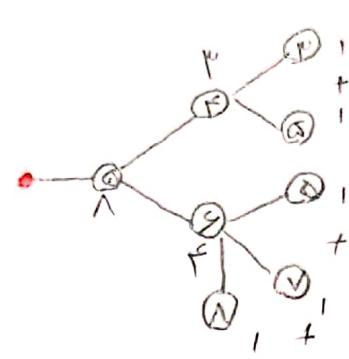
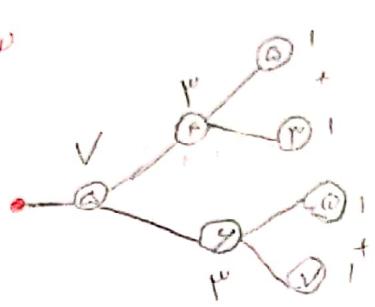


۴.۱



برای مجموع خلفاً و تند.

۴.۲

$$H^{(l)} = [h_1^{(l)}, \dots, h_{|V_l|}^{(l)}]^T$$

نفری "تیرم" از ستون به ستون $h_j^{(l+1)}$ در پرینت یعنی $h_j^{(l+1)}$ باشد که $h_j^{(l+1)}$ بستوون کی جهتیم ایند ترا نهاده اش را در نفری "تیرم" سطری سطری جهتیم

سطرن ام ماتریس مجاورت در این مرتبه A مربوط به زیر

$$\sum_{j \in N(i)} h_j^{(l)} = A_{:, i} H^{(l)}$$

$h_j^{(l)}$: j از $N(i)$ ماتریس A ماتریس $H^{(l)}$ خروج کنید (لینک ماتریس) فقری باشد در های هر قدر $M^{(l+1)}$ درجه $h_j^{(l+1)}$ باشد آن را $S^{(l+1)}$ نامید

$$D_{i,i} = \deg(i) = |N(i)|$$

$$\tilde{D}_{i,i} = \frac{1}{|N(i)|}$$

$$\sum_{j \in N(i)} \frac{h_j^{(l)}}{|N(i)|} \Rightarrow H^{(l+1)} = \tilde{D}^T A H^{(l)}$$

ماتریس انتقال \tilde{D} روی تعدادی را می توانیم صورت زیر تعریف کرد:

$$\tilde{A} = \tilde{D}^{-1} A$$

ماتریس مجاورت در این ماتریس مجاورت در این درج درج ماتریس ماتریس

عنصر مورب در مکانی انتقال مورب را نهایی می‌داند، برای نمای سازی ماتریس مجاورت و اطمینان از اینکه مجموع سطرها ماتریس انتقال آن بروز است، می‌توان این نظریه را تضمین کرد که random walk بر طرفانه اندیش و رأس های بارگذاری شده باشد. صحیده داده نشود و این انتقال روش تغییر دهنده است. عنصر غیر مورب آن نهایی دهنده انتقال از این را در مقدار مورب (انتظار بردار ورودی) (embedding)، رسماً به آن کار کردن می‌گویند توسعه برد، از random walk است. آنرا زیرا دلیل شروع لینی و برآمد که قدرم می‌باشد uniform random walk (جایی که

$\tilde{A} \rightarrow \text{ماتریس مورب} \rightarrow \text{ماتریس اسی}$

۴۰۹

برای یادگیری الگوریتم GNN، BFS و توزیع تابع پیام با جمع (aggregation) و تجزیه (disaggregation) از رسانه‌زنی مورب بازگشتی تعریف نمی‌شوند.

تابع پیام: برای یک رأس معین i تابع پیام $m_i^{(t)}$ که $m_i^{(t)} = \sum_j m_{ij}^{(t)}$ می‌باشد و می‌تواند پیام embedding مسایر خود را در یافتن می‌کند.

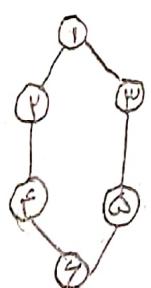
تابع تجمع: برای یک رأس می‌توان تابع تجمع در اینجا $m_i^{(t+1)} = \max_j(m_{ij}^{(t)}, \alpha)$ را در نظر گرفت که α مقدار پیام $m_i^{(t)}$ است.

قانون روزانه: برای یک رأس معین i قانون روزانه $m_i^{(t+1)} = \max_j(m_{ij}^{(t)}, \alpha \cdot \gamma_i)$ می‌گذرد.

که در آن نمایه ورودی را اس نویسید که اگر نه رأس نمایه باشد تا در غیر این فورست ه است
که همان جهادگری embedding هر چیزی را به رأس موردنظر منتقل کردارد ریاضی فوایم اطلاعاتی
در مورد این کدام راس ها تا نون تو سط BFS دیره شوده از زیر مذکور کنیم.

تابع تجمع در اینجا مفهوم را زده هستیم که این ریاضی فوایم این راس ها را به راس
در محدودیت راس ها که هم رسای ایش بروز لبیم.

خانواده روزرسانی embedding: هر راس بر اساس درخت هست که در میان embedding های ویا میان راس های
شروعه از هر راس خوب است که در این قرود کردن منبع هر روزگار را به نسبت های
راس های منبع در هر تکرار بروزگردی شود و اطلاعات مربوطه راس های بازدید شده را این میان راس های
با مجموعه از تابع های تجمع و مفهون به روزرسانی GNN می نویزد الگوریتم BFS را به اینجا
داده برای درزگاه embedding های در زنایت به عالیت همراه می شوند در آن همه راس های که مرتباً
راس مبرآ به آنها در لایر رسای پیدا کرد دارای یک embedding با همانند و مادر روش گذاشت
و درستند



در این گراف راس نه راس نمایه است و هر فرد گیری BFS ایست میتوانیم
ویگر ورودی نخرا برای راس نه تا در نظر بگیریم و برای رسای بروز

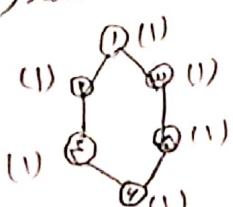
و GNN. مرادل زیر را انجام می دهد:

فقدانی اولیه: راس نه دارای ویگر ورودی تا است و همه راس های دیگر دارای
ویگر ورودی هستند هنر راس های دارای embedding اولیه هستند
هر کس را بجهوه

راس نه های بیام ترا به راس نه که ۲ و ۳ می خواهد
راس نه که ۲ و ۳ بیام ترا دریافت کرده و embedding خود را به بروزی کنند
راس های که ۴ و ۵ بیام ترا به راس های ۴ و ۵ رسای کنند
راس نه های که ۶ و ۷ بیام ترا دریافت کنند و embedding خود را به بروزی کنند
راس نه های که ۸ و ۹ بیام ترا دریافت کنند و embedding خود را به بروزی کنند

در زنایت تمام رئوس کا به دسترسی از راس نه دارای embedding تا هستند و سایر رئوس که راس

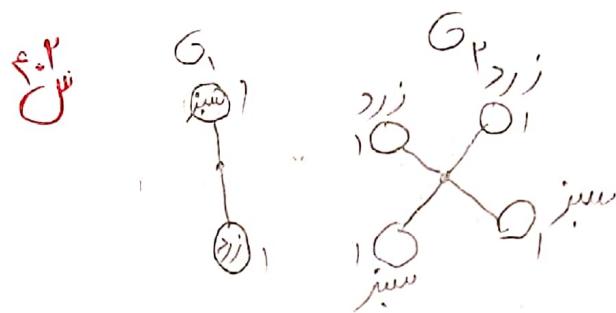
و هستند embedding



اگر مکاراف داشته باشیم بخواهیم بینیم که رفت و گرفت هستند یا نه. من تها میتوانم مکاراف تمازتر باشم
ایجاد میکردیم که همکنست پس این میکردیم که رفت و گرفت های میکنند اما فراز و رفوس مکاراف دلیر شد
و باعث میباشد تصریح از نیم که کله دهارا مخفف کند
فرصل سیزدهمی به نام مکاراف در جمله ای از مکاراف دلیر شد

$\phi: A \rightarrow 1 \quad B \rightarrow \emptyset \quad H \rightarrow 2 \quad G \rightarrow 4 \quad D \rightarrow 3 \quad C \rightarrow V \quad E \rightarrow \emptyset \quad F \rightarrow 1$

۲۷) تمازتر بر قرار است مکاراف های پرینتند.



$$\max(h_{j_1}) = 1 \quad \max(h_{j_2}) = 1$$

$$\text{mean}(h_{j_1}) = 1 \quad \text{mean}(h_{j_2}) = \frac{p}{2} = 1$$

$$\max(h_{j_3}) = 1 \quad \max(h_{j_3}) = 1$$

$$\text{mean}(h_{j_3}) = 1 \quad \text{mean}(h_{j_3}) = 1$$

$$\sum(h_{j_1}) = p \quad \sum(h_{j_2}) = p$$

$$\sum(h_{j_3}) = 3$$

در این کرد مکاراف پرینتند زیرا مکاراف اعداد پیشنهادی بودند و مکاراف دلیر شدند. \max , mean , \sum

$$\max = 4$$

$$\text{mean} = 2$$

$$\sum = 9$$

$$\max = 4$$

$$\text{mean} = 1$$

$$\sum = 1$$

سیم

برای اثبات اینکه GNN برای شرطی ℓ -دافت محدود به اندازه زمان WL قدرتمند است و تو
از تکنیکی به نام تعقیفی روش استفاده کرد که اینهاست WL است. هر آنچه این روش در راه بررسی
برحسب آن و برحسب هایی همسایه نشان دهد که این فضایی در این فضایی WL در راه بررسی

هر راس v میتوانیم هایی که نطاویش میشوند. میتوان نشان داد که اگر ℓ -نمودار را باید GNN با قوای
AGGREGATE و COMBINE گابت دو نمودار غیرهم‌سلام در رایه v باشد ℓ -نمودار v باشد آنکه جمع ℓ -نمودار
را ℓ -نمودار v باشد که آن در راه را kth فرآیند بااین روش باشان است هر دو نمودار این بدان معنیست که WL است
فر توانند بین نفوذ رهایز خالی شود و بنابراین هر آنوریکم پیشی بر GNN حداقل به $(n+1)$ است WL قدرتمند است
فرض زندگی، و بگراف های غیر یک ریخت هستند لذا WL آنها در نمودار v با همان توابع

برای $COMBINE$ و $AGGREGATE$ هر روز شود اگر تابع READOUT خروجی های متغیری باشد که تویید
کند در آن صورت باشد WL است که WL است که WL خواهد بود از v
تابع READOUT قدرتمند است زیرا READOUT باید صدقه ℓ -نمودار را باید ℓ -نمودار v باشد
بر عکس اگر WL است WL نتواند بین v که v تهایز خالی شود ℓ -نمودار v باشد ℓ -نمودار v باشد
باشد و تابع READOUT باید خروجی v باشد اگر صدقه ℓ -نمودار v تویید کند
میتوان با استفاده از برهان تففه باشد که WL -test WL -test WL قدرتمند باشد فرضی
که نسبت ℓ -نمودار v میتواند تعمیم بلور کرده آیا ℓ -دافت های غیر یک ریخت ℓ -نمودار v در راه ای
برای $node embedding$ خود یک ریخت هستند چنانست اگر توابع COMBINE و AGGREGATE برای

آنها بر روز شده باشند همچنانست v این هستند تابع READOUT برای هر دو نمودار خروجی v باشد
که دانسته WL -test WL -test WL قدرتمند باشد ای ℓ -دافت های غیر یک ریخت هستند v ای ℓ -نمودار v باشد
این کار را انجام دهد که از WL -test قدرتمند v دارد هر kth WL -test kth v باشد
بارگذاری صورت داشته باشد که v در مرحله از آزمون WL بخواستی آیدی از v باشد v و
 v یک ریخت نیستند لیکن مقدار v و موددارد که $(G_1)_{kth}$ و $(G_2)_{kth}$ یک ریخت نیستند
برحسب رأسنایی $(G_1)_{kth}$ و $(G_2)_{kth}$ را هر راس v نمودار آزمون WL در نظر بگیرید
از آنچه باید گوییم این برحسب های مقایسه هند مجموعه از برجسته های دارای قابل برای هر رأس

و $\text{Hk}(G_1)$ و $\text{Hk}(G_2)$ می‌باشد. می‌آید نتیجه‌سی شود که بر حسب‌های READOUT باز
بنابراین ممکن است دارایم: اگر مدل عصبی مانندوارد باشد، آن‌ها که توانند مسائل شود نابع
از خروجی ریسمانی برای هر دو گزینه $\text{Hk}(G_1)$ و $\text{Hk}(G_2)$ ممکن است که بر حسب‌های $\text{Hk}(G_1)$ و $\text{Hk}(G_2)$
متغیر است. بنابراین مدل عصبی مانندوارد باید $\text{WL-best} \neq \text{WL}$ قدرتی دارد. بنابراین مدل عصبی
که در آن که مدل عصبی به اندازه WL-best مادرست است.

Definition 3.3.1. The unfolding tree T_v^d of a node v up to depth d is

$$T_v^d = \begin{cases} \text{Tree}(\ell_v) & \text{if } d = 0 \\ \text{Tree}(\ell_v, T_{ne[v]}^{d-1}) & \text{if } d > 0 \end{cases}$$

where $\text{Tree}(\ell_v)$ is a tree constituted of a single node with label ℓ_v and $\text{Tree}(\ell_v, T_{ne[v]}^{d-1})$ is the tree with the root node labeled with ℓ_v and having sub-trees $T_{ne[v]}^{d-1}$. The set $T_{ne[v]}^{d-1} = \{T_{u_1}^{d-1}, T_{u_2}^{d-1}, \dots\}$ collects all unfolding trees having depth $d - 1$, with $u_i \in ne[v]$, $\forall i$.

Moreover, the *unfolding tree of v*, $T_v = \lim_{d \rightarrow \infty} T_v^d$, is obtained by merging all unfolding trees T_v^d for any d . ■



3.4 The Weisfeiler–Lehman test

The *first order Weisfeiler–Lehman test* (*1-WL test* in short) [16] is a method to test whether two graphs are isomorphic, based on a graph coloring algorithm. The coloring algorithm is applied in parallel on the two input graphs. In the end, the number of nodes for each color is counted and the numbers obtained relative to the two graphs are compared: if the numbers match, then the graphs are possibly isomorphic, while if they do not match, then the graphs are certainly non-isomorphic. Note that the test is not conclusive in the case of a positive answer, as the graphs may still be non-isomorphic. Actually, the algorithm just provides an approximate solution to the problem of graph isomorphism.

There exist different versions of the coloring algorithm: in this paper, we adopt a coloring scheme in which also the node labels are considered. Since GNNs process both the structure and the labels of the graphs, it is useful to consider both these sources of information, in order to analyse the GNN expressive power. Such an approach has been used, for example, in [26]. More precisely, the coloring is carried out by an iterative algorithm which, at each iteration, computes a *node coloring* $c_l^{(t)} \in \Sigma$, being Σ a subset of values representing the colors. The node colors are initialized on the basis of the node features and then they are updated using the coloring from the previous iteration. The algorithm is sketched in the following.

1. At iteration 0, we set

$$c_v^{(0)} = \text{HASH}_0(\ell_v)$$

where HASH_0 is a function that bijectively codes every possible feature with a color in Σ .

2. For any iteration $t > 0$, we set

$$c_v^{(t)} = \text{HASH}((c_v^{(t-1)}, \{c_n^{(t-1)} | n \in ne[v]\}))$$

where HASH bijectively maps the above pair to a unique value in Σ , which has not been used in the previous iterations.

The algorithm terminates if the number of colors between two iterations does not change, i.e. when the cardinalities of $\{c_n^{(t-1)} | n \in V\}$ and $\{c_n^{(t)} | n \in V\}$ are equal.

Finally, we can introduce the equivalence that is induced on nodes by the WL test.

Theorem 4.1.1.

Let $G = (V, E)$ be a labeled graph. Then, for each $u, v \in V$, $u \sim_{ue} v$ if and only if $u \sim_{WL} v$ holds. □

Theorem 4.1.2.

Let G_1, G_2 be two graphs. Then, $G_1 \sim_{ue} G_2$ if and only if $G_1 \sim_{WL} G_2$. □

Let $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ be a graph and let $u, v \in \mathbf{V}$, with features ℓ_u, ℓ_v . Then, $\forall t \in \mathbb{N}$

$$\mathbf{T}_u^t = \mathbf{T}_v^t \text{ iff } c_u^{(t)} = c_v^{(t)} \quad (4)$$

where $c_u^{(t)}$ and $c_v^{(t)}$ represent the node coloring of u and v at time t , respectively.

Proof. The proof is carried out by induction on t , which represents both the depth of the unfolding trees and the iteration step in the WL colouring.

For $t = 0$, $\mathbf{T}_u^0 = \text{Tree}(\ell_u) = \text{Tree}(\ell_v) = \mathbf{T}_v^0$ if and only if $\ell_u = \ell_v$ and $c_u^{(0)} = \text{HASH}_0(\ell_u) = \text{HASH}_0(\ell_v) = c_v^{(0)}$. Let us suppose that Eq. (4) holds for $t - 1$, and prove that it holds also for t .

(\rightarrow) Assuming that $\mathbf{T}_u^t = \mathbf{T}_v^t$, we have

$$\mathbf{T}_u^{t-1} = \mathbf{T}_v^{t-1} \quad (5)$$

and

$$\text{Tree}(\ell_u, \mathbf{T}_{ne[u]}^{t-1}) = \text{Tree}(\ell_v, \mathbf{T}_{ne[v]}^{t-1}) \quad (6)$$

By induction, Eq. (5) is true if and only if

$$c_u^{(t-1)} = c_v^{(t-1)} \quad (7)$$

Eq. (6) implies that $\ell_u = \ell_v$ and $\mathbf{T}_{ne[u]}^{t-1} = \mathbf{T}_{ne[v]}^{t-1}$, which means that an ordering on $ne[u]$ and $ne[v]$ exists s.t.

$$T_{ne_i(u)}^{t-1} = T_{ne_i(v)}^{t-1} \quad \forall i = 1, \dots, |ne[u]| \quad (8)$$

Hence, Eq. (8) holds iff an ordering on $ne[u]$ and $ne[v]$ exists s.t.

$$c_{ne(u)_i}^{t-1} = c_{ne(v)_i}^{t-1} \quad \forall i = 1, \dots, |ne[u]|$$

that is

$$\{c_m^{(t-1)} | m \in ne[u]\} = \{c_n^{(t-1)} | n \in ne[v]\} \quad (9)$$

Putting together Eqs. (7) and (9), we obtain:

$$\begin{aligned} \text{HASH}((c_u^{(t-1)}, \{c_m^{(t-1)} | m \in ne[u]\})) &= \\ \text{HASH}((c_v^{(t-1)}, \{c_n^{(t-1)} | n \in ne[v]\})) \end{aligned}$$

which implies that $c_u^{(t)} = c_v^{(t)}$

- (\leftarrow) The proof of the converse implication follows a similar reasoning, but some different steps are required in order to reconstruct the unfolding equivalence from the equivalence based on the 1-WL test.

Let us assume that $c_u^{(t)} = c_v^{(t)}$; by definition,

$$\begin{aligned} \text{HASH}((c_u^{(t-1)}, \{c_m^{(t-1)} | m \in ne[u]\})) &= \\ \text{HASH}((c_v^{(t-1)}, \{c_n^{(t-1)} | n \in ne[v]\})) \end{aligned} \quad (10)$$

Being the HASH function bijective, Eq. (10) implies that:

$$c_u^{(t-1)} = c_v^{(t-1)} \quad (11)$$

and

$$\{c_m^{(t-1)} | m \in ne[u]\} = \{c_n^{(t-1)} | n \in ne[v]\} \quad (12)$$

Eq. (11) is true if and only if, by induction,

$$\mathbf{T}_u^{t-1} = \mathbf{T}_v^{t-1} \quad (13)$$

which implies

$$\ell_u = \ell_v \quad (14)$$

Moreover, Eq. (12) means that an ordering on $ne[u]$ and $ne[v]$ exists such that

$$c_{ne(u)_i}^{t-1} = c_{ne(v)_i}^{t-1} \quad \forall i = 1, \dots, |ne[u]| \quad (15)$$

Instead, by induction, Eq. (15) holds iff an ordering on $ne[u]$ and $ne[v]$ exists so as $T_{ne_i(u)}^{t-1} = T_{ne_i(v)}^{t-1} \quad \forall i = 1, \dots, |ne[u]|$, i.e.

$$\mathbf{T}_{ne[u]}^{t-1} = \mathbf{T}_{ne[v]}^{t-1} \quad (16)$$

Finally, putting together Eqs. (14) and (16), we obtain

$$\text{Tree}(\ell_u, \mathbf{T}_{ne[u]}^{t-1}) = \text{Tree}(\ell_v, \mathbf{T}_{ne[v]}^{t-1})$$

that means $\mathbf{T}_u^t = \mathbf{T}_v^t$