دانشگاه صنعتی امیرکبیر

(پلی‌تکنیک تهران)

دانشکده مهندسی کامپیوتر

پروژه کارشناسی

پیاده سازی و مقایسه شبکه های عصبی گرافی با ماتریس های وزن قابل یادگیری و بدون ماتریس های وزن قابل یادگیری روی وظایف مختلف

نگارش

محمدرضا رضائی

استاد راهنما

دکتر مصطفی حقیر چهرقانی

فروردین ۱۴۰۴

بسم الله الرحمن الرحیم

صفحه فرم ارزیابی و تصویب پایان نامه- فرم تأیید اعضاء کمیته دفاع

در این صفحه فرم دفاع یا تایید و تصویب پایان نامه موسوم به فرم کمیته دفاع- موجود در پرونده آموزشی- را قرار دهید.

نکات مهم:

- نگارش پایان نامه/رساله باید به زبان فارسی و بر اساس آخرین نسخه دستورالعمل و راهنمای تدوین پایان نامه‌های دانشگاه صنعتی امیرکبیر باشد.(دستورالعمل و راهنمای حاضر)
- رنگ جلد پایان نامه/رساله چاپی کارشناسی، کارشناسی ارشد و دکترا باید به ترتیب مشکی، طوسی و سفید رنگ باشد.
- چاپ و صحافی پایان نامه/رساله بصورت پشت و رو(دورو) بلامانع است و انجام آن توصیه می شود.

دانشگاه صنعتی امیرکبیر
(پلی‌تکنیک تهران)

به نام خدا

تعهدنامه اصالت اثر

تاریخ: فروردین ۱۴۰۴

اینجانب **محمدرضا رضائی** متعهد می‌شوم که مطالب مندرج در این پایان‌نامه حاصل کار پژوهشی اینجانب تحت نظارت و راهنمایی اساتید دانشگاه صنعتی امیرکبیر بوده و به دستاوردهای دیگران که در این پژوهش از آنها استفاده شده است مطابق مقررات و روال متعارف ارجاع و در فهرست منابع و مآخذ ذکر گردیده است. این پایان‌نامه قبلاً برای احراز هیچ مدرک هم‌سطح یا بالاتر ارائه نگردیده است.

در صورت اثبات تخلف در هر زمان، مدرک تحصیلی صادر شده توسط دانشگاه از درجه اعتبار ساقط بوده و دانشگاه حق پیگیری قانونی خواهد داشت.

کلیه نتایج و حقوق حاصل از این پایان‌نامه متعلق به دانشگاه صنعتی امیرکبیر است. هرگونه استفاده از نتایج علمی و عملی، واگذاری اطلاعات به دیگران یا چاپ و تکثیر، نسخه‌برداری، ترجمه و اقتباس از این پایان نامه بدون موافقت کتبی دانشگاه صنعتی امیرکبیر ممنوع است. نقل مطالب با ذکر مآخذ بلامانع است.

محمدرضا رضائی

امضا

در همه دیر مغان نیست چو من شیدایی خرقه جایی گرو باده و دفتر جایی

سپاس‌گزاری

با کمال افتخار و قدردانی، این فرصت را مغتنم شمرده تا سپاس و قدردانی خود را از کسانی که در طول انجام این پروژه کارشناسی همراه و حامی من بوده‌اند، ابراز دارم. در ابتدا، از استاد راهنمای عزیزم، که با صبر و حوصله فراوان، دانش و تجربه‌ی بی‌بدیل خود را در اختیار من قرار داده و در هر مرحله از این پژوهش، راهنما و پشتیبان من بوده‌اند، تشکر ویژه‌ای دارم.

همچنین، قدردانی خود را نسبت به خانواده‌ی محترم و دوستان عزیزم که بی‌دریغ، عشق، حمایت و انگیزه‌ی لازم را برای پشتکار و ادامه‌ی تلاش‌هایم در این مسیر فراهم آورده‌اند، ابراز می‌دارم. وجود آنها چراغ راهی بوده است که در دشوارترین لحظات، روشنی بخش راه من بوده‌اند.

محمدرضا رضائی

فروردین ۱۴۰۴

چکیده

شبکه‌های عصبی گرافی [۱] یک چارچوب مؤثر برای یادگیری از داده‌های ساختاریافته [۲] هستند که معمولاً از ماتریس‌های وزن قابل یادگیری برای تجمیع ویژگی‌ها و انتشار اطلاعات در گراف استفاده می‌کنند. با این حال، برخی مدل‌های جدید نشان داده‌اند که حذف این وزن‌ها می‌تواند ضمن حفظ عملکرد، پیچیدگی محاسباتی را کاهش دهد. این پژوهش به مقایسه GNNهای دارای ماتریس وزن قابل یادگیری و بدون آن در وظایف مختلف می‌پردازد و دو معماری GCN و LightGCN را از نظر تأثیر وزن‌ها بر تجمیع ویژگی‌ها، قابلیت بیان‌گری [۳] مدل و کارایی محاسباتی بررسی می‌کند. مدل‌ها بر روی مجموعه داده‌های متنوع، در وظایف دسته‌بندی گره [۴] و پیش‌بینی پیوند [۵] ارزیابی شده و معیارهای استاندارد برای تحلیل عملکرد آن‌ها به کار گرفته شده است. نتایج این پژوهش به درک نقش ماتریس‌های وزن در GNNها کمک کرده و راهنمایی‌هایی درباره انتخاب معماری مناسب در کاربردهای مختلف، از جمله شبکه‌های اجتماعی و سیستم‌های توصیه‌گر [۶]، ارائه می‌دهد.

واژه‌های کلیدی:

GNN، GCN، LightGCN

[1] GNN
[2] structured data
[3] expressiveness
[4] node classification
[5] link prediction
[6] recommendation systems

فهرست مطالب

عنوان | صفحه

۱ مقدمه ۱

۱-۱ تعریف مسئله ۲

۱-۲ فهرست سایر فصل ها ۴

۲ تعاریف ۶

۳ کارهای قبلی ۱۱

۳-۱ شبکه عصبی گراف (GNN) توسط اسکارسلی و همکاران.[۵] (۲۰۰۹) ۱۲

۳-۲ Graph Convolutional Network (GCN) توسط Kipf و Welling [۴] (۲۰۱۷) . ۱۳

۳-۳ GraphSAGE توسط همیلتون و همکاران.[۲] (۲۰۱۷) ۱۴

۳-۴ شبکه‌های عصبی پیام‌رسانی (MPNN) توسط Gilmer [۱]و همکاران در سال ۲۰۱۷ ۱۶

۳-۵ شبکه توجه گرافی (GAT) توسط Veličković و همکاران[۶] (۲۰۱۸) ۱۷

۳-۶ شبکه هم‌ریختی گراف (GIN) توسط Xu و همکاران[۹] (۲۰۱۹) ۱۹

۳-۷ ساده‌سازی GCN (SGC) توسط Wu و همکاران[۷] (۲۰۱۹) ۲۰

۳-۸ LightGCN توسط He و همکاران[۳] (۲۰۲۰) ۲۱

۳-۹ یک بررسی جامع از GNNها توسط Wu و همکاران[۸] (۲۰۲۰) ۲۳

۳-۱۰ شبکه‌های عصبی گرافی: یک مرور توسط Zhou و همکاران[۱۰] (۲۰۲۰) ۲۵

۴ روش شناسی ۲۸

۴-۱ توضیح پیاده سازی ۲۹

۴-۱-۱ بارگذاری و پیش‌پردازش داده‌ها ۲۹

۴-۱-۲ تعریف مدل ۳۰

۴-۱-۳ آموزش مدل ۳۲

۴-۱-۴ ارزیابی مدل ۳۲

۴-۲ اجزای سیستم و یکپارچه‌سازی با GraphGym ۳۳

۴-۲-۱ ساختار گراف ۳۳

۴-۲-۲ تبدیل لایه GCN ۳۳

۴-۲-۳ فرآیند آموزش ۳۴
۴-۲-۴ معماری مدل ۳۴
۴-۲-۵ استفاده از GraphGym برای تنظیم مدل ۳۵
۴-۳ رابط کاربری ۳۵

۵ نتایج تجربی ۳۸
۵-۱ توضیح مجموعه داده ها ۳۹
۵-۱-۱ cora ۳۹
۵-۱-۲ citeseer ۴۰
۵-۱-۳ Pubmed ۴۰
۵-۱-۴ Amazon Photo ۴۱
۵-۲ معیارهای ارزیابی ۴۳
۵-۲-۱ دقت ۴۳
۵-۲-۲ دقت مثبت ۴۳
۵-۲-۳ بازخوانی ۴۴
۵-۲-۴ امتیاز F1 . . . . . ۴۴
۵-۲-۵ AUC-ROC ۴۴
۵-۳ نتایج ۴۵
۵-۳-۱ عملکرد طبقه بندی گره (Amazon Photo, Cora) ۴۶
۵-۳-۲ عملکرد پیش‌بینی پیوند (PubMed, Citeseer) ۴۸

۶ نتیجه گیری ۵۱
۶-۰-۱ GCN ۵۲
۶-۰-۲ GAT ۵۲
۶-۰-۳ LightGCN ۵۳
۶-۰-۴ نتیجه نهایی ۵۴

منابع و مراجع ۵۶

فهرست تصاویر

شکل صفحه

۴-۱ فرایند آموزش ۳۴
۴-۲ معماری مدل ۳۴
۴-۳ رابط کاربری ۳۶
۴-۴ رابط کاربری ۳۷

فهرست جداول

جدول صفحه

۵-۱ معیار دقت ۴۵
۵-۲ معیار دقت مثبت ۴۵
۵-۳ معیار بازخوانی ۴۵
۵-۴ معیار امتیاز F1 . . . . . ۴۵
۵-۵ معیار دقت ۴۵
۵-۶ معیار دقت مثبت ۴۶
۵-۷ معیار بازخوانی ۴۶
۵-۸ معیار امتیاز F1 . . . . . ۴۶

فصل اول

مقدمه

۱-۱ تعریف مسئله

داده های ساختاریافته کاربرد های فراوانی دارند که از جمله آنها میتوان به شبکه های اجتماعی، تجارت الکترونیک، زیست شناسی و حمل و نقل اشاره کرد. برخلاف داده های شبکه ای، گراف ها از گره ها و یال هایی با الگو های اتصال دلخواه تشکیل شده اند. یادگیری موثر از چنین داده هایی نیازمند مدل هایی است که بتوانند وابستگی گره های متصل به هم را ثبت کنند. شبکه های عصبی گرافی به عنوان ابزاری قدرتمند برای این منظور معرفی شدند زیرا میتوانند وابستگی گراف ها را از طریق ارسال پیام گره ها دریافت کنند. در سال های اخیر، شبکه های عصبی گراف توانستند با یادگیری پارامتر های ویژگی و توپولوژی گراف به نتایج خیره کننده ای دست یابند. این امر باعث پیشرفت در کارهایی همچون طبقه بندی گره ها، پیشبینی پیوند ها و طبقه بندی گراف ها شده است. برای مثال از شبکه های عصبی گرافی برای مدل سازی سیستم های فیزیکی، پیش بینی خواص مولکولی، شناسایی عملکرد های پروتئین و حتی استدلال بر ساختار های استخراج از متن و تصویر استفاده شده است. این موفقیت ها، جست و جو برای معماری های شبکه های عصبی گرافی برای مشکلات متنوع مرتبط با گراف را انگیزه بخشیده است. این پروژه بر پیاده سازی و مقایسه مدل های شبکه های عصبی گرافی با ماتریس وزن قابل یادگیری و مدل های شبکه های عصبی گرافی بدون ماتریس وزن بدون یادگیری تمرکز دارد. در شبکه های عصبی گراف معمولی، هر لایه یک تبدیل خطی آموخته شده را به ویژگی های گراف در طول فرایند ارسال پیام [۱] اعمال میکند. برخی مطالعات نشان میدهد که معماری های ساده تر بدون چنین ماتریس های وزنی به همان خوبی عمل میکنند و میتوانند کارایی را نیز بهبود دهند. LightGCN یکی از نمومه هایی است که تبدیل ویژگی و فعال سازی غیرخطی را از لایه های GCN حذف میکند و با حفظ اجزای ضروری، عملکرد بهتری را در وظایف فیلتر مشترک ارائه میدهد. این موضوع سوال مهمی را ایجاد میکند: وزن های قابل یادگیری چقدر به عملکرد GNN کمک میکند و آیا طراحی های بی وزن میتوانند پیچیدگی را بدون کاهش دقت کاهش دهند؟ با مقایسه GNN های استاندارد (آنهایی که از ماتریس وزن در هر لایه استفاده میکنند مانند GCN) با GNN های ساده شده (آنهایی که اطلاعات را بدون تغییر وزن میانی منتشر میکنند مانند LightGCN)، این پروژه به دنبال ارائه بینشی در مورد این سوال است. ما به دنبال این هستیم که موازنه [۲] در بیانگری [۳] و کارایی را کمی کنیم. بر اساس تحقیقات قبلی، پیش بینی

[1] message passing
[2] trade-off
[3] expressiveness

میشود که مدل های بی وزن به طور قابل توجهی هزینه محاسباتی و خطر بیش برازش [۴] را کاهش دهند و نتایجی رقابتی را برای وظایفی خاص ارائه کنند در حالیکه کارهایی که نیاز به یادگیری ویژگی های پیچیده دارند ممکن است از وزن های قابل یادگیری بهره بیشتری ببرند. نتیجه پروژه نشان میدهد که آیا معماری های GNN را میتوان برای دامنه کاربردی خاصی بدون از دست دادن عملکرد ساده کرد یا خیر. در طول سالها، مدل های مختلفی از GNN ها معرفی شدند. GNN های تکرارشونده [۵] جزو اولین ها بودند که به عنوان چارچوبی برای اعمال شبکه های عصبی بر روی گراف ها با انتشار مکرر اطلاعات تا زمان همگرایی معرفی شدند. انواع مدرن GNN بر اساس این ایده با مکانیسم های مختلف ساخته شده اند. به عنوان مثال شبکه های کانولوشن گراف [۶] از یک رویکرد پیچیدگی طیفی برای تجمیع [۷] ویژگی های همسایه گره به صورت خطی استفاده میکنند و قدرت GNN ها را برای طبقه بندی گره ها به صورت نیمه نظارت [۸] شده نشان میدهند. شبکه های توجه گراف [۹] مکانیزم توجهی را معرفی میکنند که به گره ها اجازه میدهد که اهمیت همسایه های مختلف را در طول تجمیع، وزن کنند. همچنین مدل هایی استقرایی مانند GraphSAGE وجود دارند که یک تابع تجمیع را یاد میگیرند که میتواند با نمونه برداری از اطلاعات همسایگی، تعبیه [۱۰] هایی را برای گره های نادیده قبلی ایجاد کنند. این مدل های عمومی GNN ایده مشترک تجمیع اطلاعات همسایه تکرار شونده را دارند اما در نحوه وزن دهی یا یادگیری پیام ها متفاوت هستند. چندین کار اخیر ارتباط نزدیکی با رویکرد این پایان نامه دارد. همانطور که گفته شد، LightGCN یک GNN ساده شده است که برای سیستم های توصیه کننده طراحی شده است که ماتریس های وزن هر لایه و توابع غیرخطی را حذف میکند اما همچنان از مدل های پیچیده تر GNN در آن حوزه بهتر عمل میکند. کار مرتبط دیگر، ساده سازی GCN توسط Wu et al است که نشان داد میتوان فعال سازی های غیرخطی [۱۱] و ماتریس های وزن متوالی در یک GCN برای ایجاد یک مدل خطی را بدون تاثیر قابل توجهی بر دقت حذف کرد. این نتایج نشان میدهد که برای وظایفی خاص، تغییر ویژگی در هر لایه ممکن است غیرضروری باشد و این فرضیه، ما را الهام میدهد که یک GNN بی وزن میتواند کافی باشد. از سوی دیگر، GNN های با بیانگری بیشتر مانند شبکه ایزومورفیسم

[4] overfitting
[5] recurrent GNNs
[6] GCNs
[7] aggregation
[8] semi-supervised
[9] GAT
[10] embedding
[11] non-linear activation

گراف [۱۲] نشان داده اند که طراحی دقیق مراحل تجمیع و به روز رسانی میتواند یک GNN را به اندازه تست ایزومورفیسم گراف Weisfeiler-Lehman قدرتمند کند. بنابراین طیفی بین GNN های ساده و پیچیده وجود دارد. هدف این پایان نامه این است که ارزیابی کند که در کجای این طیف، وظایف گوناگون بهترین عملکرد را دارند.

در این پروژه، ابتدا مدل های GAT، GCN و LightGCN را پیاده سازی کردیم و این مدل ها را با مجموعه داده های Cora، AmazonPhoto، PubMed و CiteSeer آموزش دادیم. مقادیر به دست آمده از معیار های ارزیابی را باهم مقایسه کردیم و نتیجه گرفتیم که یک مدل بهینه برای همه مجموعه داده ها و مسائل وجود ندارد. انتخاب مدل مناسب وابسته به ماهیت مسئله است و هر یک از مدل های مذکور در دسته ی متفاوتی از مسائل کاربرد دارند.

۱-۲ فهرست سایر فصل ها

بقیه پایان نامه به شرح زیر تنظیم شده است:

- تعاریف(فصل ۲): مفاهیم کلیدی و اصطلاحات مورد استفاده در سراسر پایان نامه تعریف شده است، از جمله گراف، شبکه های عصبی، شبکه های عصبی گراف، تعبیه گره، ارسال پیام و انتشار پس زمینه[۱۳]. این بخش زمینه لازم را برای خوانندگانی که با GNN آشنایی قبلی ندارند فراهم میکند.

- کارهای قبلی(فصل ۳): مروری بر تحقیقات قبلی تأثیرگذار بر روی شبکه های عصبی گراف ارائه شده است. ما ده کار اصلی را خلاصه می‌کنیم که تکامل از مدل‌های اولیه GNN تا پیشرفت‌ها و روش‌های اخیر مرتبط با رویکرد ما را پوشش می‌دهد.

- روش شناسی (فصل ۴): جزئیات مدل های GNN پیاده سازی شده ارائه شده است. معماری های مدل، فرایند آموزش، وظایف و دیتاست هایی که ارزیابی بر روی آنها انجام شده است را توصیف میکنیم. مشارکت و اصلاحات جدید در مدل های موجود نیز بررسی شدند.

- نتایج تجربی (فصل ۵): این بخش نتایج آزمایش هایی که عملکرد مدل‌های GNN وزن‌دار و بی‌وزن را مقایسه میکنند را گزارش می‌کند. ارزیابی کمی در مورد وظایف متعدد ارائه می‌کنیم،

[12] Graph Isomorphism Network (GIN)

[13] backpropagation

اثربخشی مدل‌ها را تجزیه و تحلیل می‌کنیم و مشاهداتی مانند دقت، کارایی و رفتار همگرایی را مورد بحث قرار می‌دهیم.

- نتیجه‌گیری (بخش ۶): ما با خلاصه‌ای از یافته‌ها، پیامدهای طراحی GNN و مسیرهای احتمالی برای کارهای آینده را نتیجه‌گیری می‌کنیم. محدودیت های مطالعات فعلی و بهبودهای بالقوه نیز مورد بحث قرار می گیرند.

فصل دوم

تعاریف

برای اطمینان از وضوح، چندین اصطلاح و مفهوم کلیدی مربوط به شبکه های عصبی گراف را تعریف می کنیم:

- گراف: گراف یک ساختار ریاضی است که برای مدل سازی روابط زوجی بین اشیا استفاده می شود. به طور رسمی، یک گراف G با مجموعه ای از رئوس (یا گره ها) V و مجموعه ای از یال های E تعریف می شود، جایی که هر یال دو راس را به هم متصل می کند. در گراف های ساده، یال یک جفت نامرتب از رئوس متمایز (گراف بدون جهت) یا یک جفت مرتب (گراف جهت دار) است. از گراف ها می توان برای نمایش شبکه هایی مانند شبکه های اجتماعی (افراد به عنوان گره و دوستی ها به عنوان یال)، شبکه های استنادی (مقاله ها و نقل قول ها)، ساختارهای مولکولی (اتم ها و پیوندها) و غیره استفاده کرد. هر گره v در یک گراف ممکن است ویژگی های مرتبطی داشته باشد و یال ها را می توان بسته به کاربرد، وزن دهی یا دسته بندی کرد.

- شبکه عصبی: شبکه عصبی یک مدل محاسباتی متشکل از لایه‌هایی از نورون‌های (یا گره‌ها) به هم پیوسته است که داده‌های ورودی را از طریق وزن‌های آموخته شده تبدیل به داده هایی جدید می‌کند. با الهام از شبکه های عصبی بیولوژیکی، یک شبکه عصبی مصنوعی[1] از یک لایه ورودی، یک یا چند لایه پنهان از واحدها و یک لایه خروجی تشکیل شده است. هر نورون در یک لایه، ورودی ها را جمع می کند (به عنوان یک مجموع وزنی) و نتیجه را از یک تابع فعال سازی عبور می دهد و شبکه را قادر می سازد تا نگاشت غیرخطی را یاد بگیرد. شبکه‌های عصبی بر روی داده‌ها آموزش داده می‌شوند تا وزن‌های خود را طوری تنظیم کنند که خروجی به طور تقریبی تابع هدف را شبیه سازی کند. انواع متداول شبکه های عصبی عبارتند از شبکه‌های پیش‌خور[2] (پرسپترون‌های چندلایه)، شبکه‌های عصبی کانولوشنال (برای داده‌هایی با ساختار شبکه‌ای مانند تصاویر)، و شبکه‌های عصبی تکراری (برای داده‌های متوالی). شبکه های عصبی در استخراج خودکار ویژگی ها و تقریب توابع پیچیده از طریق یادگیری لایه ای بسیار موفق هستند.

$$f(x) = f^{(L)}(f^{(L-1)}(\dots f^{(1)}(x)))$$

$$f^{(l)}(x) = \sigma^{(l)}\left(W^{(l)}x + b^{(l)}\right)$$

[1] ANN

[2] feed-forward networks

- شبکه عصبی گراف[3]: یک شبکه عصبی گراف یک معماری شبکه عصبی است که بر روی داده های ساختار یافته گراف عمل می کند. شبکه‌های عصبی گراف مدل‌های عصبی هستند که وابستگی گراف ها را از طریق ارسال پیام بین گره‌های گراف ثبت می‌کنند. در یک GNN، هر گره یک نمایش ویژگی[4] را حفظ می کند که به طور مکرر با جمع آوری اطلاعات از همسایگان خود در گراف به روز می شود. این ارتباط همسایه به گره (ارسال پیام) به GNN ها اجازه می دهد تا نمایش هایی را بیاموزند که هم ویژگی های گره و هم الگوهای اتصال گراف را رمزگذاری می کنند. GNN ها را می توان به عنوان تعمیم شبکه های عصبی دیگر (مانند CNN) به گراف های دلخواه دید: در حالی که یک CNN اطلاعات پیکسلی را از یک شبکه معمولی فیلتر و ترکیب می کند، یک GNN اطلاعات را از همسایگی نامنظم یک گره منتشر و ترکیب می کند. GNN های گوناگون در نحوه جمع آوری پیام های همسایه و به روز رسانی حالت های گره متفاوت هستند اما همه به اصل یادگیری در ساختارهای رابطه ای[5] پایبند هستند. GNN ها را می توان بر روی یک گراف کامل (تولید تعبیه در سطح گراف) یا بر روی گره ها و یال ها برای کارهایی مانند طبقه بندی گره، پیش بینی پیوند و غیره اعمال کرد.

- تعبیه گره: در یادگیری گراف، تعبیه گره، نمایش برداری از یک گره در یک فضای ویژگی پیوسته است. هدف این است که نقش و ویژگی‌های ساختاری یک گره را در یک بردار کم بعد رمزگذاری کنیم، به طوری که شباهت در فضای برداری، شباهت در گراف را نشان دهد. به عبارت دیگر، تعبیه های گره، نمایش‌های برداری کم بعدی از گره‌ها در یک گراف هستند که اطلاعات توپولوژیکی و یا ویژگی‌های گراف را به تصویر می‌کشند. تعبیه ها انجام وظایف بر روی گراف ها را برای مدل‌های یادگیری ماشین آسان‌تر می‌کنند، زیرا آنها اتصال گراف گسسته را به قالبی مناسب برای الگوریتم‌هایی مانند طبقه‌بندی یا رگرسیون[6] تبدیل می‌کنند. تعبیه های گره را می‌توان از طریق روش‌های مختلفی یاد گرفت: برای مثال، رویکردهای بدون نظارت[7] مانند الگوریتم‌های مبتنی بر پیاده‌روی تصادفی (DeepWalk، node2vec) با پیش‌بینی بافت همسایگی، تعبیه ها را یاد می‌گیرند، در حالی که GNNهای تحت نظارت[8]، تعبیه ها را با آموزش در مورد یک وظیفه خاص

[3] GNN
[4] embedding
[5] relational structures
[6] regression
[7] unsupervised
[8] supervised

یاد می‌گیرند (ویژگی های گره از طریق پس انتشار اصلاح می شوند تا برای آن وظیفه معنی دار باشند). یک تعبیه خوب، گره‌هایی که در شبکه، نقش ها و ویژگی های مشابهی دارند را در فضای برداری نیز در کنار هم قرار میدهد و کارهای پایین دستی مانند خوشه‌بندی، طبقه‌بندی یا پیش‌بینی پیوند را تسهیل می‌کند.

- ارسال پیام: ارسال پیام مکانیزم اصلی است که توسط آن GNN ها اطلاعات را از طریق یک گراف منتشر می کنند. در هر لایه ی (یا مرحله زمانی) یک GNN، هر گره، «پیام‌ها» را از همسایگان خود جمع‌آوری می‌کند و سپس وضعیت خود (تعبیه) را بر اساس پیام جمع‌آوری شده به روزرسانی می‌کند. به طور معمول، ارسال پیام شامل دو مرحله است: (۱) تجمیع - هر گره اطلاعات را از همسایگان مستقیم خود جمع آوری یا جمع می کند. (۲) به روز رسانی - گره از اطلاعات جمع آوری شده از همسایه (و اغلب وضعیت قبلی خود) برای محاسبه وضعیت جدید استفاده می کند. تابع تجمیع باید جایگشت ثابت باشد (انتخاب های رایج شامل جمع، میانگین یا حداکثر بردارهای ویژگی همسایه است) تا نتیجه به ترتیب همسایگان وابسته باشد. به روز رسانی معمولاً توسط یک شبکه عصبی (به عنوان مثال، یک لایه کاملاً متصل[۹]) اجرا می شود که تعبیه فعلی گره را با پیام جمع آوری شده ترکیب می کند. به عنوان مثال، یک به روز رسانی ساده برای ارسال پیام برای گره v می تواند: "فرمول اینجا آورده شود" W یک ماتریس وزن است. انواع پیشرفته‌تر توجه را معرفی می‌کنند (جایی که پیام‌ها با امتیازات توجه وزن می‌شوند) یا عملکردهای پیام و به روزرسانی را مانند چارچوب شبکه عصبی ارسال پیام جدا می‌کنند. از طریق چندین دور ارسال پیام (لایه‌ها)، یک گره می‌تواند اطلاعات را از قسمت‌های دورتر گراف (همسایگی k-hop) جمع‌آوری کند. این روش به GNN ها اجازه می دهد تا استدلال رابطه ای را انجام دهند، زیرا تعبیه هر گره در نهایت حاوی اطلاعاتی در مورد زیرگراف محلی آن است.

$$h_v^{(l+1)} = \text{UPDATE}^{(l)}\left(h_v^{(l)},\ \text{AGGREGATE}^{(l)}\left(\{h_u^{(l)} : u \in \mathcal{N}(v)\}\right)\right)$$

- پس انتشار: پس انتشار الگوریتم اصلی برای آموزش شبکه های عصبی با تنظیم وزن ها در جهت مخالف گرادیان خطا است. در زمینه GNN ها (و شبکه های عصبی به طور کلی)، پس انتشار به محاسبه گرادیان یک تابع خطا[۱۰] با توجه به هر پارامتر مدل و انتشار آن گرادیان ها به عقب در

[9] fully connected layer

[10] loss function

لایه های شبکه اشاره دارد. پس انتشار روشی است که برای تنظیم وزن های اتصال برای جبران هر خطای یافت شده در حین یادگیری استفاده می‌شود. پس انتشار گرادیان تابع هزینه[11] را با توجه به وزن‌ها محاسبه می‌کند. در طول آموزش، ابتدا یک پاس رو به جلو برای محاسبه خروجی شبکه و خطا (تفاوت بین پیش بینی و هدف واقعی) انجام می شود. سپس، پس انتشار به طور موثر قانون زنجیره ای حساب دیفرانسیل و انتگرال را اعمال می کند تا تعیین کند که تغییرات در هر وزن چگونه بر خطا تأثیر می گذارد. سپس هر وزن (معمولاً از طریق نزول گرادیان تصادفی[12] یا یک نوع از آن) متناسب با منفی این گرادیان به روز می شود و در نتیجه خطا را کاهش می دهد. در GNN ها، پس انتشار نه تنها از طریق محاسبات شبکه عصبی در هر گره، بلکه از طریق فرآیند ارسال پیام نیز عمل می کند و به مدل اجازه می دهد تا یاد بگیرد که چگونه اطلاعات همسایه را به طور بهینه برای یک وظیفه جمع کند. پس انتشار به طور مکرر در بسیاری از دوره ها ادامه می یابد تا زمانی که مدل به حالتی همگرا شود که در آن ضرر به حداقل می رسد (یا دیگر به طور قابل توجهی بهبود نمی یابد).

$$\delta^{(l)} = \left(W^{(l+1)}\right)^T \delta^{(l+1)} \circ \sigma' \left(z^{(l)}\right)$$

$$\frac{\partial \mathcal{L}}{\partial W^{(l)}} = \delta^{(l)} \left(a^{(l-1)}\right)^T$$

$$w \leftarrow w - \eta \frac{\partial \mathcal{L}}{\partial w}$$

[11] cost function

[12] stochastic gradient descent

فصل سوم

کارهای قبلی

تحقیقات در شبکه‌های عصبی گراف به سرعت پیشرفت کرده است و مدل‌ها و چارچوب‌های تاثیرگذار زیادی به وجود آمده اند. در زیر ده مقاله مهم مرتبط با GNN ها را خلاصه می کنیم، که از مدل های اساسی کلی شروع می شود و سپس رویکردهای خاص مرتبط با مطالعه خود را ذکر می کنیم:

۳-۱ شبکه عصبی گراف (GNN) توسط اسکارسلی و همکاران.[۵] (۲۰۰۹)

اسکارسلی و همکاران (۲۰۰۹) اولین چارچوب کلی برای شبکه های عصبی گراف را معرفی کردند که پایه و اساس تحقیقات GNN را بنا گذاشت. این مدل شبکه عصبی گراف (GNN) برای مدیریت داده های ساختاریافته گراف با انتشار مکرر اطلاعات در یال های گراف طراحی شد. این معماری از مکانیزم ارسال پیام مکرر استفاده می‌کند: وضعیت هر گره بر اساس حالات همسایگانش به‌روزرسانی می‌شود، و این فرآیند تا رسیدن به تعادل پایدار (نقطه ثابت) تکرار می‌شود. این رویکرد تکراری، مانند شبکه عصبی بازگشتی Elman که بر روی ساختار گراف پیاده می‌شود، مدل را قادر می‌سازد تا وابستگی‌های پیچیده را در گراف‌های دلخواه ثبت کند.

مشارکت‌های اصلی: این کار وظایف متمرکز بر گره و گراف را تحت یک چارچوب عصبی متحد می‌کند - GNN توانست خروجی‌های سطح گره (مانند طبقه‌بندی‌ها) و خروجی‌های سطح گراف (مانند پیش‌بینی کل گراف) را با استفاده از یک فرآیند تکرار شونده تولید کند. ایده انتقال پیام عصبی بر روی گراف ها را معرفی کرد، جایی که "پیام ها" (اطلاعات) بین گره های همسایه برای یادگیری نمایش گره ها ارسال می شوند. اسکارسلی و همکاران همچنین یک الگوریتم یادگیری ویژه را برای آموزش این GNN های مکرر پیشنهاد کردند و اطمینان حاصل کردند که به روز رسانی های تکراری به یک راه حل پایدار همگرا می شوند.

نتایج کلیدی: نتایج نشان داد که GNN می‌تواند الگوهایی را در داده‌های گراف (مانند کوتاه‌ترین مسیرها یا مشکلات طبقه‌بندی گراف) با موفقیت یاد بگیرد که یادگیری آنها با روش‌های قبلی دشوار بود.

اهمیت: به عنوان یکی از اولین مدل‌های GNN، این کار این مفهوم را ایجاد کرد که شبکه‌های عصبی می‌توانند مستقیماً بر روی ساختارهای گراف کار کنند و این مدل تقریباً بر تمام معماری‌های GNN بعدی تأثیر گذاشت. این تحقیق، هم امید به استفاده از یادگیری عمیق روی گراف و هم چالش های آن (فرآیند تکراری از نظر محاسباتی گران بود) را برجسته کرد و محققان را راهنمایی کرد تا در

سال های بعد به دنبال انواع کارآمدتر GNN ها باشند. به طور خلاصه، GNN اسکارسلی و همکاران یک طرح اساسی ارائه کرد - یک شبکه عصبی ارسال پیام مکرر متناسب گراف ها - که مسیر پژوهش درباره GNN را ایجاد کرد.

۳-۲ Graph Convolutional Network (GCN) توسط Kipf و Welling [۴] (۲۰۱۷)

شبکه کانولوشن گراف توماس کیپف و مکس ولینگ (GCN) پیشرفتی بود که GNN ها را در یادگیری نیمه نظارت شده بر روی گراف ها رایج کرد. GCN یک رویکرد پیچیدگی طیفی ساده شده را روی گراف ها معرفی کرد: از تئوری پردازش سیگنال گراف شروع می‌شود، اما یک فیلتر موضعی را استخراج می‌کند که می‌تواند به طور موثر در حوزه فضایی گراف اعمال شود. در عمل، هر لایه GCN نمایش گره را با میانگین‌گیری ویژگی‌های همسایگان خود، ترکیب آنها با ویژگی‌های خود گره، و سپس اعمال تبدیل خطی و فعال‌سازی غیرخطی، به‌روزرسانی می‌کند. با روی هم قرار دادن چنین لایه‌هایی، اطلاعات به چندین گام دورتر منتشر می‌شوند. در مقایسه با روش‌های طیفی پیشین، GCN چندین بهبود ایجاد کرد: با استفاده از تقریب مرتبه اول چندجمله‌ای‌های چبیشف، از محاسبات سنگینی مانند تجزیه ویژه اجتناب کرد و با افزودن یک ترفند بازنرمال‌سازی، پایداری را تضمین نمود. این ساده‌سازی‌ها به طور چشمگیری پیچیدگی را کاهش دادند در حالی که عملکرد را حفظ کردند، که باعث شد GCN برای شبکه‌های بزرگ عملی باشد.

مشارکت‌های اصلی: GCN نشان داد که انجام کانولوشن گراف با عملیات ساده ماتریسی امکان‌پذیر است (در اصل، ضرب ماتریس مجاورت نرمال‌شده در ماتریس ویژگی‌های گره)، که باعث شد شهود شبکه‌های عصبی کانولوشنی به داده‌های ساختاریافته به صورت گراف تعمیم یابد. این روش در طبقه‌بندی نیمه‌نظارت‌شده‌ی گره‌ها در مجموعه داده‌های استنادی به نتایج پیشرفته‌ای دست یافت و روش‌های قبلی تعبیه‌ی گراف را با اختلاف قابل توجهی پشت سر گذاشت.

نتایج کلیدی: با وجود تنها تعداد کمی گره دارای برچسب در هر کلاس، GCN توانست از اتصال گراف و ویژگی‌های گره برای گسترش اطلاعات برچسب‌ها استفاده کند و دقت را به میزان قابل توجهی افزایش دهد. سادگی این روش (در اصل تنها چند خط جبر خطی) باعث شد GCN به‌راحتی پیاده‌سازی شود و از نظر محاسباتی کارآمد باشد، که این امر پذیرش آن را تسهیل کرد.

اهمیت: این کار به یکی از سنگ بناهای یادگیری ماشین روی گراف تبدیل شد – ارزش پیام‌رسانی کانولوشنی روی گراف‌ها را اثبات کرد و نسل جدیدی از تحقیقات در زمینه‌ی شبکه‌های عصبی گرافی را پایه‌گذاری کرد. بسیاری از مدل‌های بعدی (مانند GraphSAGE، GAT و غیره) بر اساس ایده‌های معرفی‌شده توسط GCN توسعه یافتند. به طور خلاصه، کیپف و ولینگ (۲۰۱۷) شبکه‌های عصبی گرافی را به جریان اصلی تحقیقات وارد کردند، با ارائه‌ی روشی قدرتمند اما ساده برای یادگیری روی داده‌های شبکه‌ای، که گامی بزرگ در پیشرفت یادگیری نیمه‌نظارت‌شده روی گراف‌ها بود.

۳-۳ GraphSAGE توسط همیلتون و همکاران.[۲] (۲۰۱۷)

GraphSAGE (نمونه گراف و جمع) توسط همیلتون و همکاران یک چارچوب یادگیری القایی برای شبکه‌های عصبی گرافی معرفی کرد که در تعمیم به گره‌های نادیده‌گرفته‌شده یا گراف‌های کاملاً جدید عملکرد بالایی دارد. شبکه‌های عصبی گرافی پیشین عمدتاً انتقالی [۱] بودند، به این معنا که برای آموزش به حضور تمامی گره‌های گراف نیاز داشتند اما GraphSAGE یاد می‌گیرد که چگونه تعبیه‌های گره را تولید کند، به جای اینکه آن‌ها را حفظ کند، که این امر امکان پیش‌بینی در گراف‌های بزرگ و در حال تغییر را فراهم می‌کند. ایده‌ی اصلی شامل آموزش مجموعه‌ای از توابع نمونه‌گیری همسایگان و تجمیع ویژگی‌ها است: برای هر گره، GraphSAGE تعداد ثابتی از همسایه‌ها را نمونه‌گیری می‌کند، اطلاعات ویژگی آن‌ها را (با استفاده از میانگین‌گیری، تجمیع مبتنی بر ادغام[۲] یا تجمیع مبتنی بر LSTM) ادغام کرده و از این اطلاعات برای به‌روزرسانی نمایش گره استفاده می‌کند. با روی هم قرار دادن چندین لایه از این نوع، یک گره می‌تواند اطلاعات را از همسایگی‌های چندگامی جمع‌آوری کند. نکته‌ی کلیدی این است که مدل، پارامترهای تابع تجمیع را روی یک زیرگراف آموزشی یاد می‌گیرد و سپس می‌تواند همان تابع را برای تولید تعبیه‌های گره‌های جدیدی که قبلاً دیده نشده‌اند، اعمال کند. این توانایی استقرایی[۳]، GraphSAGE را برای سناریوهای پویا مناسب می کند (مانند شبکه‌های اجتماعی که در آن کاربران جدید به سیستم اضافه می‌شوند) و گراف‌های بسیار بزرگی که آموزش به صورت دسته[۴] کامل در آن‌ها امکان‌پذیر نیست.

مشارکت های اصلی: نمونه‌گیری همسایگان در GraphSAGE مقیاس‌پذیری را به میزان قابل

[1] transductive
[2] pooling
[3] inductive
[4] batch

توجهی بهبود می‌بخشد. به جای استفاده از تمامی همسایه‌ها که ممکن است درجه‌ی بسیار بالایی داشته باشند، تنها تعداد محدودی از آن‌ها را نمونه‌گیری می‌کند، که این امر محاسبات و حافظه‌ی مورد نیاز برای هر دسته را کاهش می‌دهد. علاوه بر این، GraphSAGE چندین معماری مختلف برای تجمیع معرفی کرد. تجمیع میانگین، مشابه میانگین‌گیری در GCN، یکی از این روش‌ها بود. روش دیگر، تجمیع مبتنی بر LSTM بود که امکان ترکیب ویژگی‌ها به ترتیبی خاص را فراهم می‌کرد. همچنین، یک روش تجمیع مبتنی بر ادغام با استفاده از شبکه‌ی عصبی به کار گرفته شد که اطلاعات همسایگان را به صورت غنی‌تر ترکیب می‌کرد. با ترکیب نمونه‌گیری و تجمیع، GraphSAGE می‌تواند مانند شبکه‌های عصبی عمیق معمولی روی دسته های کوچک[۵] آموزش ببیند، که این امر آن را به مدلی بسیار مقیاس‌پذیر تبدیل می‌کند.

نتایج کلیدی: Hamilton et al نشان دادند که GraphSAGE در وظایف طبقه‌بندی القایی گره عملکرد بالایی دارد. به عنوان مثال، این مدل توانست نقش گره‌ها را در شبکه‌های استنادی در حال تکامل به درستی طبقه‌بندی کند و حتی به طور کامل به داده‌های گرافی جدید تعمیم یابد. در یکی از آزمایش‌ها، GraphSAGE قادر بود عملکرد پروتئین‌ها را در یک گراف تعامل پروتئین-پروتئین که در طول آموزش دیده نشده بود، پیش‌بینی کند. این مدل روش‌های انتقالی و سایر روش‌های القایی را در این وظایف شکست داد.

اهمیت: GraphSAGE تأثیر گسترده‌ای داشته است، به ویژه در کاربردهای صنعتی یادگیری ماشین روی گراف. استراتژی نمونه‌گیری آن به عنوان الگوی بسیاری از مدل‌های مقیاس‌پذیر بعدی GNN مورد استفاده قرار گرفت. همچنین، پارادایم یادگیری القایی آن بر سیستم‌هایی مانند PinSage (سیستم پیشنهاددهی Pinterest) و سایر برنامه‌های گرافی در مقیاس وب تأثیر گذاشت. به طور کلی، GraphSAGE نشان داد که چگونه می‌توان GNNها را روی گراف‌های بزرگ آموزش داد و آن‌ها را به داده‌های جدید تعمیم بخشید. این مدل با یادگیری روش تجمیع اطلاعات همسایگان، پیشرفت چشمگیری در مقیاس‌پذیری عملی GNNها ایجاد کرد.

[5] mini-batch

۳-۴ شبکه‌های عصبی پیام‌رسانی (MPNN) توسط Gilmer [۱]و همکاران در سال ۲۰۱۷

در کار آن‌ها با عنوان "Neural Message Passing for Quantum Chemistry"، چارچوب شبکه‌های عصبی پیام‌رسانی[۶] ارائه شد که بسیاری از گونه‌های موجود شبکه‌های عصبی گرافی را تحت یک قالب مشترک یکپارچه کرد. نویسندگان مشاهده کردند که بسیاری از روش‌های شبکه‌های عصبی گرافی در آن زمان، از جمله رویکردهای طیفی و فضایی را می‌توان به صورت گره‌هایی که با همسایگان خود پیام رد و بدل می‌کنند توصیف کرد. آن‌ها این ایده را به یک فرآیند دو مرحله‌ای در هر تکرار ساده کردند. در مرحله پیام‌رسانی، هر گره اطلاعاتی را از همسایگان خود دریافت و جمع‌آوری می‌کند. در مرحله خوانش، حالات گره‌ها برای محاسبه خروجی، مانند پیش‌بینی در سطح گراف، مورد استفاده قرار می‌گیرند. در شبکه‌های عصبی پیام‌رسانی، هر یال (u,v) می‌تواند یک تابع پیام‌رسانی یادگرفتنی M(hu,hv) داشته باشد که پیامی از گره u به v تولید می‌کند. علاوه بر این، هر گره دارای یک تابع به‌روزرسانی است که پیام‌های دریافتی را ترکیب کرده و وضعیت پنهان خود را تغییر می‌دهد. با انتخاب فرم‌های مختلف برای این توابع پیام‌رسانی و به‌روزرسانی، می‌توان بسیاری از مدل‌های شناخته‌شده GNN را به عنوان حالات خاص این چارچوب به دست آورد.

مشارکت های اصلی: این چارچوب یک زبان مشترک برای توصیف شبکه‌های عصبی گرافی فراهم کرد که مشخص می‌کرد مدل‌های قبلی چگونه به یکدیگر مرتبط هستند و چه تفاوت‌هایی دارند. نکته مهم این است که Gilmer و همکاران تنها به ارائه نظریه بسنده نکردند، بلکه شبکه‌های عصبی پیام‌رسانی را در وظایف دنیای واقعی در شیمی کوانتومی به کار بردند. در آزمایش‌های خود، آن‌ها یک شبکه عصبی پیام‌رسانی را برای پیش‌بینی ویژگی‌های مولکولی، مانند انرژی اتمیزه شدن مولکول‌ها، روی مجموعه داده QM9 آموزش دادند، جایی که مولکول‌ها به طور طبیعی به صورت گراف‌هایی از اتم‌ها نمایش داده می‌شوند. این مدل در پیش‌بینی ویژگی‌های مولکولی به نتایج پیشرفته‌ای دست یافت و رویکردهای پیشین را با اختلاف قابل توجهی پشت سر گذاشت.

نتایج کلیدی: این نتیجه نشان داد که یادگیری یک تابع پیام‌رسانی روی گراف‌های مولکولی می‌تواند تعاملات پیچیده کوانتومی را بهتر از توصیف‌گرهای طراحی‌شده به‌صورت دستی مدل‌سازی کند. آن‌ها همچنین در چارچوب شبکه‌های عصبی پیام‌رسانی، تغییرات جدیدی را بررسی کردند، از جمله آزمایش با توابع پیام‌رسانی و خوانش مختلف، تا عملکرد مدل را به میزان بیشتری بهبود بخشند.

[6] MPNN

اهمیت: شبکه عصبی پیام‌رسانی به مدل مفهومی استاندارد برای GNNها تبدیل شده است، به طوری که بسیاری از مقالات بعدی از اصطلاحات و چارچوب پیام‌رسانی معرفی‌شده در این کار استفاده کرده‌اند. این پژوهش با فرمالیزه کردن GNNها، به محققان در طراحی معماری‌های جدید کمک کرد، به گونه‌ای که امکان ترکیب و تغییر توابع پیام‌رسانی و به‌روزرسانی به شکلی اصولی فراهم شد. علاوه بر این، موفقیت در حوزه شیمی، قدرت GNNها را در یادگیری روی داده‌های ساختاریافته به‌صورت گراف نشان داد و الهام‌بخش پژوهش‌های متعددی در زمینه گراف‌های مولکولی، کشف دارو و علم مواد شد که رویکرد شبکه‌های عصبی پیام‌رسانی را گسترش دادند. به طور خلاصه، کار Gilmer و همکاران (۲۰۱۷) هم یک یکپارچه‌سازی نظری از GNNها ارائه داد و هم کاربرد عملی آن‌ها را در یک حوزه چالش‌برانگیز نشان داد.

۳-۵ شبکه توجه گرافی (GAT) توسط Veličković و همکاران [۶] (۲۰۱۸)

شبکه توجه گرافی[۷] یک مکانیزم توجه جدید را به شبکه‌های عصبی گرافی معرفی کرد که به مدل امکان می‌دهد در فرآیند تجمیع، وزن‌های متفاوتی را برای همسایگان مختلف یاد بگیرد. پیش از GAT، بیشتر روش‌های گرافی فضایی مانند GCN و GraphSAGE تمام همسایگان را به‌طور یکنواخت در نظر می‌گرفتند یا از نرمال‌سازی ساده‌ای مانند میانگین‌گیری یا وزن‌دهی بر اساس درجه گره استفاده می‌کردند. GAT این روند را تغییر داد و یک استراتژی خودتوجهی [۸] را به کار گرفت.

مشارکت های اصلی: برای هر گره، یک شبکه عصبی کوچک، یک ضریب توجه را برای ویژگی‌های هر همسایه محاسبه می‌کند که نشان می‌دهد آن همسایه تا چه حد در به‌روزرسانی گره اهمیت دارد. سپس این ضرایب یادگرفته‌شده قبل از جمع‌بندی، ویژگی‌های همسایگان را مقیاس‌بندی می‌کنند. این روش به GAT امکان می‌دهد تا روی مهم‌ترین بخش‌های زمینه محلی یک گره در گراف تمرکز کند. یکی از ویژگی‌های کلیدی این مکانیزم توجه این است که انطباق‌پذیر و داده‌محور است. مدل می‌تواند به همسایه‌ای که اطلاعات مهمی را به اشتراک می‌گذارد وزن بیشتری اختصاص دهد و به همسایه‌ای که ارتباط کمتری دارد وزن کمتری بدهد، بدون تکیه‌ی صرف بر ساختار گراف یا درجه گره. Veličković

[7] GAT
[8] self-attention

و همکاران همچنین از توجه چندسری[9] استفاده کردند که در آن چندین مکانیزم توجه مستقل اعمال شده و خروجی‌های آن‌ها با یکدیگر ادغام یا میانگین‌گیری می‌شوند. این کار یادگیری را پایدارتر کرده و نمایش ویژگی‌ها را غنی‌تر می‌سازد، مشابه توجه چندسری در مدل‌های مبدل[10].

نتایج کلیدی: GAT با استفاده از تجمیع مبتنی بر توجه، ویژگی نامتغیر بودن نسبت به جایگشت را حفظ می‌کند و می‌تواند گراف‌هایی با تعداد همسایگان متغیر را مدیریت کند، در حالی که سربار محاسباتی کمی را معرفی می‌کند (تنها چند لایه خطی برای محاسبه ضرایب توجه). این روش نیاز به عملیات پرهزینه‌ای مانند محاسبه ویژه‌بردارها[11] (که در روش‌های طیفی مورد نیاز بود) را از بین برد. ضرایب توجه به‌طور مستقیم در فضای ویژگی گره‌ها محاسبه می‌شوند، که این روش را کاملاً فضایی و قابل موازی‌سازی می‌کند. این مزیت‌ها منجر به نتایج تجربی قوی شد. GAT عملکرد بهتری نسبت به مدل کلاسیک GCN در معیارهای استاندارد رده‌بندی گره به صورت انتقالی (Cora، Citeseer، Pubmed) داشت. همچنین در یک تنظیم القایی روی مجموعه داده برهم‌کنش پروتئین-پروتئین [12] دستاوردهای قابل توجهی به دست آورد، جایی که مدل باید به گراف‌های جدید تعمیم می‌یافت. این نتایج نشان داد که یادگیری نحوه‌ی تخصیص توجه در گراف‌ها می‌تواند دقت مدل را بهبود بخشد. علاوه بر این، GAT امکان تفسیرپذیری را فراهم می‌کند. از طریق ضرایب توجه، می‌توان بررسی کرد که یک گره کدام همسایگان را برای پیش‌بینی خود مهم‌تر در نظر گرفته است.

اهمیت: معرفی مکانیزم توجه راه را برای توسعه‌ی بسیاری از گونه‌های جدید GNN باز کرد. GAT به‌سرعت برای گراف‌های ناهمگن گسترش یافت، جایی که می‌توان توجه را بر انواع مختلف روابط یا انواع گره‌ها اعمال کرد. همچنین در وظایفی مانند پیش‌بینی یال و رده‌بندی گراف مورد استفاده قرار گرفت. GAT به معماری محبوبی برای کاربردهایی تبدیل شد که در آن‌ها گراف دارای نویز است یا برخی اتصالات از سایرین مهم‌تر هستند. به عنوان مثال، در شبکه‌های استنادی که همه‌ی استنادات اهمیت یکسانی ندارند یا در گراف‌های دانش[13] که برخی روابط وزن بیشتری دارند، GAT برتری خود را نشان داده است. به طور کلی، شبکه‌های توجه گرافی با یادگیری وزن‌های پویا و غیرثابت به جای میانگین‌گیری ساده‌ی همسایگان، به پیشرفت چشمگیری در یادگیری نمایش‌های گرافی منجر شدند و مرزهای روش‌های پیشرفته را گسترش دادند.

[9] multi-head attention
[10] Transformer
[11] eigenvectors
[12] PPI
[13] knowledge graphs

۳-۶ شبکه هم‌ریختی گراف (GIN) توسط Xu و همکاران[۹] (۲۰۱۹)

شبکه هم‌ریختی گراف [۱۴] که توسط Xu و همکاران در سال ۲۰۱۹ در مقاله‌ی "How Powerful are Graph Neural Networks" معرفی شد، یک GNN با دقت نظری بالا است که برای حداکثرسازی قدرت نمایشی طراحی شده است. مهم‌ترین دستاورد GIN برقراری ارتباط روشن میان GNNها و آزمون هم‌ریختی گراف Weisfeiler-Lehman (WL) است. آزمون WL یک الگوریتم کلاسیک برای تشخیص تفاوت‌های ساختاری میان دو گراف است. Xu و همکاران نشان دادند که GNNهای استاندارد مبتنی بر پیام‌رسانی (مانند GCN و GraphSAGE) در تمایز برخی ساختارهای گرافی دچار محدودیت هستند. در واقع، آن‌ها اثبات کردند که GNNها در بهترین حالت، قدرت تفکیکی برابر با آزمون WL تک‌بعدی دارند و نمی‌توانند گراف‌های غیرهم‌ریخت را بهتر از این آزمون تمایز دهند. با این بینش، آن‌ها معماری GIN را معرفی کردند که از نظر نظری، حداکثر قدرت تفکیک‌پذیری ممکن را برای یک شبکه مبتنی بر پیام‌رسانی به دست می‌آورد و قدرت تفکیکی آن با آزمون WL برابر است.

مشارکت های اصلی: GIN از یک طرح ساده برای تجمیع همسایگان استفاده می‌کند. در این روش، هر گره ویژگی‌های همسایگان خود را با هم جمع می‌کند و در صورت لزوم، یک مقدار ثابت کوچک ϵ را به ویژگی خود اضافه می‌کند. سپس این مجموع از طریق یک پرسپترون چندلایه[۱۵] عبور داده می‌شود تا نمایش جدید گره تولید شود. انتخاب تجمیع مبتنی بر جمع[۱۶] نقش کلیدی دارد. برخلاف میانگین‌گیری، جمع می‌تواند چندمجموعه مختلف از ویژگی‌های همسایگان را از هم متمایز کند (مشروط بر اینکه پرسپترون چندلایه به اندازه‌ی کافی قدرتمند باشد). این خاصیت باعث می‌شود که تابع تجمیع GIN تزریقی[۱۷] باشد، یعنی بتواند توزیع‌های مختلف ویژگی‌های همسایگان را از هم تفکیک کند. همین ویژگی است که باعث شده GIN از نظر نظری به اندازه‌ی آزمون WL در تمایز ساختارهای گرافی قدرتمند باشد.

نتایج کلیدی: مطابق با طراحی خود عمل کرد و در بسیاری از معیارهای رده‌بندی گراف شامل شبکه‌های اجتماعی و داده‌های مولکولی بیوشیمیایی، عملکردی برتر یا هم‌سطح با بهترین روش‌های موجود در آن زمان داشت. نویسندگان نشان دادند که سایر روش‌های تجمیع مانند میانگین و حداکثر

[14] GIN
[15] MLP
[16] sum aggregator
[17] injective

قدرت تفکیک کمتری دارند. آن‌ها نمونه‌هایی از گراف‌های غیرهم‌ریخت ساختند که این تجمیع‌کننده‌ها قادر به تمایز آن‌ها نبودند، اما GIN توانست آن‌ها را تفکیک کند. سادگی GIN (جمع کردن ویژگی‌ها و عبور از یک پرسپترون چندلایه) باعث شد که این مدل به یک مبنای قوی و استاندارد جدید برای وظایف رده‌بندی گراف تبدیل شود.

اهمیت: این کار یک دیدگاه نظری برای طراحی GNNها ارائه داد و محققان را تشویق کرد تا به محدودیت‌های نمایشی GNNها توجه کنند. همچنین باعث علاقه‌مندی به توسعه مدل‌هایی فراتر از محدودیت‌های آزمون WL شد، چرا که GIN این حد را اشباع کرده بود. GIN تأثیر قابل‌توجهی در طراحی مدل‌های عملی داشت. برای مثال، ایده‌ی استفاده از تجمیع مبتنی بر جمع یا به‌کارگیری پارامترهای یادگرفتنی ϵ برای تنظیم اهمیت اطلاعات خود گره، بعدها در مدل‌های دیگر مورد استفاده قرار گرفت. به طور خلاصه، شبکه‌های هم‌ریختی گراف شکاف نظری را پر کردند و نشان دادند که چگونه می‌توان یک GNN طراحی کرد که از نظر نظری، حداکثر قدرت ممکن را برای تشخیص ساختار گرافی داشته باشد. علاوه بر این، GIN به دلیل عملکرد عالی در معیارهای رده‌بندی گراف، جایگاه خود را در جامعه‌ی یادگیری گرافی تثبیت کرد.

۳-۷ ساده‌سازی GCN (SGC) توسط Wu و همکاران [۷] (۲۰۱۹)

Wu و همکاران (۲۰۱۹) در مقاله‌ای، شبکه‌های گراف کانولوشنی ساده‌شده[۱۸] را به عنوان یک تغییر حداقلی در معماری کلاسیک GCN معرفی کردند که مزایای قابل توجهی را ارائه می‌دهد. نویسندگان مشاهده کردند که توابع فعال‌سازی غیرخطی بین لایه‌های GCN ممکن است برای عملکرد رده‌بندی نیمه‌نظارتی گره‌ها ضروری نباشند. در یک GCN استاندارد، لایه‌های متعددی روی هم قرار می‌گیرند که هر کدام شامل یک ماتریس وزن، نرمال‌سازی مجاورت و تابع فعال‌سازی ReLU است. SGC این ساختار را ساده می‌کند و به‌تدریج توابع غیرخطی را حذف کرده و ماتریس‌های وزن را در لایه‌ها ادغام می‌کند. در عمل، SGC یک GCN چندلایه را به یک تبدیل خطی واحد تبدیل می‌کند که پس از K تکرار از توان ماتریس مجاورت نرمال‌شده اعمال می‌شود. از نظر مفهومی، SGC تجمیع همسایگان را از پیش محاسبه می‌کند. این کار با به توان رساندن ماتریس مجاورت تا مرتبه K و ضرب آن در ویژگی‌ها انجام می‌شود، که در نتیجه ویژگی‌های هر گره را در محدوده K-همسایگی آن انتشار می‌دهد. پس از این مرحله، مدل تنها نیاز دارد که یک رده‌بند خطی را روی ویژگی‌های تجمیع‌شده یاد بگیرد.

[18] SGC

مشارکت های اصلی: نکته قابل توجه این است که این ساده‌سازی چشمگیر، دقت را در بسیاری از معیارها به‌طور محسوسی کاهش نمی‌دهد. Wu و همکاران نشان دادند که در مجموعه داده‌های شبکه‌های استنادی (Cora، Citeseer، Pubmed)، یک SGC با تعداد مناسبی از گام‌های همسایگی می‌تواند دقتی مشابه با یک GCN کامل ارائه دهد، با این که هیچ لایه پنهان و هیچ تابع غیرخطی ندارد. آن‌ها یک توضیح نظری ارائه کردند: حذف غیرخطیت‌ها، مدل را به یک فیلتر پایین‌گذر ثابت روی گراف تبدیل می‌کند، به این معنا که مدل به‌طور صریح ویژگی‌ها را روی گراف صاف می‌کند. این صاف‌سازی ویژگی‌ها، همراه با یک لایه خطی نهایی، برای بسیاری از وظایف با میزان بالای هموفیلی گراف (شبکه‌هایی که گره‌های متصل ویژگی‌های مشابه دارند) کافی است. علاوه بر این، ادغام ماتریس‌های وزن در SGC باعث کاهش تعداد پارامترها و کاهش خطر بیش‌برازش[۱۹] می‌شود.

نتایج کلیدی: SGC سرعت آموزش و استنتاج بسیار بالاتری دارد. بدون نیاز به ضرب‌های متعدد در ماتریس‌های وزن و پس‌انتشار پرهزینه از طریق چندین لایه، SGC نسبت به GCN تا چندین برابر سریع‌تر اجرا می‌شود، به‌ویژه روی گراف‌های بزرگ. همچنین، بهینه‌سازی آن آسان‌تر است، چرا که اساساً به یک رگرسیون لجستیک روی ویژگی‌های صاف‌شده تقلیل می‌یابد. افزون بر این، تفسیرپذیری بیشتری دارد، زیرا می‌توان به‌طور مستقیم اثر انتشار ویژگی‌ها در K-همسایگی را درک کرد.

اهمیت: SGC یک یادآوری مهم بود که پیچیدگی همیشه ضروری نیست. این مدل به یک مبنای قوی برای وظایف یادگیری گرافی تبدیل شد، به طوری که هر GNN جدیدی برای اثبات برتری خود، باید نشان دهد که علاوه بر GCN، از SGC نیز عملکرد بهتری دارد. علاوه بر این، SGC تحقیقات بعدی را در زمینه‌ی جداسازی مرحله‌ی انتشار ویژگی از مرحله‌ی پیش‌بینی الهام بخشید، که منجر به توسعه معماری‌های GNN تفکیک‌شده [۲۰] شد. به طور خلاصه، مقاله Simplifying GCN نشان داد که با حذف پیچیدگی‌های اضافی و بازگشت به هسته‌ی خطی GCN، می‌توان دقت بالایی را حفظ کرد و در عین حال کارایی را به طور قابل توجهی بهبود بخشید. این پژوهش هم از نظر نظری و هم به عنوان یک مبنای عملی برای یادگیری گرافی، تأثیرگذار بود.

۳-۸ LightGCN توسط He و همکاران [۳] (۲۰۲۰)

LightGCN یک مدل GNN است که برای سیستم‌های توصیه‌ای طراحی شده و به طور خاص برای

[19] overfitting
[20] Decoupled GNNs

ساده‌سازی GCNها از طریق حذف اجزای غیرضروری در فیلترینگ مشارکتی ساخته شده است.

مشارکت های اصلی: GCN های سنتی شامل ماتریس‌های تبدیل ویژگی و فعال‌سازی غیرخطی در هر لایه هستند، اما در سناریوهای توصیه کاربر-آیتم، گره‌ها اغلب با تعبیه‌های ID نمایش داده می‌شوند (بدون ویژگی‌های پیچیده)، و حفظ تعاملات خطی ممکن است کافی باشد. نویسندگان با انجام مطالعات حذف اجزا دریافتند که دو ماژول رایج در GCN – تبدیل ویژگی و فعال‌سازی غیرخطی – تأثیر کمی دارند یا حتی عملکرد را در سیستم‌های توصیه‌ای کاهش می‌دهند. بنابراین، LightGCN تنها بخش ضروری را حفظ می‌کند: تجمیع همسایگان در گراف تعامل کاربر-آیتم. در LightGCN، هر کاربر یا آیتم با یک بردار تعبیه نمایش داده می‌شود. سپس مدل این تعبیه‌ها را با انتشار در گراف کاربر-آیتم بهبود می‌دهد: بردار تعبیه هر کاربر به‌عنوان میانگین بردارهای تعبیه آیتم‌های همسایه‌اش به‌روزرسانی می‌شود (و بالعکس، بردار هر آیتم از طریق کاربران متصل به آن آپدیت می‌شود). این انتشار در چندین لایه انجام می‌شود تا اطلاعات چندگامی همسایگی جمع‌آوری شود. نکته مهم این است که LightGCN هیچ تابع غیرخطی یا ماتریس وزنی در لایه‌های میانی ندارد و انتشار تنها یک مجموع وزن‌دار ساده است. همچنین، حلقه‌های خودی[21] از ماتریس مجاورت حذف شده‌اند. در عوض، پس از K لایه انتشار، مدل بردارهای تعبیه‌شده از تمام لایه‌ها، از جمله لایه اولیه، را به‌صورت یک مجموع وزن‌دار ترکیب می‌کند تا نمایش نهایی را از ترکیب اطلاعات ۱-همسایگی، ۲-همسایگی، ...، K-همسایگی به دست آورد.

نتایج کلیدی: با وجود سادگی، LightGCN عملکرد بهتری در توصیه‌ها نسبت به مدل‌های پیچیده‌تر مبتنی بر GCN نشان داد. در آزمایش‌های انجام‌شده روی مجموعه داده‌های معیار (مانند داده‌های امتیازدهی فیلم یا محصولات)، LightGCN به‌طور میانگین حدود ۱۶ درصد بهتر از مدل Neural Graph Collaborative Filtering (NGCF) در معیارهای بازخوانی و دقت عمل کرد، که یک پیشرفت چشمگیر در سیستم‌های توصیه‌ای محسوب می‌شود. این بهبود در حالی حاصل شد که LightGCN ساده‌تر آموزش داده می‌شود (تعداد پارامترهای کمتر، عدم نیاز به تنظیم لایه‌های عمیق غیرخطی) و سرعت استنتاج بیشتری دارد. همچنین، نویسندگان تحلیلی ارائه کردند که چرا این مدل ساده عملکرد خوبی دارد: حذف توابع غیرخطی از بیش‌صاف‌شدن[22] نمایش‌ها و بیش‌برازش جلوگیری می‌کند، که در فیلترینگ مشارکتی، به دلیل پراکندگی داده‌ها، یک مشکل جدی است.

اهمیت: LightGCN به استاندارد جدیدی در سیستم‌های توصیه‌ای مبتنی بر گراف تبدیل شد. این مدل نشان داد که برای گراف‌های کاربر-آیتم، یک رویکرد ساده مبتنی بر تجمیع همسایگان بسیار مؤثر

[21] self-loops

[22] over-smoothing

است. بسیاری از مدل‌های توصیه‌ای بعدی فلسفه‌ی LightGCN را پذیرفتند و این کار بحث‌هایی را در مورد "چه زمانی کمتر، بهتر است" در طراحی GNN برانگیخت. به طور خلاصه، He و همکاران نشان دادند که با کاهش GCN به مکانیزم اصلی انتشار و سفارشی‌سازی آن برای ساختار مسائل توصیه‌ای، می‌توان با مدلی بسیار ساده‌تر به نتایج پیشرفته در سیستم‌های توصیه‌ای دست یافت.

۳-۹ یک بررسی جامع از GNNها توسط Wu و همکاران[۸] (۲۰۲۰)

Wu و همکاران (۲۰۲۰) یک بررسی گسترده تحت عنوان "A Comprehensive Survey on Graph Neural Networks" انجام دادند که به عنوان یک منبع ارزشمند برای خلاصه‌سازی و سازمان‌دهی تحقیقات در حال رشد سریع در زمینه‌ی GNNها عمل می‌کند.

مشارکت های اصلی: مهم‌ترین مشارکت‌های این بررسی شامل طبقه‌بندی جدیدی برای دسته‌بندی مدل‌های GNN، مرور دقیق صدها مقاله منتشرشده، و ارائه‌ی یک نمای کلی از کاربردهای رایج و چالش‌های موجود در این حوزه است. نویسندگان روش‌های GNN را به چهار دسته کلی تقسیم می‌کنند:

۱. GNNهای بازگشتی[۲۳]، که شامل مدل‌های اولیه مانند مدل Scarselli است که از پیام‌رسانی بازگشتی استفاده می‌کنند.

۲. GNNهای کانولوشنی[۲۴]، که روش‌های مبتنی بر کانولوشن طیفی و فضایی را پوشش می‌دهند (مانند GCN، GraphSAGE و GAT که در این دسته قرار می‌گیرند).

۳. خودرمزگذارهای گرافی[۲۵]، برای یادگیری بدون نظارت و تولید تعبیه‌ها (از جمله روش‌هایی مانند خودرمزگذارهای گرافی متغیر).

۴. GNNهای فضایی-زمانی[۲۶]، که برای مدل‌سازی گراف‌های پویا یا داده‌های سری‌زمانی روی ساختارهای گرافی به کار می‌روند (اهمیت ویژه‌ای برای شبکه‌های حمل‌ونقل و سیستم‌های زمانی دارند).

[23] Recurrent GNNs
[24] Convolutional GNNs
[25] Graph Autoencoders
[26] Spatial-temporal GNNs

با تعریف این دسته‌بندی‌ها، این بررسی به درک بهتر چشم‌انداز تحقیقات GNN کمک می‌کند و نشان می‌دهد که مدل‌های مختلف چگونه به هم مرتبط هستند. برای هر دسته، این پژوهش توصیف‌های دقیقی از مدل‌های نماینده، مقایسه‌ای از معماری‌های آن‌ها، و جداولی که ویژگی‌ها و عملکردشان را خلاصه می‌کند ارائه می‌دهد. این سطح از سازمان‌دهی بسیار ضروری بود، زیرا در آن زمان رشد چشمگیری در تعداد مقالات مربوط به GNNها مشاهده می‌شد. فراتر از طبقه‌بندی مدل‌ها، Wu و همکاران کاربردهای GNNها را در حوزه‌های مختلف خلاصه می‌کنند. آن‌ها بررسی می‌کنند که چگونه GNNها در تحلیل شبکه‌های اجتماعی، بینایی کامپیوتری (مانند گراف‌های صحنه)، پردازش زبان طبیعی (برای درخت‌های وابستگی یا گراف‌های دانش)، شیمی و زیست‌شناسی (گراف‌های مولکولی، شبکه‌های تعامل پروتئین) و بسیاری از زمینه‌های دیگر به کار رفته‌اند. این بررسی انعطاف‌پذیری و گستردگی کاربردهای شبکه‌های عصبی گرافی را برجسته می‌کند. نویسندگان مجموعه داده‌های معیار پرکاربرد را معرفی کرده و حتی پیاده‌سازی‌های متن‌باز را فهرست می‌کنند، که این بررسی را به راهنمایی عملی برای پژوهشگران و متخصصان تبدیل می‌کند.

چالش های کنونی و مسیرهای آینده: بخشی مهم از این بررسی بحث در مورد چالش‌های فعلی و مسیرهای آینده تحقیقات GNN است. Wu و همکاران مسائل کلیدی را که در آن زمان (و حتی در برخی موارد همچنان) به‌عنوان مشکلات حل‌نشده در تحقیق روی GNN مطرح بودند، شناسایی کردند:

- چگونه می‌توان GNNهای بسیار عمیق‌تری ساخت بدون مواجهه با مشکل بیش‌صاف‌شدن
- چگونه مقیاس‌پذیری را برای پردازش گراف‌های در مقیاس وب بهبود داد
- چگونه با گراف‌های ناهمگن شامل چندین نوع گره/یال کار کرد
- چگونه می‌توان گراف‌های پویا یا در حال تغییر را به‌صورت بلادرنگ مدیریت کرد

برای هر چالش، نویسندگان راه‌حل‌های ممکن یا مسیرهای پژوهشی را معرفی می‌کنند. به عنوان مثال، آن‌ها کارهایی را که از اتصالات جهشی[۲۷] یا نرمال‌سازی برای مقابله با بیش‌صاف‌شدن استفاده می‌کنند، و تکنیک‌های نمونه‌گیری گراف برای بهبود مقیاس‌پذیری را بررسی کرده‌اند.

نتایج: Wu و همکاران تصویری جامع از وضعیت GNNها ارائه دادند و تحولات متعدد این حوزه را در یک ساختار منسجم سازمان‌دهی کردند. علاوه بر این، با مشخص کردن مسیرهای تحقیقاتی آینده، راه را برای پیشرفت‌های بیشتر در یادگیری گرافی باز کردند.

[27] skip connections

اهمیت: این بررسی به‌سرعت به مرجع اصلی برای پژوهشگران تازه‌وارد و متخصصان یادگیری گرافی تبدیل شد. با پوشش و دسته‌بندی بیش از ۲۰۰ مقاله، درک وضعیت فعلی GNNها را برای محققان تسهیل کرد و به آن‌ها در تشخیص روندها و شکاف‌های تحقیقاتی کمک نمود. به عنوان مثال، این بررسی نشان داد که چگونه روش‌های طیفی به تدریج به سمت روش‌های فضایی حرکت کردند و چگونه مکانیزم‌های توجه در GNNها رشد کردند. طبقه‌بندی و بینش‌های ارائه‌شده در این پژوهش، تحقیقات بعدی را هدایت کرده است. بسیاری از مقالات جدید، مشارکت‌های خود را در چارچوب دسته‌بندی Wu و همکاران قرار می‌دهند و بر مسیرهای تحقیقاتی مشخص‌شده تمرکز دارند. برای نمونه، امروزه تحقیقات گسترده‌ای روی GNNهای عمیق و GNNهای پویا انجام شده است، که دقیقاً به چالش‌هایی پاسخ می‌دهد که در سال ۲۰۲۰ مطرح شده بودند.

۳-۱۰ شبکه‌های عصبی گرافی: یک مرور توسط Zhou و همکاران [۱۰] (۲۰۲۰)

در همان بازه‌ی زمانی، Zhou و همکاران مقاله‌ی "Graph Neural Networks: A Review of Methods and Applications" را منتشر کردند که یکی دیگر از بررسی‌های جامع و تأثیرگذار در حوزه‌ی GNNها است. این بررسی نه تنها مدل‌های مختلف GNN را پوشش می‌دهد، بلکه بر اصول طراحی و کاربردهای عملی شبکه‌های عصبی گرافی نیز تأکید دارد.

مشارکت های اصلی: Zhou و همکاران یک جریان کلی برای طراحی GNNها پیشنهاد کردند و یک شبکه‌ی عصبی گرافی معمولی را به اجزای مختلفی تجزیه کردند، از جمله کدگذاری ورودی ویژگی‌های گراف، مکانیزم انتشار (پیام‌رسانی)، و لایه‌ی خروجی/خوانش[۲۸] برای پیش‌بینی. با این کار، آن‌ها چارچوبی ارائه دادند که در آن می‌توان تحلیل کرد که مدل‌های مختلف GNN چگونه هر یک از این اجزا را پیاده‌سازی می‌کنند. برای مثال، در مکانیزم انتشار[۲۹]، برخی مدل‌ها از توجه استفاده می‌کنند، در حالی که برخی دیگر از مکانیزم دروازه‌ای[۳۰] بهره می‌برند. همچنین، برخی مدل‌ها برای خروجی در سطح گراف از ادغام[۳۱] استفاده می‌کنند، در حالی که برخی دیگر از مجموع‌گیری بهره می‌برند. این بررسی به‌طور سیستماتیک این انتخاب‌ها را مقایسه می‌کند و نشان می‌دهد که چگونه هر مدل اجزای خود را

[28] readout
[29] propagation
[30] gating mechanism
[31] pooling

طراحی می‌کند. علاوه بر این، Zhou و همکاران طیف گسترده‌ای از گونه‌های GNN را بررسی کردند، از جمله شبکه‌های گراف کانولوشنی، شبکه‌های گراف مبتنی بر توجه، و شبکه‌های گراف دروازه‌ای[۳۲]. این بررسی از نظر پوشش مدل‌ها مشابه بررسی Wu و همکاران است، اما از منظر طراحی یک جریان کلی برای تحلیل مدل‌ها استفاده می‌کند. یکی از جنبه‌های برجسته این بررسی، پوشش گسترده‌ی کاربردهای GNN است. Zhou و همکاران کاربردهای شبکه‌های عصبی گرافی را در حوزه‌هایی از جمله شبکه‌های اجتماعی، گراف‌های دانش، سیستم‌های توصیه‌ای، بینایی کامپیوتری (گراف‌های صحنه)، پردازش زبان طبیعی (گراف‌های نحوی و معنایی)، شیمی و زیست‌شناسی (گراف‌های مولکولی، ساختارهای پروتئینی) و بسیاری زمینه‌های دیگر دسته‌بندی کرده‌اند. برای هر حوزه، آن‌ها توضیح می‌دهند که چگونه از GNNها استفاده می‌شود و نمونه‌هایی از موارد موفق ارائه می‌کنند. این کار به محققان در این حوزه‌ها کمک کرد تا ارتباط GNNها را با مسائل خود بهتر درک کنند.

چالش های کنونی و مسیرهای آینده: این بررسی همچنین بینش‌هایی درباره مسیرهای آینده تحقیقات ارائه می‌دهد. به طور خاص، Zhou و همکاران چهار چالش مهم را که محققان GNN باید به آن‌ها بپردازند، برجسته می‌کنند:

- مشکل بیش‌صاف‌شدن: زمانی که لایه‌های GNN عمیق‌تر می‌شوند، نمایش گره‌ها بیش از حد شبیه به هم شده و عملکرد مدل کاهش می‌یابد.
- مسئله مقیاس‌پذیری: بسیاری از GNNها در پردازش گراف‌های بسیار بزرگ به دلیل محدودیت‌های محاسباتی و حافظه‌ای دچار مشکل می‌شوند.
- مدیریت گراف‌های پویا: گراف‌هایی که در طول زمان تغییر می‌کنند یا دارای یال‌های موقتی هستند، نیازمند GNNهایی هستند که بتوانند نمایش‌ها را به‌صورت آنی به‌روزرسانی کنند.
- ادغام داده‌های غیرساختاریافته با گراف‌ها: مانند ترکیب متن یا تصاویر (داده‌های حسی) با ساختارهای گرافی در یک مدل یکپارچه.

نویسندگان اشاره می‌کنند که تا سال ۲۰۲۰، روش‌های کاملاً مؤثری برای برخی از این جنبه‌ها، مانند یادگیری گراف‌های پویا، وجود نداشت و خواستار تلاش‌های پژوهشی بیشتر برای حل این چالش‌ها شدند.

نتایج: در مجموع، Zhou و همکاران یک راهنمای جامع و عمیق درباره شبکه‌های عصبی گرافی ارائه دادند که طیفی از طراحی مدل‌های نظری تا کاربردهای عملی را پوشش می‌دهد. با مشخص کردن

[32] Gated Graph Networks

مسائل حل‌نشده‌ی کلیدی مانند عمق، مقیاس‌پذیری، پویایی و ادغام داده‌ها، این پژوهش نقش مهمی در شکل‌گیری مسیرهای آینده‌ی تحقیقاتی در حوزه‌ی GNNها داشته است.

اهمیت: بررسی Zhou و همکاران به‌طور گسترده‌ای استناد شده و به عنوان یک راهنمای کلیدی برای توسعه‌ی الگوریتم‌ها و کاربردهای GNN مورد توجه قرار گرفته است. با فهرست کردن چالش‌ها و پرسش‌های تحقیقاتی باز، این پژوهش مستقیماً بر مسیرهای تحقیقاتی تأثیر گذاشته است. به عنوان مثال، پس از انتشار این مقاله، جامعه علمی توجه ویژه‌ای به GNNهای عمیق‌تر و تکنیک‌های نرمال‌سازی جدید داشت که تا حدی تحت تأثیر مسئله بیش‌صاف‌شدن مطرح‌شده در این بررسی بود. پوشش گسترده‌ی کاربردهای GNN در این مقاله احتمالاً باعث تبادل ایده‌ها بین حوزه‌های مختلف شد و به محققان کمک کرد تا روش‌های GNN را در زمینه‌های جدید کشف کنند. علاوه بر این، ارائه‌ی ساختار منظم از روش‌ها همراه با یک نمودار جریان ساده و قابل‌فهم، این مقاله را به یک منبع آموزشی ارزشمند تبدیل کرد که به متخصصان در انتخاب یا طراحی معماری‌های GNN متناسب با مسائلشان کمک می‌کند.

فصل چهارم

روش شناسی

۴-۱ توضیح پیاده سازی

این فصل توضیحی جامع از پیاده سازی های استفاده‌شده در این پروژه ارائه می‌دهد. هر پیاده سازی مسئول آموزش یک مدل خاص از شبکه عصبی گرافی روی یک مجموعه داده مشخص است. از آنجایی که تمام پیاده سازی ها ساختاری مشابه دارند، یک پیاده سازی را به‌عنوان نمونه‌ی نماینده انتخاب می‌کنیم: آموزش GCN روی مجموعه داده Cora. این توضیح شامل پیش‌پردازش داده‌ها، تعریف مدل، آموزش و ارزیابی خواهد بود.

۴-۱-۱ بارگذاری و پیش‌پردازش داده‌ها

اولین گام در آموزش یک GNN، بارگذاری و پیش‌پردازش مجموعه داده است. مجموعه داده Cora شامل انتشارات علمی به‌عنوان گره‌ها و لینک‌های استناد به‌عنوان یال‌ها است که یک ساختار گرافی را تشکیل می‌دهد.

- ما از Path از کتابخانه pathlib برای تعریف مسیر مجموعه داده استفاده می‌کنیم. این روش انعطاف‌پذیری بالاتری را در سیستم‌عامل‌های مختلف فراهم می‌کند. فایل‌های مجموعه داده از درون این دایرکتوری خوانده خواهند شد.

- فایل cora.content شامل اطلاعات مربوط به هر مقاله است که شامل شناسه منحصربه‌فرد مقاله، یک بردار ویژگی و برچسب دسته‌بندی مقاله می‌شود. تابع np.genfromtxt() این فایل را به‌صورت یک آرایه NumPy با نوع داده رشته‌ای می‌خواند، زیرا داده‌های موجود ترکیبی از مقادیر عددی و متنی هستند.

- ویژگی‌ها در ستون‌های ۱ تا -۱ ذخیره شده‌اند، یعنی اولین ستون (شناسه مقاله) و آخرین ستون (برچسب دسته‌بندی) حذف می‌شوند. ما از csr_matrix برای ذخیره و پردازش بهینه داده‌های پراکنده استفاده می‌کنیم. برچسب‌ها در آخرین ستون مجموعه داده ذخیره شده‌اند.

- برچسب‌های مجموعه داده، متنی هستند (مانند "Neural Networks" و "Genetic Algorithms")، بنابراین هر برچسب منحصربه‌فرد را به یک شاخص عددی نگاشت می‌کنیم. تابع np.unique(labels) نام‌های برچسب‌های منحصربه‌فرد را استخراج کرده و آن‌ها را به ترتیب حروف الفبا مرتب می‌کند. یک دیکشنری (lbl2idx) ایجاد می‌شود که به هر برچسب یک شاخص عددی اختصاص می‌دهد. در نهایت، برچسب‌های متنی اصلی با شاخص‌های عددی جایگزین می‌شوند.

- فایل cora.cites شامل روابط استنادی است که هر سطر آن نشان‌دهنده‌ی یک یال (استناد از یک مقاله به مقاله دیگر) در گراف است. papers شناسه‌های مقالات را از مجموعه داده استخراج می‌کند. یک دیکشنری paper2idx ساخته می‌شود که هر شناسه مقاله را به یک شاخص عددی متناظر نگاشت می‌کند. ماتریس مجاورت با استفاده از coo_matrix ساخته می‌شود که به‌طور کارآمد گراف‌های پراکنده را نمایش می‌دهد

- تابع نرمال سازی، ماتریس داده‌شده را با استفاده از نرمال‌سازی درجه[1] نرمال‌سازی می‌کند.

- نرمال‌سازی ویژگی‌ها باعث می‌شود که مقدار ویژگی‌ها در محدوده‌ی مشابهی قرار گیرد و پایداری آموزش مدل بهبود یابد. eye(adj.shape[0]) حلقه‌های خودی را به گراف اضافه می‌کند، به این معنی که هر گره به خودش متصل می‌شود. حلقه‌های خودی به انتشار اطلاعات در شبکه‌های عصبی گرافی کمک می‌کنند.

- برای آموزش مدل در PyTorch، داده‌ها باید به فرمت تنسور تبدیل شوند:

  adj.todense() ماتریس پراکنده‌ی مجاورت را به فرمت چگال[2] تبدیل می‌کند و سپس به یک تنسور FloatTensor در PyTorch تبدیل می‌شود.

  features.todense() ویژگی‌ها را به فرمت چگال تبدیل کرده و سپس به PyTorch FloatTensor تبدیل می‌کند.

  برچسب‌ها به LongTensor تبدیل می‌شوند، زیرا این نوع داده برای وظایف دسته‌بندی در PyTorch مورد نیاز است.

۴-۱-۲ تعریف مدل

مدل Graph Convolutional Network (GCN) به‌عنوان یک کلاس پیاده‌سازی شده است که فرآیند انتشار رو به جلو[3] را با استفاده از لایه‌های گراف کانولوشنی تعریف می‌کند. هسته‌ی اصلی مدل GCN، لایه GraphConvolution است که پیام‌رسانی را از طریق ساختار گراف انجام می‌دهد.

- torch.nn کلاس‌ها و توابع پایه برای تعریف شبکه‌های عصبی را فراهم می‌کند.
  torch.nn.functional شامل توابع فعال‌سازی رایج مانند ReLU و Softmax است.

[1] degree normalization
[2] Dense
[3] forward propagation

torch.nn.parameter.Parameter برای تعریف پارامترهای قابل‌آموزش مانند وزن‌ها و بایاس استفاده می‌شود.

torch.nn.modules.module.Module کلاس پایه برای تمام مدل‌های PyTorch است.

math برای مقداردهی اولیه وزن‌ها با مقیاس مناسب به‌کار می‌رود.

- کلاس GraphConvolution یک لایه‌ی گراف کانولوشنی را تعریف می‌کند که ویژگی‌ها را در سراسر ساختار گراف منتشر می‌کند. این کلاس از Module ارث‌بری می‌کند که به آن اجازه می‌دهد در ساختار مدل‌های PyTorch قرار گیرد. $in_features$ و $out_features$ تعداد ویژگی‌های ورودی و خروجی را برای لایه مشخص می‌کنند. self.weight یک پارامتر قابل‌آموزش است که ماتریس تبدیل ویژگی گره‌ها را نشان می‌دهد. در صورت فعال بودن bias، یک بردار بایاس (self.bias) مقداردهی می‌شود که به مدل اجازه می‌دهد یک شیفت اضافی را یاد بگیرد. $self.register_parameter('bias', None)$ زمانی استفاده می‌شود که نیازی به بایاس نباشد. $self.reset_parameters()$ مقداردهی اولیه‌ی وزن‌ها و بایاس را انجام می‌دهد.

- مقداردهی اولیه با توزیع یکنواخت انجام می‌شود. اگر بایاس تعریف شده باشد، مقداردهی اولیه‌ی آن نیز با همان توزیع انجام می‌شود. این مقداردهی اولیه از مقدارهای بیش از حد بزرگ (که ممکن است باعث انفجار گرادیان شود) یا بیش از حد کوچک (که باعث کاهش سرعت یادگیری می‌شود) جلوگیری می‌کند.

- مدل GCN چندین لایه‌ی GraphConvolution را برای ایجاد یک شبکه‌ی عصبی گرافی دولایه‌ای ترکیب می‌کند.

nfeat: تعداد ویژگی‌های ورودی (ابعاد بردار ویژگی هر گره).

nhid: تعداد واحدهای مخفی در اولین لایه‌ی GCN.

nclass: تعداد کلاس‌های خروجی (مورد استفاده در دسته‌بندی گره‌ها).

dropout: نرخ دراپ‌اوت برای جلوگیری از بیش‌برازش.

gc1: اولین لایه‌ی گراف کانولوشنی که ویژگی‌های ورودی را به نمایش‌های مخفی تبدیل می‌کند.

gc2: دومین لایه‌ی گراف کانولوشنی که نمایش‌های مخفی را به نمرات دسته‌بندی کلاس‌ها نگاشت می‌کند.

۴-۱-۳ آموزش مدل

در این بخش، مدل GCN را مقداردهی اولیه کرده، یک بهینه‌ساز تنظیم می‌کنیم و برای اطمینان از بازتولیدپذیری، مقداردهی تصادفی را ثابت نگه می‌داریم.

- شبکه‌های عصبی به‌طور تصادفی وزن‌ها را مقداردهی اولیه می‌کنند، که می‌تواند منجر به نتایج متفاوت در هر اجرای مدل شود. $torch.manual_seed(34)$ مقدار تصادفی را ثابت می‌کند تا مقداردهی اولیه‌ی وزن‌ها در اجرای‌های مختلف یکسان باقی بماند. این کار برای اشکال‌زدایی[۴] و مقایسه عملکرد مدل بسیار مهم است.

- مدل GCN را با پارامترهای زیر مقداردهی اولیه می‌کنیم:
  nfeat=$n_features$: تعداد ویژگی‌های ورودی برای هر گره.
  nhid=20: تعداد واحدهای مخفی در اولین لایه‌ی گراف کانولوشنی.
  nclass=n_labels: تعداد کلاس‌های خروجی (تعداد دسته‌های موردنظر برای دسته‌بندی گره‌ها).
  dropout=0.5: احتمال دراپ‌اوت (نیمی از نورون‌ها به‌طور تصادفی در طول آموزش غیرفعال می‌شوند تا از بیش‌برازش جلوگیری شود).

- از بهینه‌ساز Adam استفاده می‌کنیم که یک الگوریتم تطبیقی برای بهینه‌سازی نرخ یادگیری است و از ترکیب روش‌های مومنتوم و RMSprop بهره می‌برد.
  نرخ یادگیری را به‌طور پویا برای پارامترهای مختلف تنظیم می‌کند.
  از مومنتوم استفاده می‌کند که به همگرایی سریع‌تر کمک می‌کند.
  روی گرادیان‌های پراکنده و نویزی عملکرد خوبی دارد، که در GNNها رایج است.

۴-۱-۴ ارزیابی مدل

در این بخش، گام آموزشی[۵] را تعریف کرده و یک حلقه آموزشی پیاده‌سازی می‌کنیم تا مدل به‌صورت تکراری بهینه‌سازی شود.

- مدل را در حالت آموزش قرار می‌دهد. این کار لایه‌های دراپ‌اوت را فعال کرده و در شبکه‌های پیچیده‌تر، آمار نرمال‌سازی دسته‌ای را به‌روزرسانی می‌کند.

[4] debugging
[5] training step

- در PyTorch، گرادیان‌ها به‌طور پیش‌فرض جمع می‌شوند. $zero_grad()$ تضمین می‌کند که گرادیان‌های مرحله قبلی در محاسبات فعلی تداخل نداشته باشند.
- مدل، ویژگی‌های ورودی و ماتریس مجاورت را پردازش می‌کند تا خروجی را تولید کند. خروجی شامل احتمالات دسته‌بندی برای هر گره است.
- از تابع زیان منفی درست‌نمایی[6] استفاده می‌کنیم، زیرا لایه آخر مدل از $log_softmax$ استفاده می‌کند. زیان فقط روی مجموعه داده‌ی آموزشی محاسبه می‌شود، تا مدل فقط از داده‌های دارای برچسب یاد بگیرد.
- تابع accuracy برچسب‌های پیش‌بینی‌شده را با برچسب‌های واقعی مقایسه می‌کند. این مقدار به عنوان یک معیار ارزیابی برای بررسی عملکرد مدل در نظر گرفته می‌شود.
- loss.backward() گرادیان‌های زیان نسبت به پارامترهای مدل (وزن‌ها و بایاس‌ها) را با استفاده از پس‌انتشار محاسبه می‌کند.
- بهینه‌ساز، پارامترهای مدل را بر اساس گرادیان‌های محاسبه‌شده به‌روزرسانی می‌کند. از آنجایی که بهینه‌ساز Adam استفاده شده است، نرخ یادگیری به‌طور پویا برای هر پارامتر تنظیم می‌شود.
- این حلقه مدل را برای ۱۰۰۰ تکرار آموزش می‌دهد و زیان و دقت را در هر مرحله ثبت می‌کند.

۴-۲ اجزای سیستم و یکپارچه‌سازی با GraphGym

۴-۲-۱ ساختار گراف

ورودی مدل GCN یک مجموعه داده‌ی گرافی است. گره‌ها نشان‌دهنده‌ی نقاط داده (مثلاً مقالات پژوهشی در مجموعه داده Cora) هستند. یال‌ها نشان‌دهنده‌ی روابط بین داده‌ها (مثلاً ارجاعات مقالات به یکدیگر) هستند.

۴-۲-۲ تبدیل لایه GCN

کانولوشن گرافی ویژگی‌های گره‌ها را با استفاده از اطلاعات همسایگان به‌روزرسانی می‌کند.

[6] NLL Loss

۴-۲-۳ فرآیند آموزش

فرآیند آموزش GCN از یک جریان کاری ساختاریافته پیروی می‌کند:

۱. بارگذاری مجموعه داده

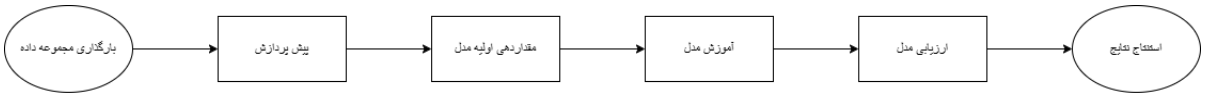
۲. پیش‌پردازش داده‌ها (نرمال‌سازی ویژگی‌ها و ماتریس مجاورت)

۳. مقداردهی اولیه مدل GCN

۴. آموزش مدل (انتشار رو به جلو + پس‌انتشار گرادیان)

۵. ارزیابی مدل (محاسبه دقت و زیان)

۶. تصویری‌سازی نتایج (تحلیل روند زیان و دقت)



شکل ۴-۱: فرایند آموزش

۴-۲-۴ معماری مدل

معماری مدل GCN شامل چندین لایه کلیدی است:

۱. لایه ورودی: پردازش ویژگی‌های گره‌ها

۲. لایه گراف کانولوشنی ۱: استخراج تعبیه‌های محلی گره‌ها

۳. لایه دراپ‌اوت: جلوگیری از بیش‌برازش

۴. لایه گراف کانولوشنی ۲: بهبود تعبیه‌های گره‌ها

۵. لایه Softmax: خروجی احتمالات برای دسته‌بندی گره‌ها

شکل ۴-۲: معماری مدل

۴-۲-۵ استفاده از GraphGym برای تنظیم مدل

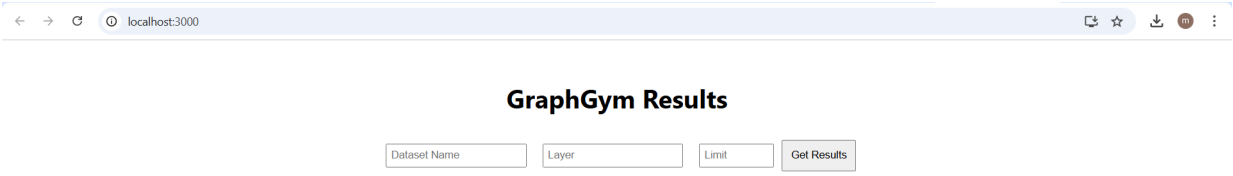
علاوه بر پیاده‌سازی مدل‌های GNN که خود پیکربندی کردیم، این پروژه از GraphGym استفاده می‌کند که یک چارچوب تخصصی برای جستجوی خودکار معماری و تنظیم ابرپارامترها در شبکه‌های عصبی گرافی است. به‌جای تنظیم دستی پارامترهای مدل و پیکربندی‌های آموزشی، GraphGym یک رویکرد ساختاریافته را برای بررسی سیستماتیک معماری‌های مختلف GNN، روش‌های تجمیع، استراتژی‌های آموزشی و تنظیمات بهینه‌سازی فراهم می‌کند.

مزایای استفاده از GraphGym در این پروژه:

- جستجوی خودکار ابرپارامترها: با تعریف فضای جستجوی شبکه‌ای در فایل dataset.txt، می‌توان پیکربندی‌های مختلفی از GNNها (مانند تعداد لایه‌ها، توابع فعال‌سازی و انواع تجمیع) را به‌طور کارآمد آزمایش کرد.
- انعطاف‌پذیری در آزمایش‌ها: فایل پیکربندی dataset.yaml امکان کنترل دقیق تنظیمات مدل را فراهم می‌کند، که نمونه‌سازی سریع و مقایسه‌ی سیستماتیک معماری‌های مختلف GNN را ممکن می‌سازد.
- مقیاس‌پذیری: GraphGym اجازه می‌دهد تا چندین پیکربندی به‌طور موازی آموزش داده شوند، که زمان موردنیاز برای یافتن بهترین مدل ممکن را به‌شدت کاهش می‌دهد.
- بازتولیدپذیری: با ذخیره‌ی تنظیمات مدل در فایل‌های YAML و TXT، آزمایش‌ها به‌راحتی قابل تکرار هستند و نتایج مدل در پیکربندی‌های مختلف GNN قابل اطمینان و سازگار باقی می‌ماند.

۴-۳ رابط کاربری

برای نمایش نتایج، یک رابط کاربری نیز توسعه دادیم. تصویر رابط کاربری در ادامه آمده است.



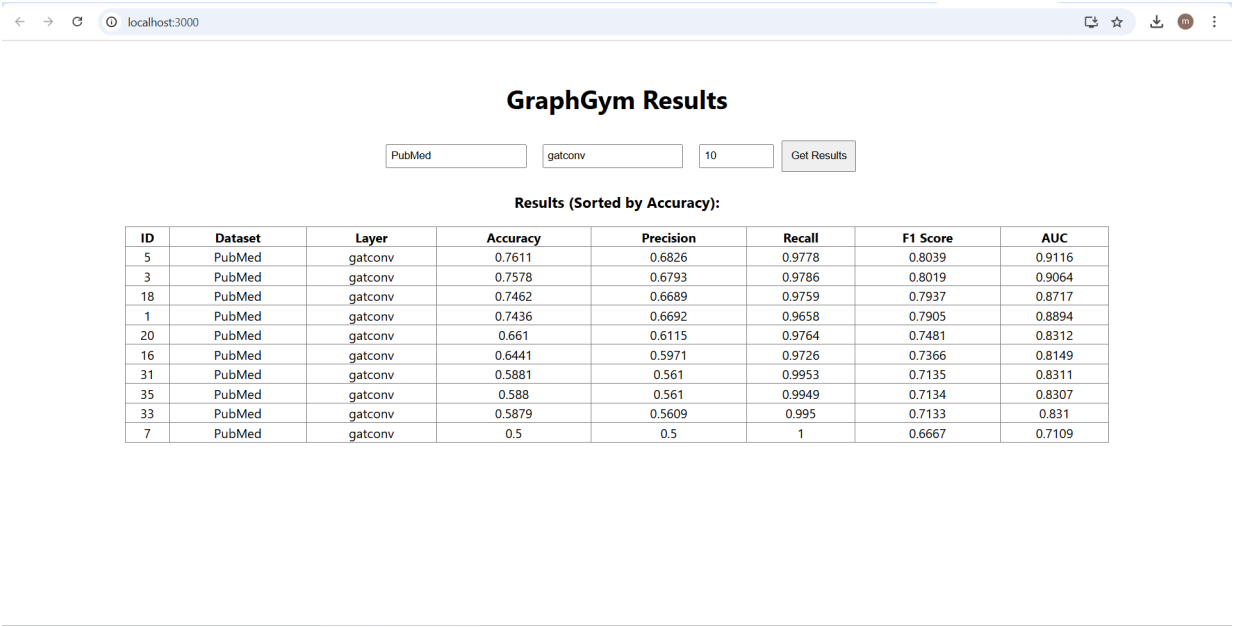
شکل ۴-۳: رابط کاربری

این رابط کاربری برای این طراحی شده است که به کاربران امکان جستجو و نمایش نتایج از GraphGym را بدهد. این رابط شامل بخش‌های زیر است:

- نام مجموعه داده[۷]: یک فیلد ورودی متنی که کاربران در آن نام مجموعه داده‌ای که می‌خواهند جستجو کنند را وارد می‌کنند.
- بخش لایه[۸]: یک فیلد ورودی متنی که کاربران نوع لایه‌ای که به آن علاقه دارند را مشخص می‌کنند. انواع لایه‌های موجود شامل LightGCN، GCN و GAT هستند.
- بخش محدودیت[۹]: یک فیلد ورودی متنی که کاربران در آن تعداد رکوردهای برتری که می‌خواهند در نتایج نمایش داده شوند را تعیین می‌کنند.
- دکمه دریافت نتایج[۱۰]: با کلیک بر روی این دکمه، نتایج فیلتر شده بر اساس نام مجموعه داده، نوع لایه و مقدار محدودیت مشخص‌شده دریافت و نمایش داده می‌شوند.

در شکل زیر، رابط کاربری پس از دریافت نتایج را مشاهده میکنید.

[7] Dataset Name
[8] Layer
[9] Limit
[10] Get Results



ID	Dataset	Layer	Accuracy	Precision	Recall	F1 Score	AUC
5	PubMed	gatconv	0.7611	0.6826	0.9778	0.8039	0.9116
3	PubMed	gatconv	0.7578	0.6793	0.9786	0.8019	0.9064
18	PubMed	gatconv	0.7462	0.6689	0.9759	0.7937	0.8717
1	PubMed	gatconv	0.7436	0.6692	0.9658	0.7905	0.8894
20	PubMed	gatconv	0.661	0.6115	0.9764	0.7481	0.8312
16	PubMed	gatconv	0.6441	0.5971	0.9726	0.7366	0.8149
31	PubMed	gatconv	0.5881	0.561	0.9953	0.7135	0.8311
35	PubMed	gatconv	0.588	0.561	0.9949	0.7134	0.8307
33	PubMed	gatconv	0.5879	0.5609	0.995	0.7133	0.831
7	PubMed	gatconv	0.5	0.5	1	0.6667	0.7109

شکل ۴-۴: رابط کاربری

فصل پنجم

نتایج تجربی

۵-۱ توضیح مجموعه داده ها

۵-۱-۱ cora

این یک شبکه‌ی استنادی از مقالات علمی در حوزه‌ی یادگیری ماشین است که در تحقیقات GNN اغلب به‌عنوان "MNIST گراف‌ها" شناخته می‌شود. این مجموعه داده شامل ۲،۷۰۸ گره (مقاله) و ۵،۴۲۹ یال استنادی است که پس از پیش‌پردازش به‌صورت یال‌های بدون‌جهت در نظر گرفته شده‌اند. هر مقاله در یکی از ۷ حوزه‌ی پژوهشی دسته‌بندی شده است. ویژگی‌های گره‌ها به‌صورت بردارهای پراکنده‌ی ۱،۴۳۳ بعدی هستند که نشان‌دهنده‌ی حضور کلمات در مقاله می‌باشند (کیسه‌ی کلمات از چکیده‌ی مقاله). ویژگی‌های کلیدی این گراف شامل درجه‌ی متوسط گره حدود ۴ (۵،۴۲۹ یال برای ۲،۷۰۸ گره) و چگالی بسیار پایین (۱۵.۰ درصد از کل اتصالات ممکن) است. این شبکه دارای ضریب خوشه‌بندی متوسط (۲۴.۰) است که نشان می‌دهد مقالات تمایل دارند در خوشه‌های موضوعی خاصی قرار گیرند. Cora یک گراف با هموفیلی بالا است. تقریباً ۸۱ درصد از یال‌ها بین گره‌هایی با برچسب یکسان برقرار شده‌اند، که این مجموعه داده را برای GNNهایی که فرض می‌کنند گره‌های همسایه دارای برچسب‌های مشابه هستند، مناسب می‌سازد. در مطالعات علمی، Cora یک معیار استاندارد برای دسته‌بندی نیمه‌نظارتی گره‌ها محسوب می‌شود. پژوهشگران معمولاً از تقسیم‌بندی ثابت "Planetoid" که توسط Yang و همکاران (۲۰۱۶) معرفی شد، استفاده می‌کنند. این تقسیم شامل ۱۴۰ گره برچسب‌دار برای آموزش، ۵۰۰ گره برای اعتبارسنجی و ۱۰۰۰ گره برای آزمایش است و در مقاله‌ی اصلی GCN نیز به کار گرفته شده است. در پژوهش‌های جدید، آزمایش‌ها اغلب با چندین تقسیم‌بندی تصادفی (مثلاً ۲۰ گره بر کلاس برای آموزش) تکرار می‌شوند تا ارزیابی مدل‌ها پایدارتر و دقیق‌تر باشد. پیش‌پردازش‌های رایج شامل تبدیل گراف به بدون‌جهت (با افزودن یال‌های متقابل برای استنادات) و حذف گره‌های منفرد یا گره‌هایی که در مجموعه داده اصلی وجود ندارند است. به‌عنوان مثال، در Citeseer برخی لینک‌های استنادی به مقالاتی اشاره داشتند که در لیست گره‌ها وجود نداشتند، که در این مرحله حذف می‌شوند. به دلیل هموفیلی قوی و ویژگی‌های غنی گره‌ها، مدل‌هایی که از ویژگی‌های همسایگان استفاده می‌کنند (مانند GCN و GAT) عملکرد بهتری دارند. همچنین، درجه‌ی میانگین پایین این گراف نشان می‌دهد که بیشتر همسایگان گره‌ها مقالات علمی مرتبط هستند، بنابراین استفاده از مکانیزم‌های پیچیده‌ی توجه در اینجا اهمیت کمتری دارد.

۵-۱-۲ citeseer

یک شبکه‌ی استنادی دیگر شامل ۳،۳۱۲ گره (مقاله) و ۴،۷۳۲ یال استنادی است. مقالات این مجموعه در ۶ دسته‌ی پژوهشی طبقه‌بندی شده‌اند. هر گره دارای یک بردار ویژگی دودویی ۳،۷۰۳ بعدی است که نشان‌دهنده‌ی حضور کلمات در سند می‌باشد. این ویژگی باعث می‌شود ماتریس ویژگی‌ها بسیار پراکنده باشد (هر مقاله تنها شامل زیرمجموعه‌ی کوچکی از واژگان دیکشنری است). گراف نیز بسیار پراکنده است و دارای درجه‌ی میانگین تقریباً ۹.۲ می‌باشد. این گراف دارای هموفیلی بالایی است (تقریباً ۷۳ درصد از یال‌ها بین گره‌های یک کلاس قرار دارند)، اگرچه این میزان کمی کمتر از Cora است. همچنین، شامل چندین گره‌ی مجزا یا اجزای کوچک قطع‌شده می‌باشد. در صورتی که این گره‌ها حذف شوند، بزرگ‌ترین مؤلفه‌ی متصل شامل حدود ۲،۱۱۰ گره خواهد بود. در عمل، Citeseer همراه با Cora به‌عنوان یک معیار استاندارد برای دسته‌بندی گره‌ها در GNNها استفاده می‌شود تا توانایی این مدل‌ها در پردازش ویژگی‌های پراکنده و هموفیلی کمتر ارزیابی شود. تقسیم‌بندی استاندارد[1] شامل ۱۲۰ گره‌ی آموزشی (۲۰ گره در هر کلاس) در تنظیم انتقالی است. پژوهشگران اشاره کرده‌اند که Citeseer می‌تواند برای GNNها چالش‌برانگیزتر از Cora باشد. زیرا با داشتن میانگین لینک‌های کمتر برای هر گره، اطلاعات همسایگی کمتری برای تجمیع در دسترس است. همچنین، تقریباً ۲۵ درصد از همسایگان هر گره ممکن است متعلق به کلاس‌های متفاوتی باشند (یال‌های هتروفیل) که می‌تواند نویز در مدل ایجاد کند. پیش‌پردازش آن مشابه با Cora است، به این صورت که یال‌ها بدون‌جهت در نظر گرفته شده و یال‌های نامعتبر با گره‌های ناموجود حذف می‌شوند. بعد بالای ویژگی‌ها (۳،۷۰۳ بعدی) لزوماً به معنای اطلاعات غنی‌تر نیست، زیرا بسیاری از ویژگی‌ها بسیار پراکنده هستند؛ بنابراین، مدل‌ها باید بتوانند سیگنال مفید را از فضای ویژگی‌های پراکنده و با ابعاد بالا استخراج کنند.

۵-۱-۳ Pubmed

مجموعه داده PubMed Diabetes یک شبکه‌ی استنادی بسیار بزرگ‌تر است که شامل ۱۹،۷۱۷ گره (اسناد از PubMed) و ۴۴،۳۳۸ یال می‌باشد. هر مقاله مرتبط با دیابت است و در یکی از ۳ کلاس دسته‌بندی شده است: دیابت نوع ۱، دیابت نوع ۲، یا دیابت به‌صورت کلی. ویژگی‌های گره‌ها به‌جای بردارهای دودویی کیسه‌ی کلمات، بردارهای ۵۰۰ بعدی وزن‌گذاری‌شده با TF-IDF هستند. این ویژگی‌ها نسبتاً چگال بوده و اصطلاحات مهم در هر سند را بهتر نمایش می‌دهند. از نظر ساختاری، PubMed

[1] Planetoid

یک گراف پراکنده است (درجه‌ی میانگین ۵.۴) و هموفیلی بالایی دارد (حدود ۸۰درصد از یال‌ها بین مقالات هم‌کلاس برقرار شده‌اند). در واقع، بسیاری از همسایگان یک مقاله دارای همان برچسب هستند که این امر ناشی از الگوهای استنادی خاص برای موضوعات علمی است. مقیاس بزرگ PubMed و نوع ویژگی‌های آن، این مجموعه داده را از Cora و Citeseer متمایز می‌کند. از آنجا که بردارهای ویژگی TF-IDF اطلاعات بیشتری را حمل می‌کنند، حتی یک مدل دسته‌بندی ساده فقط با استفاده از ویژگی‌های گره (بدون در نظر گرفتن گراف) می‌تواند به دقت بالایی دست یابد. به‌عنوان مثال، یک پرسپترون چند لایه که فقط از ویژگی‌های گره استفاده کند، می‌تواند به دقت بیش از ۸۷درصد در PubMed برسد، که تقریباً برابر با عملکرد GCN در این مجموعه داده است. این موضوع نشان می‌دهد که ساختار گراف در PubMed نسبت به Cora و Citeseer اطلاعات اضافی کمتری ارائه می‌دهد، نکته‌ای که در مطالعات GNN مورد توجه قرار گرفته است. با این وجود، GNNها همچنان برای هموارسازی و انتشار برچسب‌ها در PubMed مفید هستند و این مجموعه داده همچنان یک معیار استاندارد برای دسته‌بندی نیمه‌نظارتی گره‌ها محسوب می‌شود. پروتکل‌های آزمایشی اغلب مشابه Cora/Citeseer هستند (با ۶۰ گره آموزشی در تقسیم‌بندی Planetoid، یعنی ۲۰ گره در هر کلاس برای ۳ کلاس) و نشان داده شده است که مدل‌های GCN/GAT می‌توانند به دقت ۸۰درصد در PubMed دست یابند. PubMed همچنین در تحقیقات پیش‌بینی لینک مورد استفاده قرار می‌گیرد، زیرا اندازه‌ی بزرگ آن تعداد کافی از یال‌ها را برای ارزیابی فراهم می‌کند. در این زمینه، یک زیرمجموعه از یال‌ها برای ارزیابی حذف می‌شود و اطمینان حاصل می‌شود که گراف همچنان متصل باقی بماند؛ همچنین، یال‌های منفی از میان جفت گره‌هایی که لینک استنادی ندارند، نمونه‌برداری می‌شوند. هموفیلی بالا و ویژگی‌های نسبتاً چگال در PubMed نشان می‌دهد که روش‌هایی که علاوه بر ارتباطات، از محتوای گره نیز استفاده می‌کنند، می‌توانند در پیش‌بینی لینک عملکرد بهتری داشته باشند (زیرا دو مقاله با محتوای بسیار مشابه احتمال زیادی دارند که یکدیگر را استناد کنند، حتی اگر به‌طور مستقیم متصل نباشند، که یک مدل مبتنی بر ویژگی می‌تواند از این الگو بهره ببرد).

۵-۱-۴ Amazon Photo

این دیتاست، بخشی از شبکه‌ی خرید مشترک محصولات آمازون است که در مجموعه‌ی داده‌ی "Pitfalls of GNN Evaluation" معرفی شده است. گره‌ها نشان‌دهنده‌ی محصولات (در دسته‌ی "Photo"، مانند دوربین‌ها و لنزها) هستند و یال‌ها نشان می‌دهند که دو محصول به‌طور مکرر با هم خریداری شده‌اند. گراف

Amazon Photo شامل ۷،۶۵۰ گره و ۱۱۹،۰۴۳ یال بدون‌جهت است، که به‌طور قابل‌توجهی متراکم‌تر از شبکه‌های استنادی مانند Cora و Citeseer است (درجه‌ی میانگین ۳۱). هر گره محصول دارای یک بردار ویژگی ۷۴۵ بعدی است که از رمزگذاری کیسه‌ی کلمات[۲] روی نظرات مشتریان استخراج شده است. اگرچه این ویژگی‌ها پراکنده هستند، اما معمولاً نشان‌دهنده‌ی دسته‌ی محصول می‌باشند (به‌عنوان مثال، نظرات مربوط به دوربین معمولاً شامل اصطلاحات فنی مرتبط با عکاسی هستند). برچسب‌های این مجموعه داده، دسته‌های محصول هستند که شامل ۸ کلاس مختلف می‌باشند. Amazon Photo یک گراف با هموفیلی بالا است (مشابه Cora و شبکه‌های مشابه). محصولات در یک دسته معمولاً به‌طور مکرر با هم خریداری می‌شوند، بنابراین یال‌ها تمایل دارند محصولات مشابه را به هم متصل کنند. در واقع، مدل‌های GNN می‌توانند در این مجموعه داده به دقت بسیار بالایی دست یابند (بیش از ۹۰درصد در بسیاری از موارد) که نشان‌دهنده‌ی هموفیلی قوی و ساختار خوشه‌ای واضح در شبکه است. همچنین، این شبکه احتمالاً دارای ضریب خوشه‌بندی بالاتری است، زیرا گروه‌های محصولات مرتبط، جوامع خرید خاصی را تشکیل می‌دهند. Amazon Photo معمولاً برای ارزیابی مدل‌های دسته‌بندی گره در مقالات جدید GNN استفاده می‌شود (در تنظیمات القایی و انتقالی، به‌ویژه برای بررسی عملکرد روی گراف‌هایی که از Cora بزرگ‌تر و متراکم‌تر هستند اما همچنان هموفیلی بالایی دارند. روش استاندارد تقسیم داده‌ها به‌صورت تصادفی است (مثلاً ۲۰ یا ۵۰ گره در هر کلاس برای آموزش)، زیرا این مجموعه داده بعداً برای ارزیابی گسترده معرفی شد و برخلاف Cora و Citeseer دارای یک تقسیم‌بندی ثابت نیست. یکی از تفاوت‌های مهم با شبکه‌های استنادی این است که گراف‌های آمازون نوعی گراف کاربر-محصول یک‌حالته هستند. در اصل، این یک گراف محصول-محصول است که یال‌ها از رفتار کاربران ناشی می‌شوند. برخلاف شبکه‌های استنادی، این گراف فاقد بُعد زمانی یا جهت‌دار بودن یال‌ها است. این ویژگی، همراه با درجه‌ی میانگین بالاتر، می‌تواند بر مدل‌های GNN تأثیر بگذارد:

- یک گره در Amazon Photo ممکن است ده‌ها همسایه داشته باشد که در چندین زیرمجموعه‌ی مرتبط قرار می‌گیرند، بنابراین مکانیزم‌های توجه (مانند GAT) ممکن است به مدل کمک کند تا روی مهم‌ترین همسایگان تمرکز کند (به‌عنوان مثال، برای یک دوربین، همسایگان دیگر دوربین‌ها احتمالاً اطلاعات پیش‌بینی بهتری درباره‌ی دسته‌ی آن ارائه می‌دهند نسبت به یک کیف دوربین یا سه‌پایه).

- چگالی بالاتر گراف باعث افزایش بار محاسباتی در GNNها می‌شود (زیرا هر گره پیام‌های بیشتری

[۲] bag-of-words

از همسایگان خود دریافت می‌کند)، که می‌تواند بر مقیاس‌پذیری مدل‌های مبتنی بر توجه تأثیر بگذارد.

۵-۲ معیارهای ارزیابی

برای ارزیابی عملکرد مدل در وظایف دسته‌بندی گره و پیش‌بینی لینک، از چندین معیار استاندارد استفاده می‌کنیم. در ادامه، هر معیار را به زبان ساده تعریف می‌کنیم.

۵-۲-۱ دقت

دقت نسبت پیش‌بینی‌های صحیح به کل پیش‌بینی‌ها میباشد. در دسته‌بندی گره، این مقدار برابر است با تعداد گره‌هایی که مدل آن‌ها را به‌درستی برچسب‌گذاری کرده است، تقسیم بر کل گره‌ها. به‌عنوان مثال، اگر مدل ۱۰۰ گره را دسته‌بندی کند و ۹۵ مورد صحیح باشند، دقت مدل ۹۵درصد خواهد بود. دقت یک نمای کلی از میزان صحت مدل ارائه می‌دهد، اما بین کلاس‌های مختلف یا نتایج متفاوت تمایزی قائل نمی‌شود.

۵-۲-۲ دقت مثبت

این معیار کیفیت پیش‌بینی‌های مثبت را اندازه‌گیری می‌کند. این معیار برابر است با تعداد نمونه‌های مثبت درست[۳] تقسیم بر مجموع کل نمونه‌های پیش‌بینی‌شده به‌عنوان مثبت[۴]. در یک مسئله‌ی پیش‌بینی لینک یا دسته‌بندی دودویی، دقت به این سؤال پاسخ می‌دهد: "از بین تمام لینک‌هایی که مدل به‌عنوان مثبت پیش‌بینی کرده، چند مورد واقعاً صحیح هستند؟". دقت بالا به این معناست که زمانی که مدل یک لینک یا کلاس را مثبت تشخیص می‌دهد، معمولاً پیش‌بینی‌اش درست است و تعداد موارد مثبت کاذب[۵] کم است.

[3] True Positives
[4] True Positives + False Positives
[5] False Positives

۵-۲-۳ بازخوانی

بازخوانی توانایی مدل در یافتن تمام نمونه‌های مثبت واقعی را اندازه‌گیری می‌کند. این معیار برابر است با تعداد نمونه‌های مثبت درست تقسیم بر کل نمونه‌های مثبت واقعی[۶]. در پیش‌بینی لینک، بازخوانی به این سؤال پاسخ می‌دهد: "از بین تمام لینک‌های واقعی در گراف، مدل چند مورد را شناسایی کرده است؟". بازخوانی بالا نشان می‌دهد که مدل اکثر لینک‌های واقعی را پیدا می‌کند، اما نمی‌گوید که آیا پیش‌بینی‌های اضافی اشتباه هم وجود دارد یا نه.

۵-۲-۴ امتیاز F1

امتیاز F1 میانگین هارمونیک دقت و بازخوانی است و این دو معیار را در یک مقدار واحد ترکیب می‌کند. مقدار F1 بالا تنها زمانی حاصل می‌شود که هم دقت و هم بازخوانی بالا باشند؛ اگر یکی از این معیارها پایین باشد، مقدار F1 کاهش می‌یابد. این معیار به‌ویژه برای وظایف نامتوازن مانند پیش‌بینی لینک (که تعداد عدم لینک‌ها بسیار بیشتر از لینک‌هاست) مفید است. امتیاز F1 یک معیار کلی از دقت مدل در شناسایی نمونه‌های مثبت ارائه می‌دهد، به‌گونه‌ای که هم بازخوانی و هم دقت را در نظر می‌گیرد.

۵-۲-۵ AUC-ROC

AUC-ROC یک معیار متداول برای ارزیابی مدل‌های دسته‌بندی دودویی است، به‌ویژه زمانی که کلاس‌ها نامتوازن باشند. منحنی ROC، نرخ مثبت واقعی[۷] را در برابر نرخ مثبت کاذب[۸] در آستانه‌های مختلف ترسیم می‌کند. مقدار AUC بین ۰.۵ تا ۱.۰ قرار دارد. AUC = 0.5 به این معناست که مدل به اندازه‌ی حدس تصادفی عمل می‌کند. AUC = 1.0 یعنی مدل کاملاً کلاس‌ها را از هم تفکیک می‌کند. در پیش‌بینی لینک، AUC-ROC بیانگر احتمال این است که مدل به یک لینک واقعی امتیاز بالاتری نسبت به یک عدم لینک بدهد. AUC بالاتر به این معناست که مدل به‌طور کلی لینک‌های واقعی را در مقایسه با لینک‌های نادرست، بهتر رتبه‌بندی می‌کند.

[6] True Positives + False Negatives
[7] Recall
[8] False Positive Rate

۵-۳ نتایج

نتایج به دست آمده در معیار های گوناگون در جداول زیر قابل مشاهده اند.

وظایف دسته بندی گره:

جدول ۵-۱: معیار دقت

	GAT	GCN	LightGCN
Cora	**0.8881**	0.8868	0.8795
AmazonPhoto	**0.9647**	0.9641	0.9601

جدول ۵-۲: معیار دقت مثبت

	GAT	GCN	LightGCN
Cora	0.8749	**0.8805**	0.8725
AmazonPhoto	**0.9553**	0.9552	0.9447

جدول ۵-۳: معیار بازخوانی

	GAT	GCN	LightGCN
Cora	0.8707	0.8689	**0.8713**
AmazonPhoto	0.9507	**0.9517**	0.9492

جدول ۵-۴: معیار امتیاز F1

	GAT	GCN	LightGCN
Cora	0.8721	**0.8732**	0.8709
AmazonPhoto	0.9527	**0.953**	0.948

وظایف پیش بینی پیوند:

جدول ۵-۵: معیار دقت

	GAT	GCN	LightGCN
PubMed	0.735	0.659	**0.7473**
CiteSeer	0.7358	0.7284	**0.7433**

جدول ۵-۶: معیار دقت مثبت

	GAT	GCN	LightGCN
PubMed	0.6718	0.6101	**0.6744**
CiteSeer	**0.703**	0.6845	0.688

جدول ۵-۷: معیار بازخوانی

	GAT	GCN	LightGCN
PubMed	0.9203	**0.9736**	0.9566
CiteSeer	0.8357	0.8167	**0.8917**

جدول ۵-۸: معیار امتیاز F1

	GAT	GCN	LightGCN
PubMed	0.7767	0.7462	**0.7911**
CiteSeer	0.7556	0.7574	**0.7764**

۵-۳-۱ عملکرد طبقه بندی گره (Amazon Photo, Cora)

در آزمایش‌های طبقه‌بندی گره روی مجموعه داده‌های Cora (شبکه استنادی) و Amazon Photo (شبکه خرید مشترک محصولات)، هر سه مدل GNN (یعنی GCN, GAT , LightGCN) نتایج قوی‌ای به دست آوردند، به طوری که تفاوت دقت آن‌ها تنها حدود ۱ تا ۲ درصد بود. GAT در هر دو مجموعه داده عملکرد بهتری نسبت به سایر مدل‌ها داشت، در حالی که LightGCN کمی ضعیف‌تر عمل کرد. برای مثال، در Amazon Photo، مدل GAT دقتی در حدود ۹۶.۴۷ درصد به دست آورد که اندکی بهتر از GCN (۹۶.۴۱ درصد) و LightGCN (۹۶.۰۱ درصد) بود. روند مشابهی در Cora مشاهده شد، جایی که GAT به دقت ۸۸.۸۱ درصد رسید، در حالی که GCN ۸۸.۶۸درصد و LightGCN ۸۷.۹۵درصد دقت داشتند. این تفاوت‌های جزئی نشان می‌دهد که تمامی مدل‌ها به‌خوبی از ساختار هموفیلی گراف‌ها بهره برده‌اند تا گره‌ها را با دقت بالا طبقه‌بندی کنند.

:GAT

شبکه توجه گراف در هر دو مجموعه داده برتری اندکی در دقت نشان داد. توانایی آن در اختصاص ضرایب توجه آموخته‌شده به گره‌های همسایه احتمالاً به آن کمک کرده است تا روی مفیدترین اتصالات تمرکز کند و عملکرد را کمی بهبود بخشد. این نتیجه با معیارهای قبلی همخوانی دارد، جایی که GAT

معمولاً در شبکه‌های استنادی عملکرد بهتری نسبت به GCN دارد. برای مثال، Veličković و همکارانش گزارش دادند که GAT در Cora دقتی حدود ۸۳ درصد به دست آورده است، در حالی که GCN دقت ۸۱.۵ درصد را ثبت کرده است. به طور مشابه، در Amazon Photo، یک معیار ارزیابی اخیر نشان داد که GAT با دقت حدود ۹۶.۶ درصد عملکردی اندکی بهتر نسبت به GCN (۹۶.۱ درصد) دارد. نتایج ما این یافته‌ها را تأیید می‌کند . GAT در طبقه‌بندی گره‌ها عملکرد برجسته‌ای دارد، به‌ویژه زمانی که تمایزهای ظریف بین همسایگان اهمیت دارد. بااین‌حال، باید توجه داشت که این بهبود معمولاً جزئی است. در حقیقت، ارزیابی‌های مقیاس بزرگ نشان داده‌اند که هیچ مدل GNN به‌طور یکنواخت بر تمامی گراف‌ها برتری ندارد. علاوه بر این، GAT در برخی مجموعه داده‌ها پایداری کمتری در آموزش دارد؛ برای مثال، در یک مطالعه مشاهده شد که دقت GAT روی گراف‌های محصولات آمازون دارای واریانس بالا بود، هرچند در آزمایش‌های ما GAT پایدار باقی ماند و بهترین نتایج را به دست آورد.

:GCN

GCN به‌عنوان یک مبنای قوی و قابل اعتماد عملکردی تقریباً برابر با GAT داشت. در Cora و Amazon Photo، دقت GCN کمتر از ۱ درصد از GAT پایین‌تر بود. این موضوع نشان می‌دهد که در شبکه‌های بسیار هموفیلی با ویژگی‌های اطلاعاتی غنی (مانند ویژگی‌های متن کیسه کلمات در Cora و ویژگی‌های محصول در Amazon)، میانگین‌گیری ساده همسایگی در GCN بسیار مؤثر است. تحقیقات اخیر تأکید دارند که GCNهای کلاسیک با تنظیمات مناسب می‌توانند در معیارهای مختلف طبقه‌بندی گره‌ها به عملکردی در حد بهترین مدل‌های موجود دست یابند. بنابراین، نتایج قوی GCN در آزمایش‌های ما با شهرت آن به‌عنوان یک مدل کارآمد و رقابتی در طبقه‌بندی گره‌ها همخوانی دارد. کاهش جزئی دقت GCN در مقایسه با GAT احتمالاً به دلیل این است که GCN همه همسایگان را به‌طور مساوی در نظر می‌گیرد و فاقد مکانیزم توجه برای وزن‌دهی دقیق‌تر به همسایگان است. اما در شرایطی که همه همسایگان به یک اندازه مرتبط هستند (که در این مجموعه داده‌ها رایج است)، این موضوع تأثیر منفی زیادی ندارد، که توضیح می‌دهد چرا GCN تقریباً با GAT برابری کرده است.

:LightGCN

LightGCN با وجود معماری ساده‌تر خود، دقت بالایی را ارائه داد، هرچند در تمامی موارد رتبه سوم را کسب کرد. این مدل تبدیل ویژگی‌ها و فعال‌سازی غیرخطی را که در GCN/GAT وجود دارد حذف می‌کند و تنها از انتشار وزنی ماتریس مجاورت استفاده می‌کند. این سادگی احتمالاً باعث ایجاد فاصله عملکردی جزئی در Cora و Amazon Photo شده است، زیرا در این گراف‌ها ویژگی‌های گره غنی و تعاملات غیرخطی ویژگی‌ها نقش مهمی در دقت طبقه‌بندی دارند. بااین‌حال، LightGCN همچنان

توانست بیش از ۸۷درصد در Cora و حدود ۹۶درصد در Amazon دقت کسب کند که تنها ۱-۲ درصد پایین‌تر از GAT/GCN است. این موضوع نشان می‌دهد که بخش قابل توجهی از اطلاعات پیش‌بینی‌کننده از ساختار شبکه استخراج می‌شود، که LightGCN با انتشار تعبیه‌ها از طریق گراف آن را به‌خوبی ثبت می‌کند. نتایج ما با اهداف طراحی بیان‌شده توسط He و همکارانش مطابقت دارد: در وظایف طبقه‌بندی گره با ویژگی‌های معنایی غنی به‌عنوان ورودی، تحولات غیرخطی ویژگی‌ها (که در GCN/GAT وجود دارد اما در LightGCN حذف شده است) سودمند هستند. به عبارت دیگر، LightGCN کمی دقت را در وظایف طبقه‌بندی مبتنی بر ویژگی‌های غنی فدا می‌کند، اما در عوض یک مدل ساده‌تر ارائه می‌دهد. با وجود این مصالحه، عملکرد آن همچنان قابل تحسین است و نشان می‌دهد که حتی بدون تبدیل‌های پیچیده، انتشار اطلاعات همسایگی (در اصل، یک نسخه خطی‌شده از GCN) می‌تواند در محیط‌های بسیار هموفیلی تقریباً به اندازه مدل‌های عمیق‌تر عمل کند.

نتایج کلی:

به‌طور کلی، برای طبقه‌بندی گره‌ها در این دو مجموعه داده:

- GAT به دلیل مکانیزم توجه و درک بهتر اهمیت همسایگان، برتری جزئی در دقت داشت.
- GCN در رتبه دوم قرار گرفت و نشان داد که می‌تواند ساختار و ویژگی‌ها را به‌طور مؤثری ترکیب کند.
- LightGCN، در حالی که برای طبقه‌بندی گره طراحی نشده بود، عملکرد شگفت‌انگیزی داشت، اما به دلیل عدم پردازش ویژگی‌ها کمی عقب ماند.
- Cora و Amazon Photo گراف‌هایی با هموفیلی قوی هستند (گره‌های متصل اغلب برچسب‌های مشترک دارند)، که روش‌های ساده مبتنی بر انتشار اطلاعات را بهینه می‌کند.
- در سناریوهای کمتر هموفیلی یا مجموعه داده‌های با ویژگی‌های ضعیف‌تر، عملکرد نسبی این مدل‌ها ممکن است تغییر کند.

۵-۳-۲ عملکرد پیش‌بینی پیوند (PubMed, Citeseer)

برای وظیفه پیش‌بینی لینک در شبکه‌های PubMed و Citeseer، مدل‌ها را با استفاده از چندین معیار ارزیابی (دقت، یادآوری، امتیاز F1 و AUC) بررسی کردیم تا تصویری جامع از عملکرد آن‌ها به دست آوریم. برخلاف طبقه‌بندی گره، پیش‌بینی لینک یک طبقه‌بندی دودویی (لینک در مقابل عدم لینک)

در یک شرایط نامتوازن است (تعداد لینک‌های واقعی بسیار کمتر از عدم لینک‌ها است)، بنابراین توجه به یادآوری و AUC اهمیت بالایی دارد.

:LightGCN

نتایج ما نشان داد که LightGCN معمولاً در پیش‌بینی لینک برتری داشت و بهترین تعادل بین دقت و یادآوری را ارائه داد، در حالی که GCN و GAT بسته به معیارهای مختلف نتایج متفاوتی داشتند. ما مشاهده کردیم که در این مدل‌ها بین معیارهای ارزیابی نوعی مبادله وجود دارد. مدلی که بالاترین میزان یادآوری را داشت، همیشه بهترین F1 یا دقت را کسب نکرد. دقت به‌تنهایی می‌تواند در پیش‌بینی لینک گمراه‌کننده باشد، اما همچنان معیار مهمی محسوب می‌شود. LightGCN بالاترین دقت را در هر دو مجموعه داده به دست آورد. در PubMed، LightGCN به ۷۴.۷۳ درصد دقت رسید، که بالاتر از GAT (۷۳.۵۰ درصد) و بسیار بالاتر از GCN (۶۵.۹۰ درصد) بود. در Citeseer، LightGCN با ۷۴.۳۳ درصد دقت از GAT (۷۳.۵۸ درصد) و GCN (۲۷.۸۴ درصد) پیشی گرفت. این تفاوت‌های دقت نشان می‌دهند که LightGCN نسبت به بقیه مدل‌ها خطاهای کمتری در طبقه‌بندی لینک‌ها و عدم لینک‌ها داشته است.

:GCN

GCN در PubMed دقت بسیار پایین‌تری داشت، که نشان می‌دهد بسیاری از عدم لینک‌ها را به اشتباه به‌عنوان لینک شناسایی کرده است. با این حال، در معیار یادآوری عملکرد فوق‌العاده‌ای داشت. GCN در PubMed به یادآوری ۹۶.۳۶ درصد رسید که بالاترین مقدار در بین تمام مدل‌ها بود (LightGCN: ۹۵.۶۶ درصد، GAT: ۹۲.۰۳ درصد). این به این معناست که GCN تقریباً همه لینک‌های واقعی را شناسایی کرد، اما با هزینه افزایش زیاد نرخ مثبت کاذب[۹]. در Citeseer، وضعیت متفاوت بود. GCN پایین‌ترین میزان یادآوری را داشت (۸۱.۶۷ درصد)، در حالی که LightGCN بالاترین مقدار را داشت (۸۹.۱۷ درصد) و GAT در سطح میانی قرار گرفت (۸۳.۵۷ درصد). این نشان می‌دهد که GCN گاهی بیش از حد "مشتاق" به پیش‌بینی لینک‌ها است، در حالی که LightGCN رویکرد متعادل‌تری اتخاذ می‌کند و GAT بین این دو قرار دارد.

امتیاز F1:

در PubMed، LightGCN با امتیاز F1 = ۰.۷۹۱۱ بهترین عملکرد را داشت و از GAT (۰.۷۷۶۷) و GCN (۰.۷۴۶۲) پیشی گرفت. در Citeseer، LightGCN همچنان در صدر قرار داشت (F1 ۰.۷۷۶۴ =) در مقایسه با GCN (۰.۷۵۷۴) و GAT (۰.۷۵۵۶). این نشان می‌دهد که LightGCN تعادل بهتری

[9] False Positives

بین دقت و یادآوری ایجاد کرده است، در حالی که GCN هرچند یادآوری بالایی داشت، اما به دلیل دقت پایین، امتیاز F1 آن کاهش یافت.

:AUC

معیار AUC (مساحت زیر منحنی ROC) نشان می‌دهد که مدل تا چه اندازه می‌تواند لینک‌های واقعی را از عدم لینک‌ها در یک طیف رتبه‌بندی، مستقل از آستانه تصمیم‌گیری خاص، جدا کند. برخلاف نتایج دقت، GCN بالاترین AUC را در PubMed داشت (۰.۹۰۷۷) در مقایسه با LightGCN (۰.۸۷۸۹) و GAT (۰.۸۶۲۹). این نشان می‌دهد که GCN به‌طور کلی لینک‌های واقعی را از غیرواقعی‌ها بهتر رتبه‌بندی می‌کرد، حتی اگر در نقطه تصمیم‌گیری خاص، بسیاری از لینک‌ها را اشتباه طبقه‌بندی کرده باشد. این امر می‌تواند به انتخاب آستانه یا وزن‌دهی نادرست کلاس‌ها در هنگام تصمیم‌گیری نهایی مرتبط باشد. در Citeseer، LightGCN بالاترین AUC را داشت (۰.۸۴۹۳) در حالی که GAT (۰.۸۱۵۵) و GCN (۰.۸۱۱۶) کمی پایین‌تر بودند. AUC بالاتر LightGCN در Citeseer با برتری آن در یادآوری و امتیاز F1 همخوانی دارد، که نشان می‌دهد این مدل لینک‌های واقعی را به‌طور مؤثرتری نسبت به عدم لینک‌ها رتبه‌بندی کرده است.

نتایج کلی:

- AUC و F1 معیارهای حیاتی برای مقایسه مدل‌ها در پیش‌بینی لینک هستند، زیرا دقت به‌تنهایی می‌تواند به دلیل عدم توازن کلاس‌ها گمراه‌کننده باشد.
- LightGCN و مدل‌های ساده‌شده GNN می‌توانند به‌شدت رقابتی باشند، و در برخی موارد از معماری‌های پیچیده‌تر عملکرد بهتری ارائه دهند.
- مقایسه GCN و GAT به مجموعه داده بستگی دارد – GCN در PubMed برتری داشت، اما LightGCN در Citeseer بهتر عمل کرد.
- شواهد تجربی نشان می‌دهند که در شبکه‌های استنادی مانند PubMed و Citeseer، پیچیدگی GAT برای پیش‌بینی لینک ضروری نیست، و یک GCN تنظیم‌شده یا LightGCN می‌تواند عملکردی برابر یا حتی برتر ارائه دهد.

فصل ششم

نتیجه گیری

در نتیجه، GCN، GAT و LightGCN هر کدام نقاط قوت و نقاط ضعف خاص خود را در وظایف دسته‌بندی گره‌ها و پیش‌بینی پیوندها نشان می‌دهند. نتایج تجربی ما، در کنار دیدگاه‌های استخراج‌شده از منابع علمی، تصویری متعادل ارائه می‌دهند.

۶-۰-۱ GCN

GCN عملکرد پایه‌ای قوی‌ای را هم در دسته‌بندی گره‌ها و هم در پیش‌بینی پیوندها ارائه می‌دهد. این مدل ساده، از نظر محاسباتی کارآمد است و ساختار گراف و ویژگی‌های گره‌ها را به‌خوبی بهره‌برداری می‌کند. GCN تقریباً بالاترین دقت را در دسته‌بندی گره‌ها کسب کرد (کمتر از ۱ درصد پایین‌تر از GAT) و در پیش‌بینی پیوندها در مجموعه داده PubMed نیز مقادیر بازخوانی و AUC استثنایی‌ای از خود نشان داد. نقطه ضعف GCN در این است که وزن‌دهی صریحی برای همسایه‌ها ندارد. یعنی تمام همسایه‌ها را به‌صورت یکنواخت در نظر می‌گیرد، که می‌تواند در سناریوهایی که برخی همسایه‌ها دارای نویز یا کم‌اهمیت هستند، باعث کاهش عملکرد شود. همچنین مشاهده کردیم که GCN در پیش‌بینی پیوندها ممکن است بیش‌پیش‌بینی کند (یعنی بازخوانی بالا ولی همراه با مثبت کاذب[1] زیاد)، بنابراین نیاز به کالیبراسیون دقیق (مثلاً تنظیم آستانه تصمیم‌گیری یا استفاده از روش‌های منظم‌سازی مناسب) دارد.

توصیه: از GCN به‌عنوان یک GNN چندمنظوره و قابل‌اعتماد استفاده کنید، به‌ویژه زمانی که به مدلی سریع و آماده استفاده نیاز دارید. این مدل برای شبکه‌های با هموفیلی بالا و سناریوهایی که تفسیر دقیق تأثیر همسایه‌ها اهمیت زیادی ندارد، بسیار مناسب است. با تنظیم درست ابرپارامترها، GCN می‌تواند به نتایج سطح بالا یا حتی بهتر از مدل‌های پیشرفته برسد و به‌عنوان یک انتخاب پیش‌فرض خوب برای دسته‌بندی گره‌ها مطرح شود. برای پیش‌بینی پیوند نیز GCN مؤثر است، اما باید تعادل بین دقت مثبت و بازخوانی را مد نظر قرار داد؛ اگر بالا بودن بازخوانی مهم است و وجود برخی مثبت کاذب ها قابل‌قبول باشد، GCN انتخاب مناسبی است، اما اگر دقت مثبت حیاتی باشد، ممکن است نیاز به تنظیم آستانه یا انتخاب مدلی دیگر باشد.

۶-۰-۲ GAT

GAT این توانایی را دارد که اهمیت همسایه‌های مختلف را از طریق ضرایب توجه یادگرفته‌شده تمایز

[1] false positive

دهد. در آزمایش‌های ما برای دسته‌بندی گره‌ها، GAT بالاترین دقت را در هر دو مجموعه‌داده Cora و Amazon Photo به‌دست آورد، هرچند تنها اندکی بیشتر از GCN بود. این نشان می‌دهد که مزیت GAT در گراف‌های با هموفیلی بالا، اندک است. اما در گراف‌های هتروفیل، که تشخیص همسایه‌های مؤثر اهمیت دارد، می‌تواند بسیار مهم‌تر باشد. در زمینه پیش‌بینی پیوند، GAT برتری آشکاری در داده‌های ما نشان نداد. عملکرد آن قابل‌قبول بود اما نه بالاترین مقدار بازخوانی را داشت، نه بهترین F1 یا AUC. از جمله معایب اصلی GAT می‌توان به پیچیدگی محاسباتی و احتمال بی ثباتی در آموزش اشاره کرد. محاسبه ضرایب توجه برای هر یال می‌تواند در گراف‌های بزرگ منابع زیادی مصرف کند و GAT در برخی وظایف ممکن است نوسانات عملکردی زیادی داشته باشد.

توصیه: از GAT در شرایطی استفاده کنید که احتمال می‌دهید اهمیت همسایه‌ها تفاوت زیادی دارد، یا زمانی که به تفسیرپذیری بیشتری در مورد اینکه کدام همسایه‌ها تأثیرگذار هستند نیاز دارید. GAT برای مسائل دسته‌بندی گره در گراف‌هایی که اتصالات نامربوط دارند یا فقط بخشی از همسایه‌ها اطلاعاتی هستند (مثل شبکه‌های اجتماعی که همه دوستان تأثیر یکسانی ندارند) بسیار مناسب است. همچنین GAT در مقابله با هتروفیلی بهتر عمل می‌کند، چون همسایه‌های هماهنگ‌تر را وزن بیشتری می‌دهد. با این حال باید به هزینه آن توجه داشت. در گراف‌های بسیار بزرگ یا کاربردهای حساس به تأخیر، باید سنجید که آیا این افزایش جزئی در دقت ارزش سربار محاسباتی را دارد یا خیر. برای پیش‌بینی پیوند،GAT معمولاً انتخاب اول نیست؛ اگر همچنان بخواهید از مکانیزم توجه استفاده کنید، معماری‌های خاص‌تر (مثل پیش‌بینی‌کننده‌های پیوند مبتنی بر توجه یا ترکیب GAT با ویژگی‌های ابتکاری) ممکن است مزایای محسوسی نسبت به مدل‌های ساده‌تری مانند GCN ارائه دهند. در مجموع، GAT مدلی قدرتمند است، اما باید در شرایطی به‌کار رود که انتخاب دقیق همسایه‌ها واقعاً ارزش افزوده ایجاد کند.

۶-۰-۳ LightGCN

این مدل یک نسخه سبک‌شده از GCN است که غیرخطی‌سازی‌ها و تبدیلات یادگیری‌شده را حذف کرده و تمرکز خود را به‌طور کامل بر روی اتصالات مرتبه‌بالای گراف معطوف می‌کند. نتایج ما نشان می‌دهد که این طراحی، که در ابتدا برای سیستم‌های توصیه‌گر پیشنهاد شده بود، در وظایف پیش‌بینی پیوند عملکرد بسیار خوبی دارد. LightGCN بهترین مقادیر امتیاز F1 و AUC را در مجموعه‌داده‌های PubMed و Citeseer به دست آورد، که نشان می‌دهد این مدل در پیش‌بینی پیوندها از طریق انتشار تأثیر در شبکه بسیار مؤثر عمل می‌کند. عملکرد آن در دسته‌بندی گره‌ها کمی پایین‌تر بود، که قابل انتظار است؛ زیرا

برخلاف GCN یا GAT از ویژگی‌های گره‌ها به‌طور کامل استفاده نمی‌کند. مزیت کلیدی LightGCN در کارایی بالا و تخصص در سیگنال‌های ساختاری است. این مدل به دلیل نداشتن ماتریس‌های وزنی در هر لایه، پارامترهای کمتری دارد، در نتیجه سریع‌تر آموزش می‌بیند و کمتر دچار بیش‌برازش روی نویز ویژگی‌ها می‌شود، که به بهبود عملکرد آن در پیش‌بینی پیوند کمک کرده است. در مقابل، نقطه ضعف آن کاهش قابلیت بیانی در وظایفی است که نیاز به یادگیری ویژگی دارند. چرا که LightGCN اساساً فرض می‌کند که ساختار گراف منبع اصلی اطلاعات است.

توصیه: از LightGCN (یا مدل‌های ساده‌شده‌ی مشابه) در پیش‌بینی پیوند و سیستم‌های توصیه‌گر، به‌ویژه در گراف‌های بزرگ‌مقیاس استفاده کنید. اگر داده‌های شما شامل شبکه تعامل کاربر-آیتم، شبکه ارجاع مقالات یا هر سناریویی است که در آن هدف، پیش‌بینی یال‌هاست و معتقدید ساختار گراف اهمیت بیشتری نسبت به ویژگی‌های گره‌ها دارد، LightGCN انتخابی عالی است. همان‌طور که در نتایج دیدیم، این مدل احتمالاً مقادیر بالایی از بازخوانی و عملکرد خوب در رتبه‌بندی کلی ارائه خواهد داد و همچنین به‌خوبی در گراف‌های بزرگ مقیاس‌پذیر است (ویژگی مهم برای سیستم‌های صنعتی توصیه‌گر). از سوی دیگر، برای وظایف دسته‌بندی گره‌ها با ویژگی‌های غنی، LightGCN احتمالاً بهترین عملکرد را نخواهد داشت؛ در این حالت، یک GCN یا GAT استاندارد که قابلیت تبدیل ویژگی‌ها را دارد، معمولاً بهتر خواهد بود. با این حال، اگر مقیاس‌پذیری حتی در دسته‌بندی نیز مهم باشد (مثلاً میلیون‌ها گره) و به مدلی بسیار سریع نیاز دارید، می‌توان LightGCN را به‌عنوان یک خط پایه[۲] در نظر گرفت. با این آگاهی که ممکن است کمی از دقت کاسته شود. برخی پژوهش‌های جدید حتی پیشنهاد داده‌اند که می‌توان از ترکیب مزایا بهره برد: مثلاً استفاده از بدنه‌ی اصلی LightGCN برای کارایی بالا، اما افزودن یک تبدیل سبک برای ویژگی‌ها در لایه نهایی به منظور دسته‌بندی. اگرچه بررسی این مدل‌های ترکیبی خارج از حوزه این بحث است.

۶-۰-۴ نتیجه نهایی

در مجموع، هر مدل نقطه قوت خاص خود را دارد: GCN مدلی متعادل برای بسیاری از سناریوهاست، GAT در تشخیص اهمیت همسایه‌ها تخصص دارد (مفید برای وظایف گره‌محور)، و LightGCN در استخراج ساختار پیوند در گراف‌های بزرگ عملکرد بالایی دارد. به‌جای انتخاب یک مدل برتر مطلق، پیشنهاد می‌شود با توجه به ماهیت مسئله، مدل مناسب را انتخاب کنید. حتی می‌توان از ترکیب مدل‌ها

[۲] baseline

بهره برد. مثلاً GAT برای دسته‌بندی گره و LightGCN برای پیش‌بینی پیوند در همان گراف. نتایج ما و منابع علمی نشان می‌دهند که انتخاب و تنظیم دقیق مدل‌ها می‌تواند بدون نیاز به معماری‌های پیچیده‌تر، نتایج بهینه‌ای ارائه دهد. در نهایت، درک ویژگی‌های گراف (هم‌دوستی، غنای ویژگی‌ها، مقیاس) و شاخص‌های عملکردی مورد نظر، کلید انتخاب مدل مناسب GNN است.

منابع و مراجع

[1] Gilmer, Justin, Schoenholz, Samuel S., Riley, Patrick F., Vinyals, Oriol, and Dahl, George E. Neural message passing for quantum chemistry, 2017. https://proceedings.mlr.press/v70/gilmer17a.html.

[2] Hamilton, William L., Ying, Zhitao, and Leskovec, Jure. Inductive representation learning on large graphs, 2017. https://arxiv.org/abs/1706.02216.

[3] He, Xiangnan, Deng, Kuan, Wang, Xiang, Li, Yan, Zhang, Yongdong, and Wang, Meng. Lightgcn: Simplifying and powering graph convolution network for recommendation, 2020. https://arxiv.org/abs/2002.02126.

[4] Kipf, Thomas N. and Welling, Max. Semi-supervised classification with graph convolutional networks, 2017. https://openreview.net/forum?id=SJU4ayYgl.

[5] Scarselli, Franco, Gori, Marco, Tsoi, Ah Chung, Hagenbuchner, Markus, and Monfardini, Gabriele. The graph neural network model, 2009. https://arxiv.org/abs/2009.11859.

[6] Veličković, Petar, Cucurull, Guillem, Casanova, Arantxa, Romero, Adriana, Liò, Pietro, and Bengio, Yoshua. Graph attention networks, 2018. https://arxiv.org/abs/1710.10903.

[7] Wu, Felix, Souza, Amauri, Zhang, Tianyi, Fifty, Christopher, Yu, Tao, and Weinberger, Kilian Q. Simplifying graph convolutional networks, 2019. https://proceedings.mlr.press/v97/wu19e.html.

[8] Wu, Zonghan, Pan, Shirui, Chen, Fengwen, Long, Guodong, Zhang, Chengqi, and Yu, Philip S. A comprehensive survey on graph neural networks, 2020. https://arxiv.org/abs/1901.00596.

[9] Xu, Keyulu, Hu, Weihua, Leskovec, Jure, and Jegelka, Stefanie. How powerful are graph neural networks?, 2019. https://openreview.net/forum?id=ryGs6iA5Km.

[10] Zhou, Jie, Cui, Ganqu, Zhang, Zhengyan, Yang, Cheng, Liu, Zhiyuan, Wang, Lifeng, Li, Changcheng, and Sun, Maosong. Graph neural networks: A review of methods and applications, 2020. https://arxiv.org/abs/1812.08434.