《SimGNN: A Neural Network Approach to Fast Graph Similarity Computation》

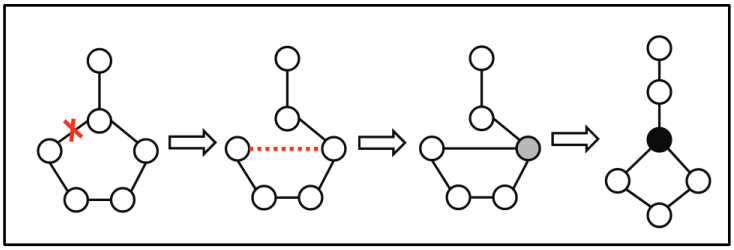
《SimGNN：一种快速图相似度计算的神经网络方法》

摘要：

1. **本文的背景：**图相似度搜索是基于图的重要应用之一，比如找到与化合物最相似的目标化合物。但是，关于图相似度/距离的计算，例如图编辑距离（GED）和最大公共子图（MCS）等方法，它们在实践中的计算成本却是很高的。因此，为了在减轻计算负担的同时又保持良好的性能，本文试图提出一种基于图神经网络的图相似度计算方法。
2. **本文的贡献：**提出了一种基于图神经网络的新方法——SimGNN，来快速计算两张图之间的相似性。
3. **主要创新点：**设计了一个可学习的嵌入函数，能将每个图映射成一个嵌入向量，从而获得一个图级的全局嵌入表示；提出了一种新型的注意力机制，能根据特定的相似性指标来找出重要节点；还设计了一种成对的节点比较方法，能用更细粒度的节点级信息来对图级嵌入进行补充。
4. **实验结果：**以 GED 计算为例，在三个真实的图数据集上的实验结果表明，SimGNN 模型相比其他基线方法，能在更短的时间内计算出结果，并且平均误差也更小。

图编辑距离（GED）：

在形式上，$G1$和$G2$之间的编辑距离，我们用$GED(G1, G2)$来表示，它是指在将$G1$转换为$G2$的最佳对齐中的编辑操作的数量，而对图G的编辑操作是指**“插入”**或**“删除”**（顶点/边）和**“重新标记”**顶点。但直观来说，如果两个图是同构的，则它们的$GED = 0$。下图展示了两个简单图之间的GED示例：



如上图所示，最左图$G1$和最右图$G2$之间的$GED = 3$，因为转换需要至少 3 个编辑操作：1）边删除；2）边插入；3）节点重新标记。

此外，一旦我们得出了两个图之间的编辑距离，我们就可以进一步将其转换为介于 0 到 1 之间的相似度指标，即对图编辑距离进行归一化，具体公式如下：

$$nGED(\mathcal{G}_1, \mathcal{G}_2) = \frac{GED(\mathcal{G}_1, \mathcal{G}_2)}{(|\mathcal{G}_1| + |\mathcal{G}_2|)/2}$$

其中，$|\mathcal{G}_i|$表示图$\mathcal{G}_i$的节点数。然后，再采用指数函数$\lambda(x) = e^{-x}$将归一化的GED转换为范围为$(0, 1]$的相似度得分$s(\mathcal{G}_1, \mathcal{G}_2)$。

注意：这里的GED和相似度得分之间存在一一对应的映射关系。

图卷积网络（GCN）：

由于在 SimGNN 中，我们需要计算每张图上各节点的嵌入表示。所以，本文使用到了当时最流行的、基于邻居聚合的方法——GCN。而它的核心操作是“图卷积”，即对一个节点的表示u_n进行操作，具体定义如下：

$$conv(u_n) = f_1\left(\sum_{m \in \mathcal{N}(n)} \frac{1}{\sqrt{d_n d_m}} u_m W_1^{(l)} + b_1^{(l)}\right)$$

其中，$\mathcal{N}(n)$是节点n的一阶邻域加上节点自身，d_n是节点n的度加一，$W_1^{(l)} \in \mathbb{R}^{D^l \times D^{l+1}}$是和 GCN 第$l$层卷积相关的权重矩阵，$b_1^{(l)} \in \mathbb{R}^{D^{l+1}}$是偏移，$f_1(\cdot)$是类似于$ReLU = max(0, x)$的激活函数。总之，直观来说，图卷积操作聚合了节点一阶邻居的特征和节点自身的特征。

SimGNN 网络结构：

概括地说，SimGNN 是一种基于端到端的神经网络方法，它试图学习一个函数来将一对图映射成一个相似度得分。而 SimGNN 的总体框架如下图所示：

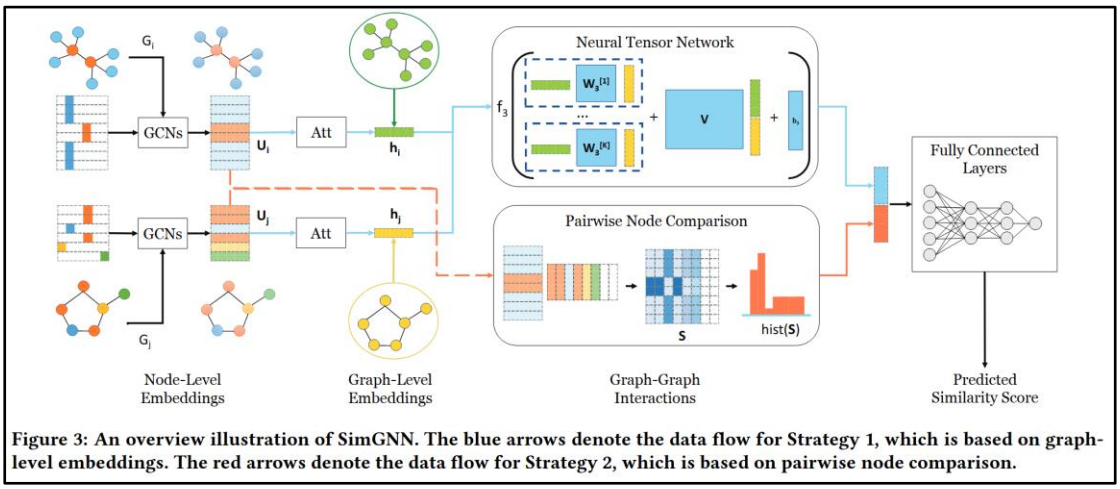
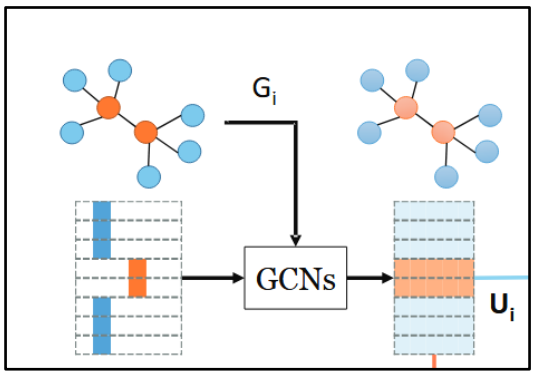


Figure 3: An overview illustration of SimGNN. The blue arrows denote the data flow for Strategy 1, which is based on graph-level embeddings. The red arrows denote the data flow for Strategy 2, which is based on pairwise node comparison.

首先，模型先将每个图的节点转换为嵌入向量，即对每个节点周围的特征和结构属性进行编码。然后，设置了两种策略来分别模拟两个图之间的相似性，一种是基于两个图级嵌入之间的交互，另一种是基于比较两组节点级的嵌入。最后，再将这两种策略的结果组合在一起，输入到一个全连接网络中，得到最终的相似度得分。

各模块具体操作如下：

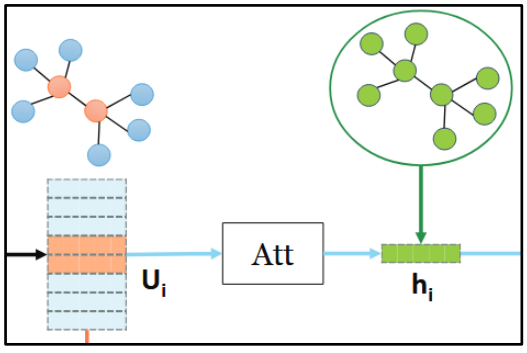


（1） 首先，对不同类型的节点进行$one-hot$编码。如上图所示，有两类共 8 个节点，所以我们就可以对应得到 8 个$one-hot$向量。

（2） 通过 Graph convolutional network 模块进行特征提取操作，即聚合节点一阶邻居的特征和节点自身的特征，GCN 公式如下：

$$conv(u_n) = f_1\left(\sum_{m\in\mathcal{N}(n)} \frac{1}{\sqrt{d_n d_m}} u_m W_1^{(l)} + b_1^{(l)}\right)$$

（3） 在经过 3 层图卷积后，我们就可以得到每个图的节点级嵌入表示了，即如上图所示的U_i。



（4） 接着，利用节点级嵌入表示得到图级嵌入表示。一种简单的方法是直接平均或者加权平均，但是考虑到不同节点的重要性不同，作者提出了一种新型的注意力机制，具体过程如下：

• 首先，采用平均法得到全图信息的表征

$$c = \tanh\left(\frac{1}{N} W_2 \sum_{m=1}^{N} u_m\right) \in \mathbb{R}^{D\times 1}, \qquad W_2 \in \mathbb{R}^{D\times D}$$

• 然后，将全图信息与节点信息进行内积的结果作为注意力系数（内积后再通过$sigmoid$进行范围压缩）

$$a_n = f_2(u_n^T c), \qquad f_2(\cdot) = \sigma(\cdot)$$

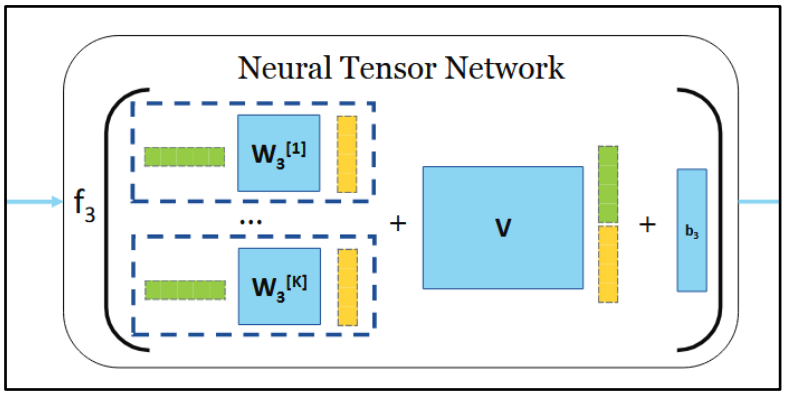
• 而在得到每个节点的注意力后，可得最终的图级嵌入表示为

$$h = \sum_{n=1}^{N} a_n u_n \in \mathbb{R}^D$$

（5） 上述注意力机制的过程可进一步等价为如下公式：

$$h = \sum_{n=1}^{N} f_2(u_n^T c) u_n = \sum_{n=1}^{N} f_2\left(u_n^T \tanh\left(\frac{1}{N} W_2 \sum_{m=1}^{N} u_m\right)\right) u_n$$

（6） 因此，现在我们就得到了每一个图的图级嵌入表示，即如上图所示的h_i。

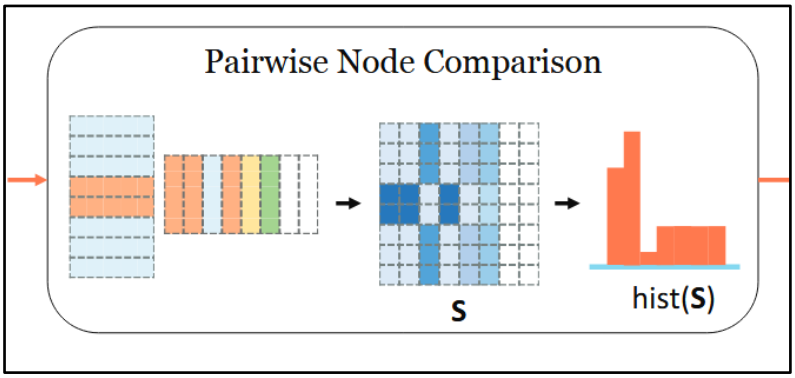


（7） 再经过 graph embedding 得到两个表示图的特征向量后，接下来我们需要计算这两个向量的相似性，一种简单的做法是直接求向量的内积，而本文在这里采用的是利用神经张量网络来计算相似度，公式如下：

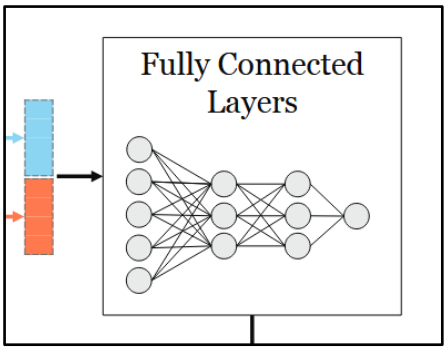
$$g(h_i, h_j) = f_3(h_i^T W_3^{[1:K]} h_j + V[h_i | h_j] + b_3)$$

其中，参数$W_3^{[1:K]} \in \mathbb{R}^{D\times D\times K}$，$V \in \mathbb{R}^{K\times 2D}$，$b_3 \in \mathbb{R}^K$，$f_3(\cdot)$代表激活函数。相关流程如上图所示。（该网络的输入就是两个图级嵌入表示，而输出为一个K维的相似度向量）

（8） 此外，为了进一步考虑局部节点的信息，本文还提出了一个成对节点比较的方法，即将前面得到的节点嵌入表示分别内积一下（维度不够的补0），然后得到一个相关性矩阵。而考虑到如何利用这个相关性矩阵，本文的方式是将其转化为直方图特征，即如下图所示：



（9） 然后，将神经张量网络输出的相似度向量与 pairwise node comparison 得到的直方图特征进行拼接，再经过几层全连接层进行维度压缩得到最终的图相似度$\hat{s}_{ij}$。如下图所示：



（10）利用以下均方误差损失函数L将前面得到的图相似度$\hat{s}_{ij}$与利用GED得到的基准相似性得分$s(\mathcal{G}_i, \mathcal{G}_j)$进行比较：

$$L = \frac{1}{|D|} \sum_{(i,j) \in D} \left(\hat{s}_{ij} - s(\mathcal{G}_i, \mathcal{G}_j) \right)^2$$

其中，D是训练图对集。

（11）最后，利用反向传播算法不断迭代优化 SimGNN 模型中的相关参数。

SimGNN 的时间复杂度分析：

一旦 SimGNN 被训练，它就可以用来计算任何一对图的相似度得分。而时间复杂度涉及两部分：

（1）节点级和全局级嵌入计算阶段，每个图需要计算一次：

- 时间复杂度为$O(E)$，其中E是图的边数。
- 注意：图级嵌入可以预先计算和存储，并且在图相似度搜索的设置中，未

见过的查询图只需处理一次即可获得其图级嵌入。

（2）相似度计算阶段，需要对每对图进行计算：

• 神经张量网络方法的时间复杂度为$O(D^2K)$，其中D是图级嵌入的维度，K是NTN的特征图维度。

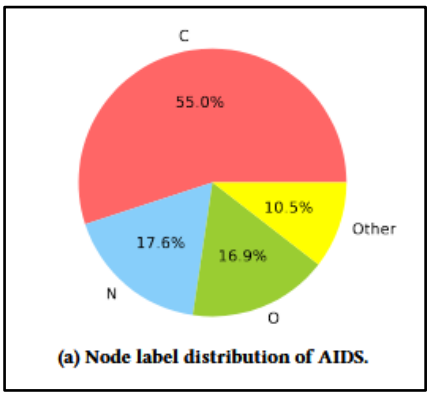
• pairwise node comparison 的时间复杂度为$O(DN^2)$，其中N是较大图中的节点数，因此我们可以通过节点采样构建相似矩阵S来减少该方法的复杂度。此外，这里的矩阵乘法也可以通过GPU进行加速。

实验及结果：

1）数据集

实验使用了三个真实世界的图形数据集。

第一个数据集是 AIDS，它是 NCI/NIH 开发治疗计划中的抗病毒筛选化合物的集合，已用于几个现有的图形相似性搜索工作。它含有 42687 种化合物结构，其中省略了氢原子。本文选择了 700 个图，而每个图都有 10 个或少于 10 个节点。每个节点都标有 29 种类型中的一种，如下图所示：



(a) Node label distribution of AIDS.

第二个数据集是 LINUX。它由 48747 个从 Linux 内核生成的程序依赖关系图（PDG）组成。每个图表示一个函数，其中节点表示一条语句，边表示两条语句之间的依赖关系。本文随机选择 1000 个图，每个图的节点数等于或小于 10 个。节点未标记。

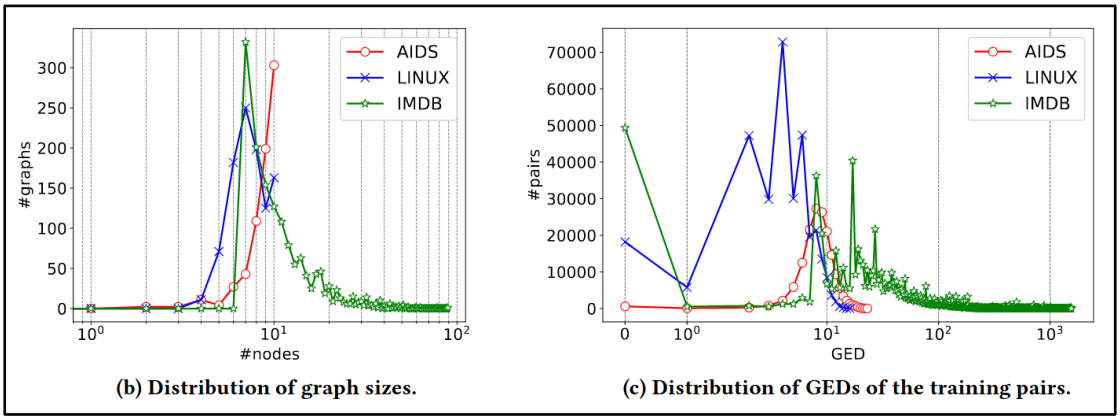
第三个数据集是 IMDB。IMDB 数据集由 1500 个电影演员的自我网络组成，如果这两个人出现在同一部电影中，就有边相连。而为了测试本文所提出的方法的可伸缩性和效率，实验中使用了完整的数据集，且节点未标记。

上述三个数据集的统计数据如下表所示：

Table 1: Statistics of datasets.

Dataset	Graph Meaning	#Graphs	#Pairs
AIDS	Chemical Compounds	700	490K
LINUX	Program Dependency Graphs	1000	1M
IMDB	Actor/Actress Ego-Networks	1500	2.25M

下图为三个数据集的图大小分布和图编辑距离分布：



(b) Distribution of graph sizes.　　(c) Distribution of GEDs of the training pairs.

2）数据预处理

对于每个数据集，我们将所有图的 60%、20%和 20%分别随机拆分为训练集、验证集和测试集。而实验的评估反映了图查询的真实场景：对于测试集中的每个图，本文将其视为一个查询图，并让模型计算查询图与数据库中每个图的相似度。数据库图根据计算的与查询的相似性进行排名。

3）参数设置

对于模型架构，本文将 GCN 的层数设置为 3，并使用 ReLU 作为激活函数。对于初始节点表示，本文针对有节点标记的 AIDS 采用反映节点类型的 one-hot 编码方案，对于没有节点标记的 LINUX 和 IMDB 采用常量编码方案。

此外，GCN 的第 1 层、第 2 层和第 3 层的输出维度分别为 64、32 和 16。对于 NTN 层，本文将 K 设置为 16。对于成对节点比较策略，本文将直方图 bin 的数量设置为 16。并且最后使用 4 个全连接层来降低 NTN 模块的串联结果，从 32 到 16、16 到 8、8 到 4 和 4 到 1。

至于训练，本文将批量大小设置为 128，并使用 Adam 算法进行梯度优化，

将初始学习率固定为 0.001，将迭代次数设置为 10000，并根据最低验证损失来选择最佳模型。

4）评价指标

本文采用以下指标用于评估所有模型：

- **时间。**收集每个模型计算一对图的相似度得分所需的时间。
- **均方误差（MSE）。**均方误差测量计算的相似度和真实相似度之间的均方差。

5）实验结果

- 有效性

Table 3: Results on AIDS.

Method	mse(10^{-3})	ρ	τ	p@10	p@20
Beam	12.090	0.609	0.463	**0.481**	0.493
Hungarian	25.296	0.510	0.378	0.360	0.392
VJ	29.157	0.517	0.383	0.310	0.345
SimpleMean	3.115	0.633	0.480	0.269	0.279
HierarchicalMean	3.046	0.681	0.629	0.246	0.340
HierarchicalMax	3.396	0.655	0.505	0.222	0.295
AttDegree	3.338	0.628	0.478	0.209	0.279
AttGlobalContext	1.472	0.813	0.653	0.376	0.473
AttLearnableGC	1.340	0.825	0.667	0.400	0.488
SimGNN	**1.189**	**0.843**	**0.690**	**0.421**	**0.514**

Table 4: Results on LINUX.

Method	mse(10^{-3})	ρ	τ	p@10	p@20
Beam	9.268	0.827	0.714	**0.973**	0.924
Hungarian	29.805	0.638	0.517	0.913	0.836
VJ	63.863	0.581	0.450	0.287	0.251
SimpleMean	16.950	0.020	0.016	0.432	0.465
HierarchicalMean	6.431	0.430	0.525	0.750	0.618
HierarchicalMax	6.575	0.879	0.740	0.551	0.575
AttDegree	8.064	0.742	0.609	0.427	0.460
AttGlobalContext	3.125	0.904	0.781	0.874	0.864
AttLearnableGC	2.055	0.916	0.804	0.903	0.887
SimGNN	**1.509**	**0.939**	**0.830**	**0.942**	**0.933**

Table 5: Results on IMDB. Beam, Hungarian, and VJ together are used to determine the ground-truth results.

Method	mse(10^{-3})	ρ	τ	p@10	p@20
SimpleMean	3.749	0.774	0.644	0.547	0.588
HierarchicalMean	5.019	0.456	0.378	0.567	0.553
HierarchicalMax	6.993	0.455	0.354	0.572	0.570
AttDegree	2.144	0.828	0.695	0.700	0.695
AttGlobalContext	3.555	0.684	0.553	0.657	0.656
AttLearnableGC	1.455	0.835	0.700	0.732	0.742
SimGNN	**1.264**	**0.878**	**0.770**	**0.759**	**0.777**

从上些表中可知，SimGNN 模型在三个数据集上均有效，且在三个数据集上的所有指标都始终能达到最佳或次佳性能。而在基于神经网络的方法中，SimGNN 在所有指标上始终取得了最佳结果。这表明 SimGNN 可以学习一个很好的嵌入函数，并且该函数也可以推广到看不见的测试图中。

• 节点注意力可视化分析

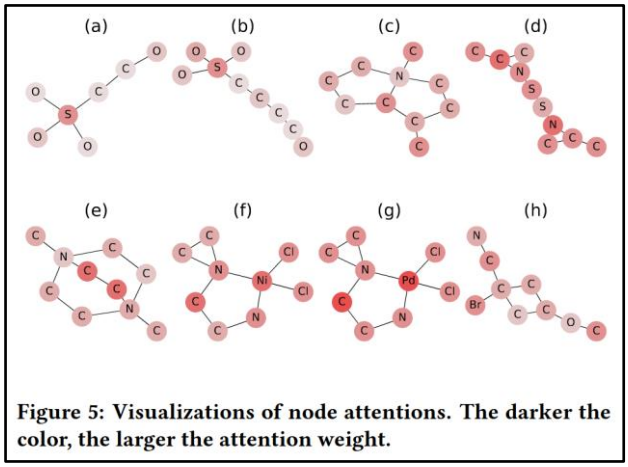


Figure 5: Visualizations of node attentions. The darker the color, the larger the attention weight.

如上图所示，可以观察到以下类型的节点能接收到相对较高的注意力权重：1）度数较大的中心节点，例如(a)和(b)中的“S”；2）带有标签的节点且很少出现在数据集中，例如(f)中的“Ni”，(g)中的“Pd”，(h)中的“Br”；3）形成特殊子结构的节点，例如(e)中的两个中间“C”等。这些模式具有直观意义，进一步证实了 SimGNN 的有效性。

• 效率分析

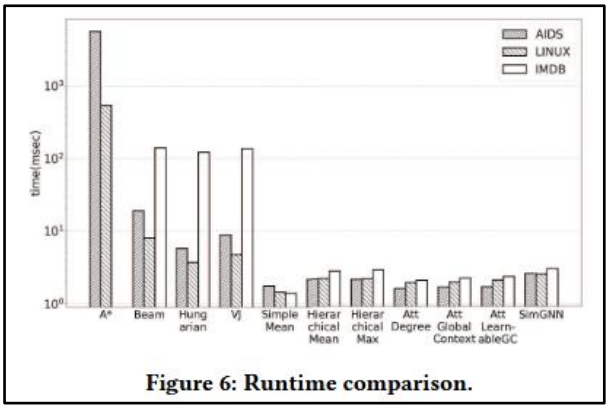
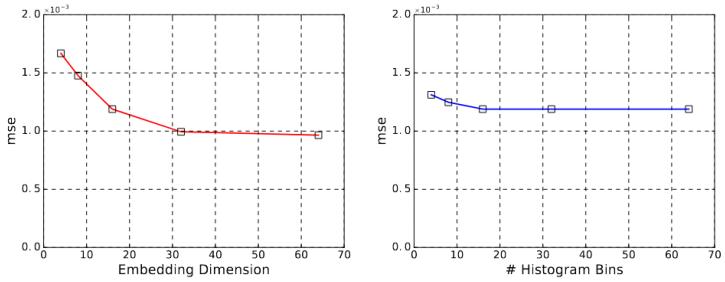


Figure 6: Runtime comparison.

三个数据集的效率比较如上图所示，而我们可以看到基于神经网络的模型在所有三个数据集上始终取得了最佳结果。与精确算法 A*相比，SimGNN 在 AIDS 上快了 2174 倍，在 LINUX 上快了 212 倍。而 A*算法甚至不能应用于大型图，并且在 IMDB 的情况下，它的变体 Beam 仍然比 SimGNN 慢 46 倍。

• 参数灵敏度分析



(a) (b)

Figure 7: Mean squared error w.r.t. the number of dimensions of graph-level embeddings, and the number of histogram bins.

如图 7a 所示，如果使用更大的嵌入维度，模型性能会变得更好。这很直观，因为更大的嵌入维度能使模型有更多的能力来表示图。而如图 7b 所示，模型性能对直方图的 bin 数量相对不敏感。这表明在实践中，只要直方图的 bin 不是太少，就可以获得相对较好的性能。

• 示例查询结果

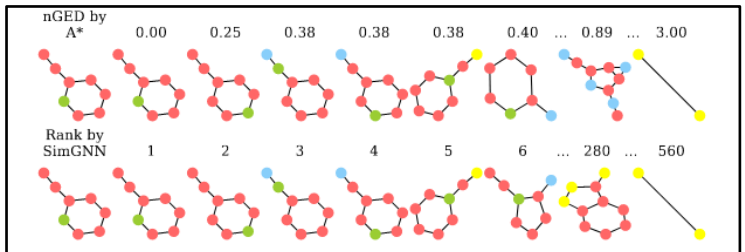


Figure 8: A query case study on AIDS. Meanings of the colors can be found in Fig. 4a.

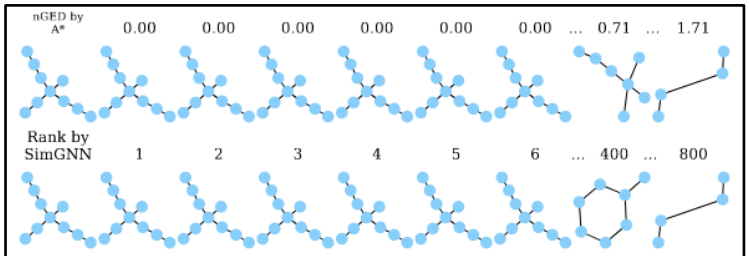


Figure 9: A query case study on LINUX.

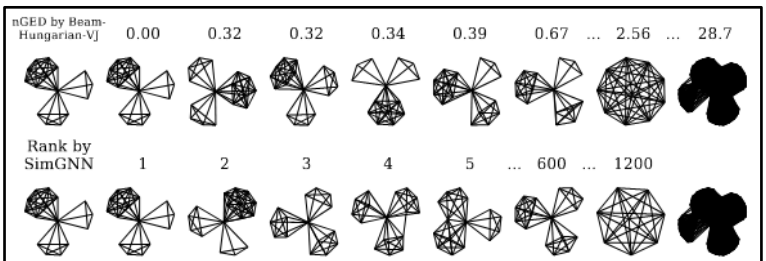


Figure 10: A query case study on IMDB.

如上图所示，每张图的第一行描述了查询以及真实排名结果，并标有查询的标准化 GED；底行显示了 SimGNN 模型返回的图表，每个图表的排名都显示在顶部。**而上述结果表明，SimGNN 有能力检索类似于查询的图形。**

总结：

本文设计了一个基于图神经网络的函数，将一对图映射到一个相似度分数。在训练阶段，该函数所涉及的参数将通过最小化预测的相似度得分与真实相似度得分之间的差异来学习，每个训练数据点都是一对图以及它们的真实相似度得分。而在测试阶段，通过向学习函数提供任意一对图，模型就可以获得预测的相似度分数。而最后的实验结果表明，SimGNN 模型能在减轻计算负担的同时又保持良好的性能。

对本文的感悟：

论文的直觉和动机都很明显，而最后的实验结果也证明了 SimGNN 的成功，也再一次间接证明了端到端学习的有效性。