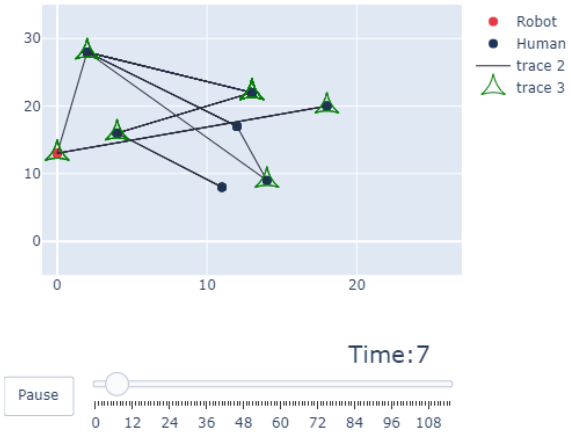


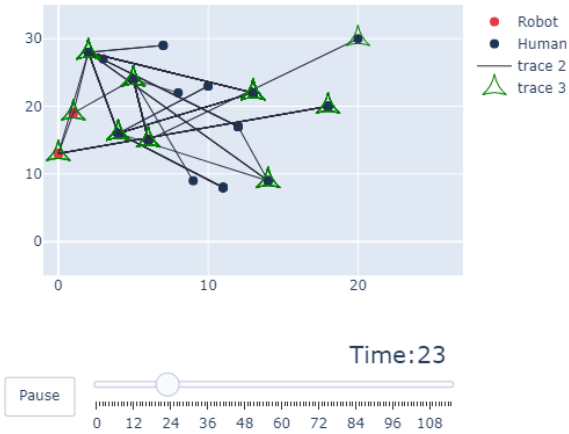
图 4-8 TwiBot-20 中 Business 聚类的完整网络图



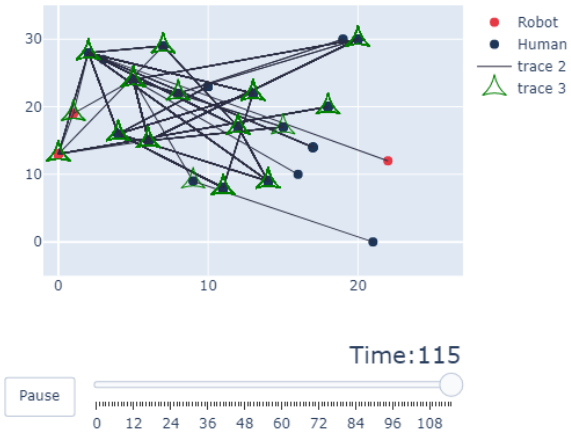
（a）从机器人发出的第一个信息传播事件



（b）信息已经影响 7 个用户

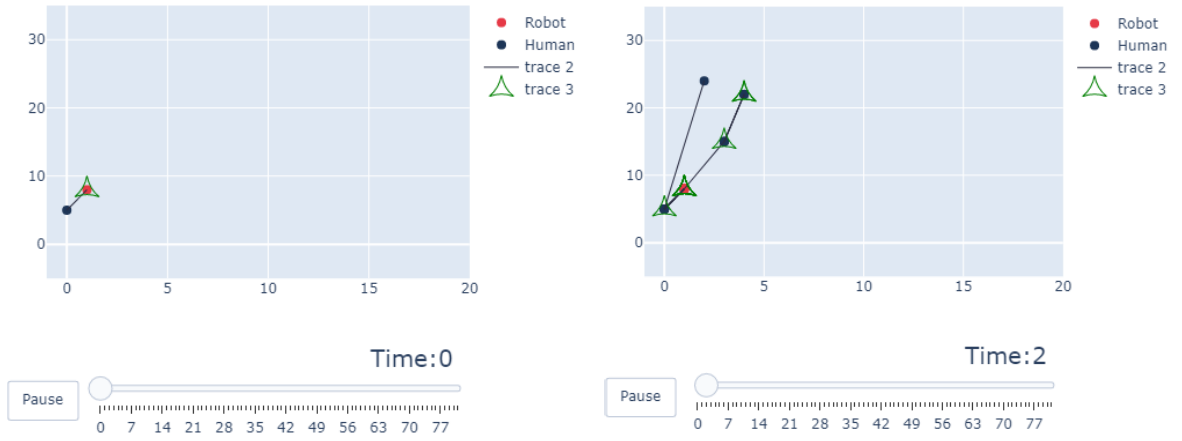


（c）信息的扩散初具规模



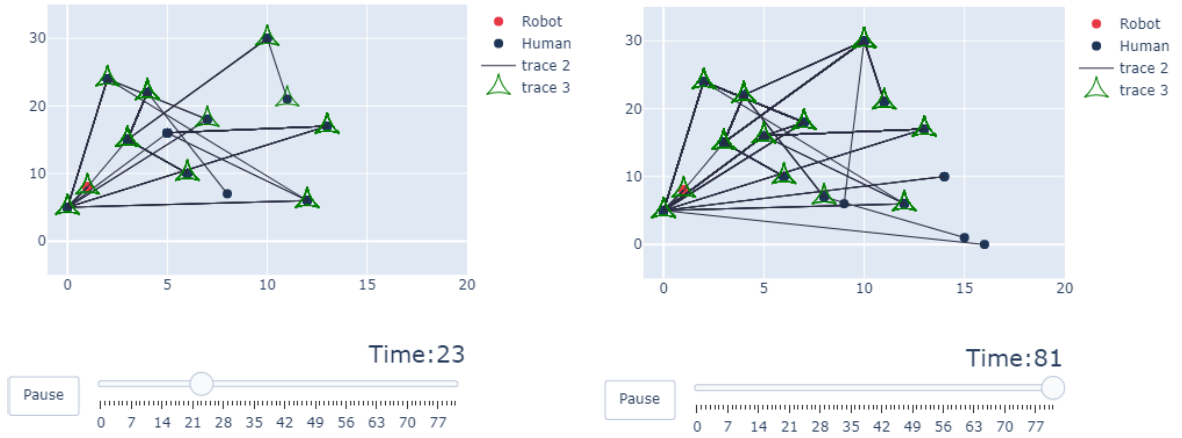
（d）信息最终的影响节点和范围

图 4-9 Politics 聚类中信息的传播过程



（a）从机器人发出的第一个信息传播事件

（b）信息已经影响 4 个用户



（c）信息的扩散初具规模

（d）信息最终的影响节点和范围

图 4-10 Business 聚类中信息的传播过程

观察图 4-9 底端代表时间的进度条能够发现，信息在用户间的传播和扩散主要集中在整个过程的开始阶段：从时间 0 到时间 23 之间，增加了 15 个新节点，而从时间 23 到时间 115 之间，只增加了 4 个新节点。

图 4-10 中信息在用户间的传播和扩散过程也是相似的：从时间 0 到时间 23 之间，增加了 11 个新节点，而从时间 23 到时间 81 之间，只增加了 4 个新节点。说明机器人发布一条推文后的一天内将是信息扩散的高峰期；若时间距离机器人发布推文越远，信息传播将更慢。这种规律是符合现实社交网络中信息的传播特征的。

4.5 本章小结

本章节主要对研究实验和结果进行展示和分析，主要分为三部分：在动态链接预测模型的正确性和有效性验证部分，本研究构建的模型在 Social Evolution 数据集上各项链接预测评价指标的结果都优于或持平于 DyRep 模型原论文中展现的结果，其中在 HITS@10 准确率高于 0.99，证明了本研究所用模型可以相对准确地进行动态链接预测；在动态社交网络链接预测部分，本模型对 TwiBot-20 数据集构建的包含社交媒体机器

人的动态社交网络进行链接预测，分析实验结果发现本模型能够适用于较大型的社交网络，对用户间的链接进行预测，并且在测试过程中不断迭代更新和学习；在社交网络信息传播部分，本文对 TwiBot-20 的实验结果进行可视化，展现社交媒体机器人用户和人类用户之间进行信息传递的过程和社交媒体机器人在社交网络中的影响范围。

5 结论与展望

5.1 研究结论

本文基于动态链接预测方法对动态社交网络中社交机器人信息传播行为进行分析研究。本研究首先搭建了时间和空间信息学习能力较强的基于时序点过程和图注意力机制的动态预测模型，并用 Social Evolution 数据集测试该模型在小型的动态社交网络中的链接预测效果。链接预测任务被分为两个类别，第一个类别为以时间为条件的链接预测，第二个类别为预测时间的链接预测任务。测试发现，模型的预测效果具有较高的准确率和较低的误差。与实验基线的链接预测结果相比，本研究使用的经过改进的模型的效果更优，证明了该链接预测模型的正确性和有效性。同时，该实验通过可视化节点表示向量的聚类过程，证实了模型具有较强的节点表示学习能力，能够让交互事件频繁的两个节点拥有更为相似的节点表示向量。

本文进一步的研究针对 Twitter 上的数据集 TwiBot-20。研究先对该原始数据集进行结构化整理，将所有的训练和测试数据构建成每一个事件格式为$e = (u, v, t, k)$的事件序列。然后，利用整理得到的数据构建大型动态社交网络并进行链接预测分析，并得出以下结论：1）本研究所使用的模型更适用于聚类系数较大的网络图。2）该模型即使在测试阶段依然能够不断学习和更新节点表示，使得模型依旧可以对距离训练结束的时间点较远的测试事件进行较为准确的链接预测。最后，本研究将数据集上时间预测结果进行处理和可视化，展示连续时间段内社交机器人和用户之间的信息交互过程，体现了社交网络中信息传播的规律：1）社交网络中信息在用户间的传播总是发散性的。2）信息传播扩散的高峰期总是处于用户发出信息的近一天内。该研究扩展的信息传播过程预测任务可以应用于监测社交媒体机器人发表的言论的影响力，从而预防恶意机器人散播虚假信息误导社会舆论。

5.2 研究展望

本研究实现了一个基础但有效的动态图网络上的链接预测模型，仍存在许多进一步研究的空间。首先，由于模型在计算链接的概率时多次使用蒙特卡洛抽样法，存在一些随机性，所以模型的时间预测任务准确率还不够理想，特别是对于聚类系数较低的数据，测试的误差偏高。因此，模型的预测任务需要寻找更准确的计算方法。其次，本研究使用的模型虽然拥有出色的节点时空信息学习能力，但由于需要在每一个事件发生后更新节点的表示向量，无法进行批并行处理，所有的训练和测试数据只能串行执行，且需要维护两个N^2大小的矩阵，因此该设定导致了模型训练耗时较长，内存占用也较大。因此，更多的研究应解决模型的开销问题。此外，本研究对社交网络机器人的监测是建立在所有用户节点身份已知的基础上，直接锁定机器人用户观察信息传播，

而从大多现实社交媒体中得到的数据并不具有节点标签以表明其机器人或人类用户的身份。因此，进一步的研究可以尝试在没有标签的网络中通过检测每一个用户的行为和信息传播过程来判断哪些用户可能是机器人用户。

致　谢

从去年 11 月确定毕设选题至今年 6 月，足足八个月的时光见证了这篇毕业论文的诞生。三分之二年的时间说长不长，不及我进入西安交通大学前准备高考的那三年，更不及我人生已经走过的 22 年又 3 个月；说短也不短，因为在此期间我也收获了曾经从未得到过的经验、技能等等。在论文即将完稿之际，我想感谢所有在我努力的道路上帮助过我的老师、父母、朋友和同学们。

学贵得师，亦贵得友。首先，我想特别感谢我的导师——罗敏楠老师。罗老师不仅是我的毕业设计导师、新蕾计划项目的导师，她也是我过好大学生活和确定未来目标的导师。在罗老师的指导下，我掌握了不少科研技能。是她首先带领着我在图神经网络领域迈出了第一步，并帮助我对科研工作有了由浅入深的认知。当我工作遇到问题时，罗老师一直耐心地为我答疑解惑、指点迷津。在毕业设计的过程中，罗老师更是在确定研究方向、查找相关资料、理论分析、数据处理和实验设置等方面都为我提供了极大的帮助。当我生活中经历困难时，也是罗老师的鼓励和支持令我能够坚定自己的理想并继续为之奋斗。罗老师给予了我学习和生活上的信心，引导我对自己的价值更加清晰，让我不再迷茫。

其次，我想感谢大学四年中我所有的课程老师。是他们让我对计算机相关的知识有了专业性的了解和掌握，帮助我奠定做出更优秀的毕业设计所需的计算机理论基础和编程技能；也是在他们的教导下，我对计算机这门学科产生了更加浓厚的兴趣，令我有决心继续攻读硕士学位。

然后，我也想感谢我的朋友和同学。他们不仅为我的毕设工作提出过改进建议，也给予了我在工作之余的生活中很多的快乐，消解了我在工作中积压的负面情绪，让我拥有积极向上的心态和饱满的精神来直面困难。

最后，我要感谢我的父母。感谢他们不仅赐予我骨肉和生命，并在二十二年的时光中一直作为我学习和生活上坚强的后盾，一直默默地给予我近乎无条件的支持和爱。是他们在我多次想要放弃的时候给我加油打气让我重拾坚持到底的决心，也是他们能够一直有耐心有兴趣听我分享学习的酸甜苦辣，生活的悲欢离合。谢谢他们，让我走过的路花团锦簇，人声鼎沸。

参考文献

[1] Newman MEJ. Clustering and preferential attachment in growing networks[J]. Physical review E, 2001, 64(2): 025102.

[2] Kossinets G, Watts D J. Empirical analysis of an evolving social network[J]. Science, 2006, 311(5757): 88-90.

[3] Kossinets G, Watts D J. Origins of homophily in an evolving social network[J]. American journal of sociology, 2009, 115(2): 405-450.

[4] Holland P W, Laskey K B, Leinhardt S. Stochastic blockmodels: First steps[J]. Social networks, 1983, 5(2): 109-137.

[5] Guimerà R, Sales-Pardo M. Missing and spurious interactions and the reconstruction of complex networks[J]. Proceedings of the National Academy of Sciences, 2009, 106(52): 22073-22078.

[6] Perozzi B, Al-Rfou R, Skiena S. Deepwalk: Online learning of social representations[C]//Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014: 701-710.

[7] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.

[8] Grover A, Leskovec J. node2vec: Scalable feature learning for networks[C]//Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016: 855-864.

[9] Zhang Z, Cui P, Zhu W. Deep learning on graphs: A survey[J]. IEEE Transactions on Knowledge and Data Engineering, 2020.

[10] Bruna J, Zaremba W, Szlam A, et al. Spectral networks and locally connected networks on graphs[J]. arXiv preprint arXiv:1312.6203, 2013.

[11] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.

[12] Veličković P, Cucurull G, Casanova A, et al. Graph attention networks[J]. arXiv preprint arXiv:1710.10903, 2017.

[13] Nguyen G H, Lee J B, Rossi R A, et al. Continuous-time dynamic network embeddings[C]//Companion Proceedings of the The Web Conference 2018. 2018: 969-976.

[14] Wang Y, Chang Y Y, Liu Y, et al. Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks[J]. arXiv preprint arXiv:2101.05974, 2021.

[15] Qu L, Zhu H, Duan Q, et al. Continuous-time link prediction via temporal dependent graph neural network[C]//Proceedings of The Web Conference 2020. 2020: 3026-3032.

[16] Ma Y, Guo Z, Ren Z, et al. Streaming graph neural networks[C]//Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2020: 719-728.

[17] Kumar S, Zhang X, Leskovec J. Predicting dynamic embedding trajectory in temporal interaction networks[C]//Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2019: 1269-1278.

[18] Trivedi R, Farajtabar M, Biswal P, et al. Dyrep: Learning representations over dynamic graphs[C]//International Conference on Learning Representations. 2019.

[19] Cox D R, Lewis P A W. Multivariate point processes[M]. Contributions to Probability Theory. University of California Press, 2020: 401-448.

[20] Aalen O, Borgan O, Gjessing H. Survival and event history analysis: a process point of view[M].

Springer Science & Business Media, 2008: 1-39.
[21] Skarding J, Gabrys B, Musial K. Foundations and modelling of dynamic networks using dynamic graph neural networks: A survey[J]. arXiv preprint arXiv:2005.07496, 2020.
[22] Twitter Inc. Final-Q4'20-TWTR-Shareholder-Letter [EB/OL]. [2021-02-09]. https://s22.q4cdn.com/826641620/files/doc_financials/2020/q4/FINAL-Q4'20-TWTR-Shareholder-Letter.pdf.
[23] Farine D. The dynamics of transmission and the dynamics of networks[J]. Journal of Animal Ecology, 2017, 86(3): 415-418.
[24] Artime O, Ramasco J J, San Miguel M. Dynamics on networks: competition of temporal and topological correlations[J]. Scientific reports, 2017, 7(1): 1-10.
[25] Metropolis N, Ulam S. The monte carlo method[J]. Journal of the American statistical association, 1949, 44(247): 335-341.
[26] Qu L, Zhu H, Duan Q, et al. Continuous-time link prediction via temporal dependent graph neural network[C]//Proceedings of The Web Conference 2020. 2020: 3026-3032.
[27] Pareja A, Domeniconi G, Chen J, et al. Evolvegcn: Evolving graph convolutional networks for dynamic graphs[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2020, 34(04): 5363-5370.
[28] Madan A, Cebrian M, Moturu S, et al. Sensing the" health state" of a community[J]. IEEE Pervasive Computing, 2011, 11(4): 36-45.