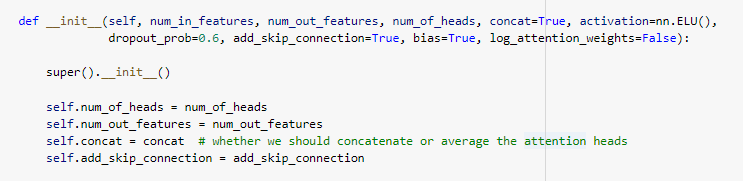
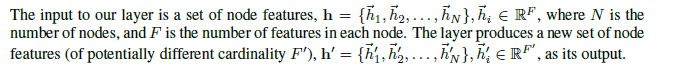
شبکه‌های گرافی توجه: گزارش فاز پیاده‌سازی

*محمد ابراهیمی، پارسا عباسی*

*دانشگاه علم و صنعت ایران، تهران* [*m\_ebrahimi74@comp.iust.ac.ir*](mailto:m_ebrahimi74@comp.iust.ac.ir)*,* [*parsa\_abbasi@comp.iust.ac.ir*](mailto:parsa_abbasi@comp.iust.ac.ir)

**1) لایۀ GAT**

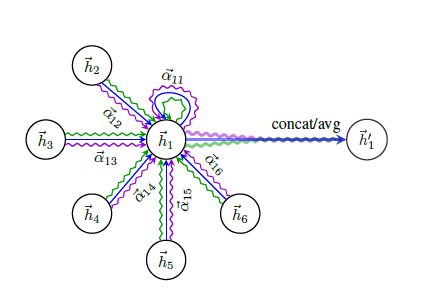




ورودی‌های این لایه به شرح زیر هستند:

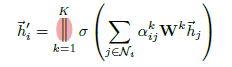
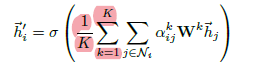
|  |  |
| --- | --- |
| **نام متغیر** | **مشخصات** |
| num\_in\_features | تعداد ویژگی‌های ورودی لایه |
| num\_out\_features | تعداد ویژگی‌های خروجی لایه |
| num\_of\_heads | تعداد هد **(1)** |
| concat | مشخص کنندۀ اینکه خروجی هر هد باید با یکدیگر الحاق شود (True) یا از آن‌ها میانگین گرفته شود (False) **(2)** |
| activation | تابع فعالسازی (پیش فرض برابر Exponential ReLU یا به اختصار elu) |
| dropout\_prob | احتمال حذف تصادفی |
| add\_skip\_connection | استفاده از تکنیک skip connection **(3)** |
| bias | اضافه کردن بایاس |
| log\_attention\_weights | تعیین کنندۀ لاگ گرفتن از وزن‌های توجه |

**(1)** برای پایدار کردن[[1]](#footnote-1) فرایند یادگیری ممکن است از چندین هد استفاده کنیم. به عنوان مثال در شکل 1-1 سه هد مختلف مورد استفاده قرار گرفته که با سه رنگ بنفش، آبی و سبز نشان داده شده‌اند.

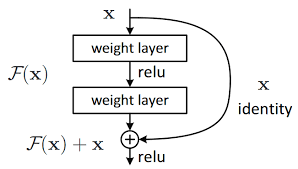


**شکل 1-1:** نحوه عملکرد هدهای مختلف در لایۀ GAT

**(2)** متغیر مربوطه مشخص‌کنندۀ این است که خروجی هدها را باید با یکدیگر الحاق کرد (لایه‌های میانی) یا از آن‌ها میانگین (لایۀ خروجی) گرفت.

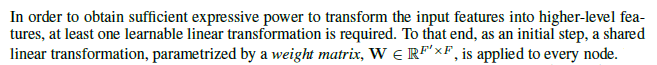
 

**(3)** استفاده از تکنیک رایج skip connection که مطابق با شکل 2-1 به منظور انتقال ورودی به کار گرفته می‌شود.



**شکل 2-1:** عملکرد تکنیک skip connection

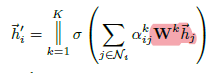
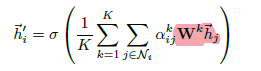
**1-1) تبدیل خطی و رگولاریزیشن[[2]](#footnote-2)**



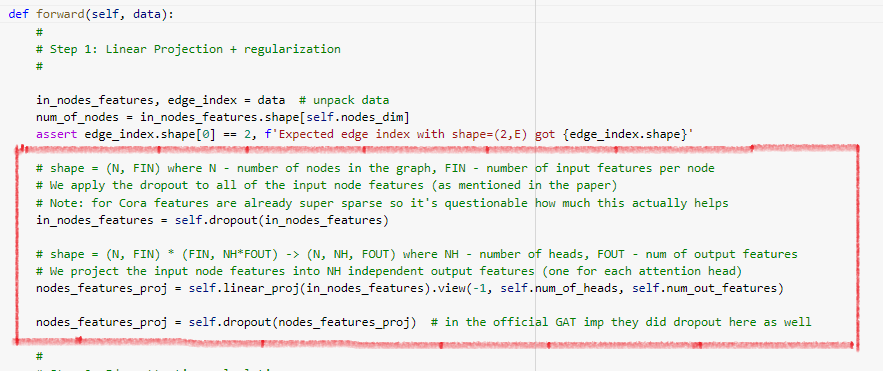
مطابق با آنچه که در مقاله بدان اشاره شده است، به منظور به دست آوردن ویژگی‌های سطح بالاتری از روی ویژگی‌های ورودی، بر روی آن‌ها حداقل یک تبدیل خطی اعمال خواهیم کرد. ماتریس خطی (معادل با در مقاله)، وزن‌های توجه (معادل با در مقاله) و بایاس (در مقاله بدان اشاره‌ای نشده اما در کد اصلی [1] آن پیاده‌سازی شده است)، وزن‌های قابل آموزش این لایه هستند. از آنجا که چندین هد می‌تواند مورد استفاده قرار گیرد بجای اینکه برای هرکدام ماتریس وزن جداگانه تعریف کنیم، می‌توان به شکل زیر همه را در یک لایۀ مشترک جای داد.



همانطور که در روابط زیر مشاهده می‌شود به تعداد هدها ماتریس وزن مستقل اعمال می‌شود، در نتیجه به تعدادی معادل با num\_of\_heads ماتریس وزن خواهیم داشت (که در پیاده‌سازی همگی با یکدیگر یک ماتریس واحد را تشکیل می‌دهند).

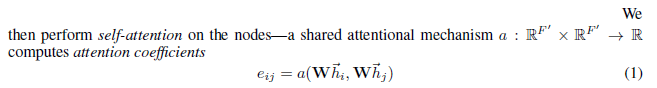
در هنگام آموزش رو به جلو، ابتدا بر روی بردارهای ویژگی رئوس یک Dropout صورت می‌گیرد و سپس تبدیل خطی که در بالا تعریف شد بر روی آن‌ها اعمال شده و نهایتاً یکبار دیگر از تکنیک Dropout استفاده می‌شود. استفاده از Dropout قبل و بعد از تبدیل خطی مطابق با مقاله و پیاده‌سازی اصلی آن صورت گرفته و بیش‌برازش زودهنگام مدل‌های گرافی به عنوان یکی از دلایل آن ذکر شده است. کد پیاده‌سازی این مراحل در ادامه آورده شده است:



همانطور که مشاهده می‌شود بردارهای ویژگی گراف ورودی با ابعاد *(اندازه ویژگی ورودی، تعداد راس‌ها)* پس از تبدیل خطی دارای ابعادی معادل *(اندازه ویژگی خروجی، تعداد هدها، تعداد راس‌ها)* خواهد شد.

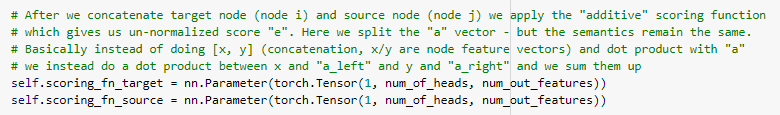
**1-2) محاسبه‌ی ضرایب توجه**

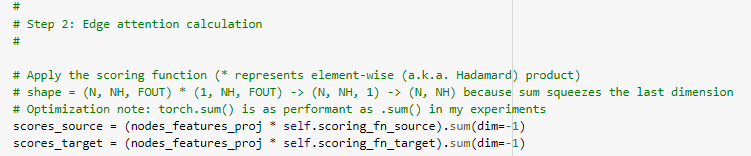
در این قسمت به محاسبۀ ضرایب توجه نرمال نشده یعنی در مقاله می‌پردازیم.



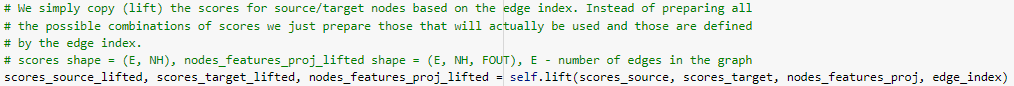
در کد پیاده‌سازی شده به جای اینکه بردار را در الحاق دو بردار و ضرب داخلی کنیم، را به دو نیمه‌ی چپ و راست تقسیم کرده و نیمه‌ی چپ را در و نیمه‌ی راست را در ضرب داخلی می‌کنیم و سپس نتایج حاصل شده را با هم جمع خواهیم کرد. مشخص است که نتیجه نهایی هر دو رویکرد مشابه با یکدیگر خواهد بود.

لازم به ذکر است که ابعاد هر نیمه از بردار با *(اندازه ویژگی خروجی، تعداد هدها، 1)* برابر است و پس از ضرب داخلی با حاصل تبدیل خطی با ابعاد *(اندازه ویژگی خروجی، تعداد هدها، تعداد راس‌ها)*، *نهایی دارای ابعاد (تعداد هدها، تعداد راس‌ها) خواهد شد.*

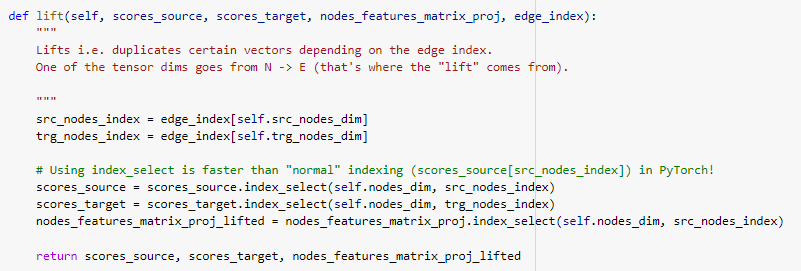




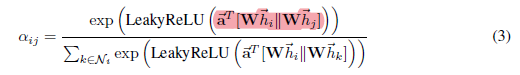
با این حال، باید توجه داشت که در محاسبۀ دو ماتریس پیشین تمام راس‌ها حضور دارند. حال آنکه به ازای هر راس ()، نیاز است را صرفاً براساس راس‌های همسایه () محاسبه کنیم.



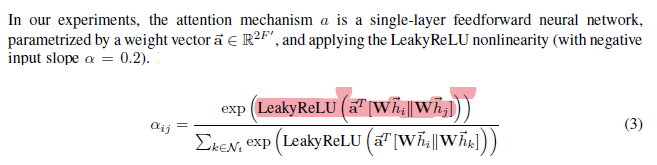
از همین روی، همانطور که در کد بالا مشاهده می‌شود تابعی به نام *lift* مورد استفاده قرار گرفته که به منظور محاسبه‌ی ضرب‌های داخلی و با توجه به یالهای بین راس‌های و پیاده‌سازی شده است.



مقدار  *که تاکنون نسبت به محاسبۀ آن پرداختیم* مربوط به قسمت رنگ شده در فرمول زیر است:



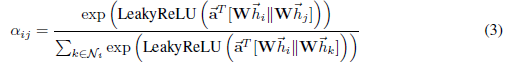
حال جهت رسیدن به ضرایب توجه نرمال شده، ابتدا حاصل جمع دو و را از یک تابع غیرخطی‌ساز LeakyReLU عبور می‌دهیم.

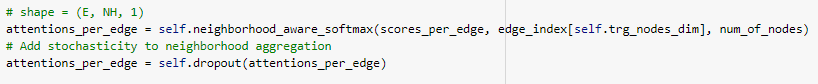




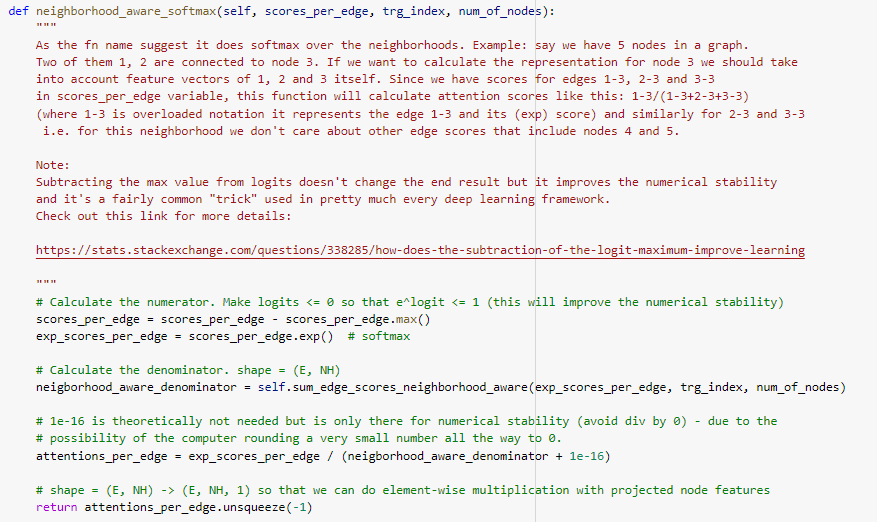


در نهایت با اعمال تابع softmax روی هر گره با توجه به همسایگانش (که خود گره را نیز شامل می‌شود) ضرایب توجه نرمال شده را به دست می‌آوریم:

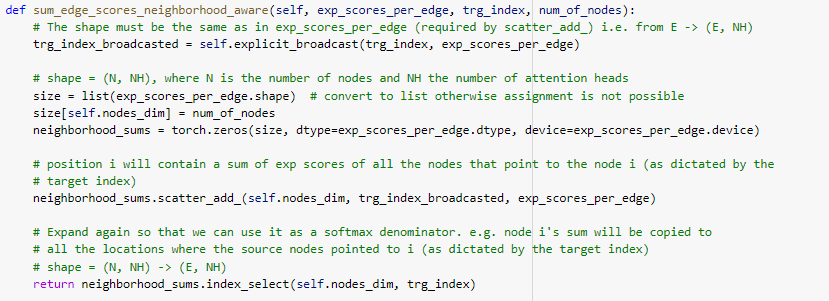


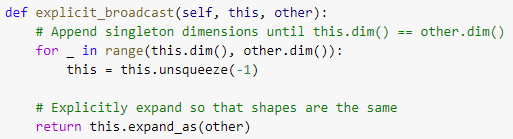


پس از به دست آوردن softmax روی خروجی آن که در گام بعدی استفاده می‌شود dropout اعمال شده است. تابعی که به منظور اعمال softmax با توجه به همسایگان یک راس و به دست آوردن ضرائب توجه نرمال طراحی شده به شرح زیر است.

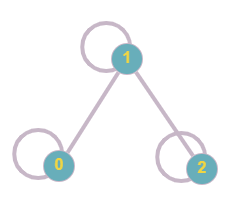


باید توجه داشت که در مخرج کسر، تنها آن راس‌هایی در نظر گرفته خواهند شد که مستقیماً (با یک یال) به راس موردنظر متصل هستند. به منظور محاسبۀ مخرج کسر برای یک راس مشخص تابع زیر مورد استفاده قرار می‌گیرد. ابتدا به دلیل اینکه چندین هد مورد استفاده قرار گرفته لازم است ابعاد بردار شاخص trg\_index را از *(تعداد یال‌ها)* به *(تعداد هدها، تعداد یال‌ها)* تبدیل کنیم. سپس به ازای هر راس، قصد داریم به کمک تابع scatter\_add تورچ، مجموع امتیاز نمایی راس‌هایی که بدان اشاره می‌کنند (همسایه‌ها) را محاسبه کنیم بنابراین ماتریس نهایی دارای ابعادی معادل با *(تعداد هدها، تعداد راس‌ها)* خواهد بود. با این حال از آنجا که در حال محاسبۀ softmax بر اساس یال‌ها بودیم، یکبار دیگر این ماتریس را بسط داده و در هرجا که راسی به راس موردنظر ما اشاره می‌کند، حاصل جمع بدست آمده برای راس موردنظر را کپی خواهیم کرد.





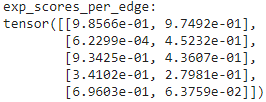
* به منظور درک بهتر عملکرد این تابع به شرح یک مثال از آن می‌پردازیم:



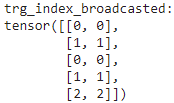
فرض کنیم سه راس و دو هد داریم و یال‌های بین رئوس به شرح زیر است:



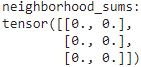
بنابراین ماتریس امتیازهای نمایی طبق یال‌ها دارای ابعاد *(تعداد هدها، تعداد یال‌ها)* یعنی *(2، 5)* و به عنوان مثال به شرح زیر خواهد بود:



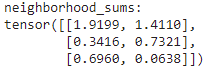
در گام اول ابعاد بردار شاخص هدف را بسط داده و آن را به ماتریسی با ابعاد *(تعداد هدها، تعداد یال‌ها)* یعنی *(2، 5)* تبدیل می‌کنیم.



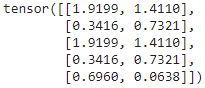
از آنجا که قصد داریم به ازای هر راس، جمع امتیاز رئوس همسایۀ آن را محاسبه کنیم، ماتریس حاصل دارای ابعاد *(تعداد هدها، تعداد راس‌ها)* یعنی *(2، 3)* خواهد بود. بنابراین یک ماتریس اولیه با مقدار صفر و این ابعاد ایجاد می‌کنیم.



اکنون می‌توان با توجه به ماتریس trg\_index\_broadcasted، نسبت به جمع امتیازهای همسایه‌های هر راس اقدام کرد. به عنوان مثال طبق این ماتریس یال‌های اول و سوم به راس شماره صفر متصل هستند، بنابراین جمع مقادیر امتیاز آن‌ها که برابر 1.9199 برای هد اول و 1.4110 برای هد دوم است را در ردیف مربوط به راس شماره صفر (ردیف نخست) در ماتریس neighborhood\_sums ذخیره خواهیم کرد. پس از تکمیل این ماتریس به طریق مشابه، خواهیم داشت:

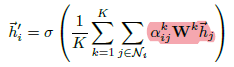
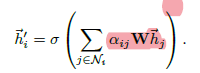


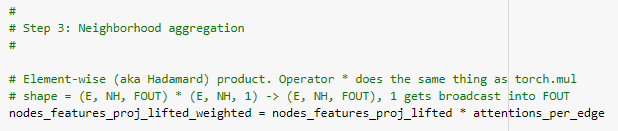
اکنون از آنجا که قصد داریم ماتریس نهایی دارای ابعاد *(تعداد هدها، تعداد یال‌ها)* یعنی *(2، 5)* باشد، طبق trg\_index\_broadcasted هرجا که به راسی اشاره شده، مجموع امتیاز آن را طبق نتایج ذخیره شده در neighborhood\_sums خواهیم نوشت. خروجی نهایی برابر است با:



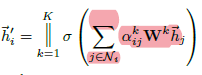
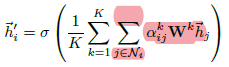
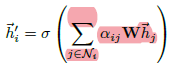
**1-3) تجمیع همسایه‌ها**

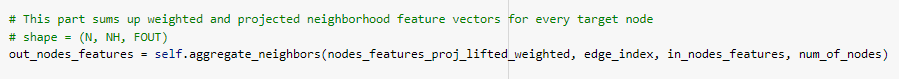
بعد از به دست آوردن ضرایب توجه نرمال شده، جهت محاسبه‌ی بردار ویژگی جدید هر گره یا همان بردار ویژگی سطح بالاتر، این ضرایب را در بردار ویژگی تبدیل‌یافته ضرب می‌کنیم. خروجی نهایی این مرحله دارای ابعاد *(اندازه ویژگی خروجی، تعداد هدها، تعداد یال‌ها)* خواهد بود.



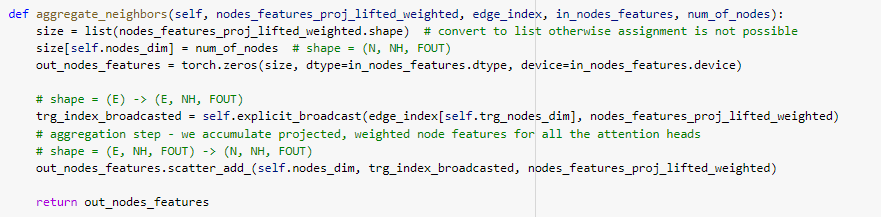


در آخرین قسمت از این مرحله برای به دست آوردن بردار ویژگی جدید هر گره، بردار ویژگی تبدیل‌ یافته‌ی وزن‌دار همسایگان آن را با هم جمع می‌کنیم که توسط تابع aggregate\_neighbors انجام شده است.

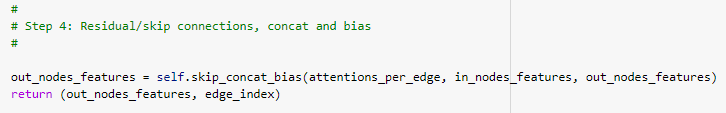




پیاده‌سازی تابع aggregate\_neighbors به صورت زیر بوده و عملکردی مشابه با تابع sum\_edge\_scores\_neighborhood\_aware دارد که پیش از این معرفی شد. خروجی نهایی این تابع دارای ابعادی برابر با *(اندازه ویژگی خروجی، تعداد هدها، تعداد راس‌ها)* خواهد بود.



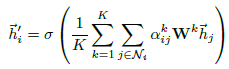
**1-4) بایاس، الحاق و skip\_connection**



در این مرحله با توجه به متغیرهای تنظیم شده درباره سه مورد زیر تصمیم گرفته و در صورت نیاز آن‌ها را اعمال می‌کنیم:

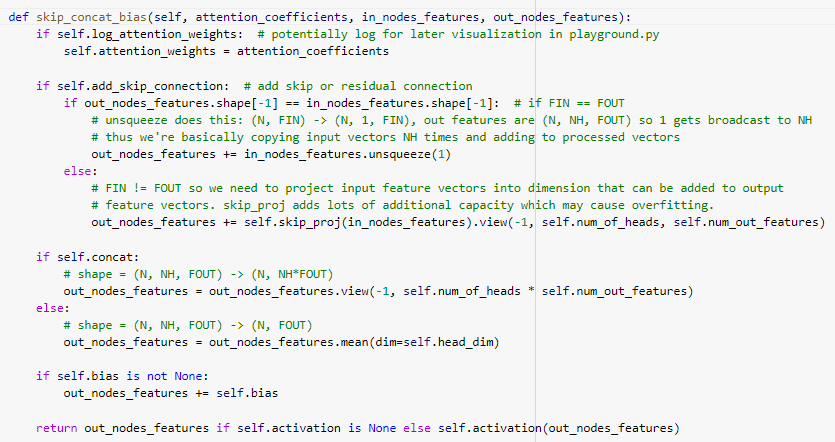
* آیا از تکینک skip connection استفاده کنیم؟
* خروجی هدها را با هم الحاق کنیم یا از آن‌ها میانگین بگیریم؟
* از bias در لایۀ GAT استفاده کنیم یا خیر؟

پس از طی این مراحل، خروجی نهایی لایۀ GAT تولید خواهد شد.



استفاده یا عدم استفاده از log\_attention\_weights مربوط به مصورسازی است و ضرایب توجه را جهت استفاده بصری از آن‌ها ذخیره می‌کند.

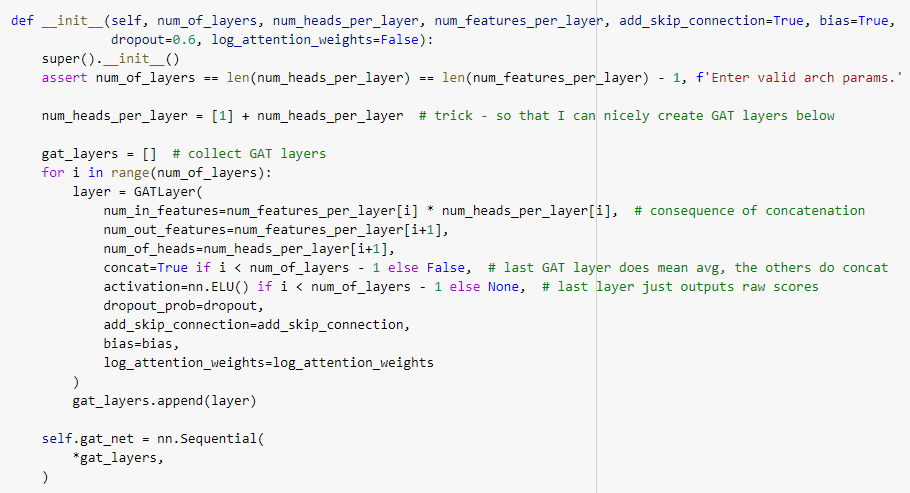
استفاده یا عدم استفاده از skip connection، بایاس و همچنین استفاده از الحاق برای خروجی هدها یا میانگین‌گیری از آنها همگی در ابتدا توضیح داده شده‌اند و فقط پیاده‌سازی آن‌ها در تابع زیر انجام شده است. در صورتیکه از تکنیک skip connection استفاده شود ویژگی‌های ورودی را با نتایج بدست آمده توسط لایه جمع خواهیم کرد، که البته ممکن است برای این کار نیاز به تغییر ابعاد ورودی داشته باشیم.



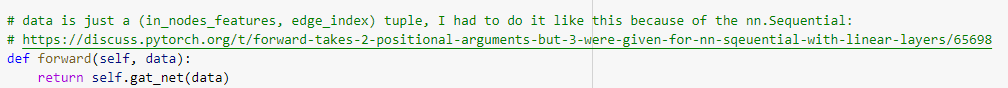
**2) مدل GAT**



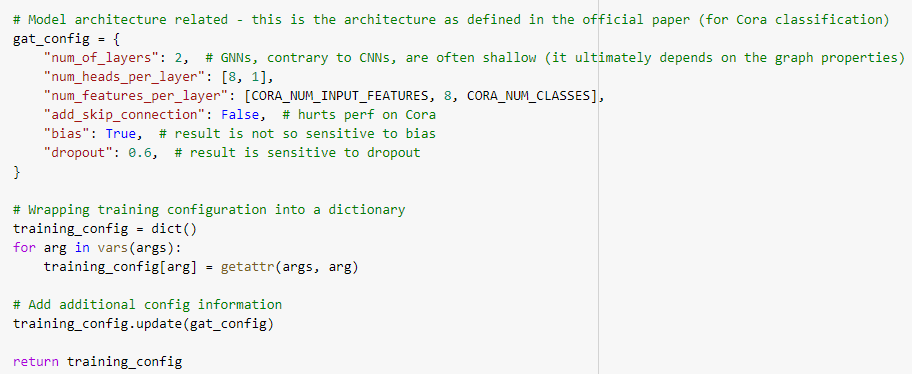
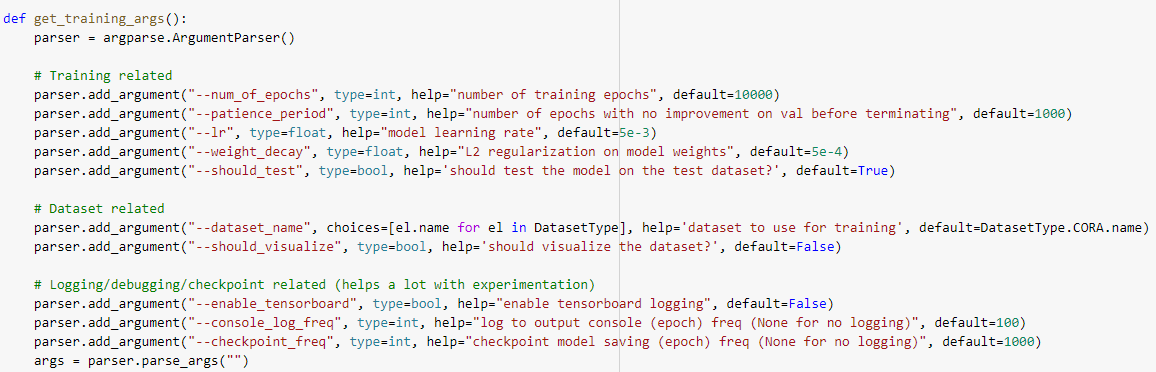
برای پیاده‌سازی مدل با استفاده از چندین لایۀ GAT، یک کلاس با همین نام تعریف شده که از ویژگی‌های یک مدل تورچ بهره می‌برد. در تعریف این مدل علاوه بر پارامترهای لایه‌های GAT پارامتری به نام num\_of\_layers برای مشخص کردن تعداد لایه‌های GAT تعریف شده است. تعداد لایه‌های GAT به مشخصات گراف بستگی دارد و با افزایش تعداد لایه‌ها، هر راس می‌تواند تحت تاثیر راس‌های دورتری در گراف نیز قرار بگیرد.



تعداد ویژگی‌های ورودی هر لایه برابر حاصل‌ضرب تعداد خروجی لایه قبل و تعداد هدهای آن لایه است. در تمام لایه‌ها به جز لایه‌ی آخر خروجی هدها با هم concat می‌شوند و در لایه‌ی آخر میانگین آن‌ها حساب می‌شود. برای تمام لایه‌ها به جز لایه‌ی آخر از تابع فعالسازی ELU استفاده شده است و در لایه‌ی آخر هیچ نوع‌ تابع فعالسازی را اعمال نکردیم تا امتیاز نهایی تولید شود. در نهایت مدلی Sequential با کنار هم قرار گرفتن لایه‌ها ساخته می‌شود.



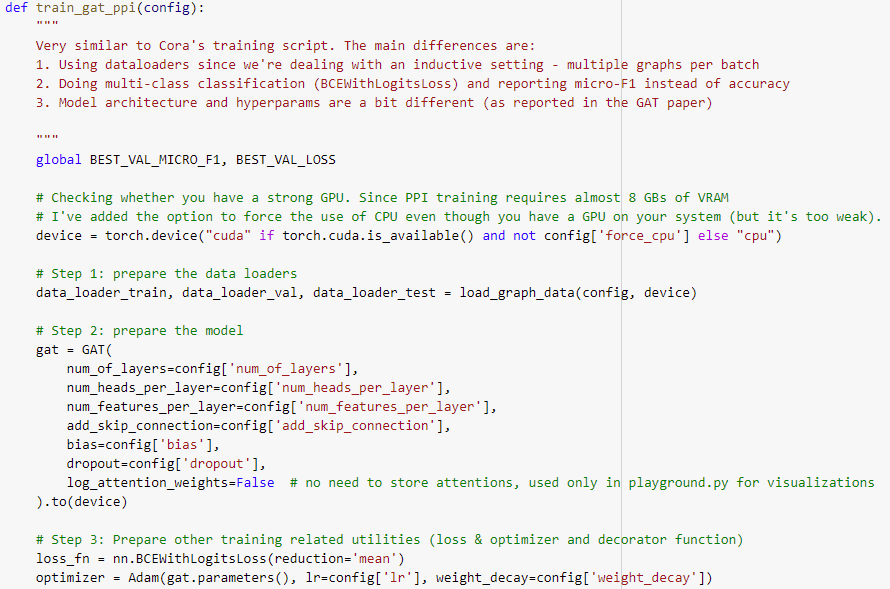
به وسیله‌ی تابع فوق مدل ساخته شده روی داده‌ی ورودی (یعنی گراف) اعمال می‌شود و خروجی لایه آخر به حل مسئله (مثلا طبقه‌بندی) کمک می‌کند.



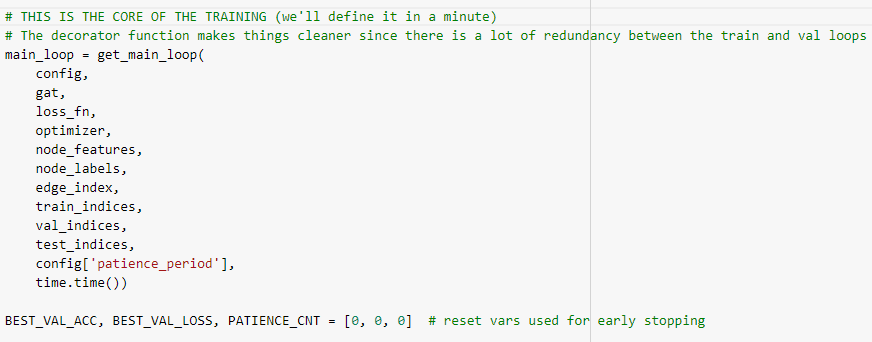
در تابع get\_training\_args پارامترهای (هایپرپارامترهای) آموزش مثل تعداد دورها[[3]](#footnote-3)، نرخ آموزشی[[4]](#footnote-4)، زمان صبر[[5]](#footnote-5)، رگولاریزیشن-L2 [[6]](#footnote-6)و پارامترهای مدل مقداردهی شده‌اند. این پارامترها به عنوان کانفیگ اولیه به تابع train\_gat داده شدند.

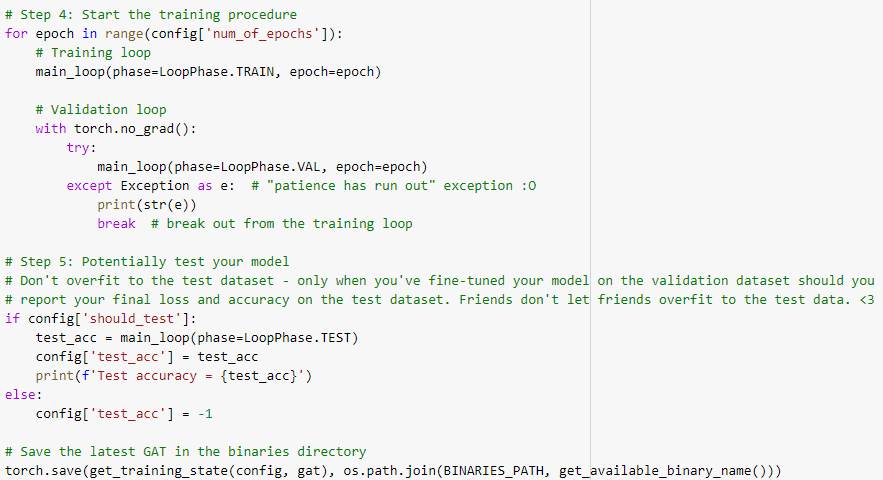
در تابع train\_gat علاوه بر کنترل دسترسی به CPU یا GPU و بارگذاری دیتا، پارامترها و لایه GAT بر روی آن‌ها، در تنظیمات Transductive از تابع ضرر Crossentropy و در تنظیمات Inductive از تابع ضرر BCEWithLogitsLoss استفاده شده است. در ضمن در هر دو تنظیمات بهینه‌ساز Adam برای آموزش انتخاب شده‌است. همچنین مطابق با آنچه در مقاله گزارش داده شده برای تنظیمات Transductive از معیار Accuracy و تنظیمات Inductive از معیار F1-score استفاده خواهد شد.

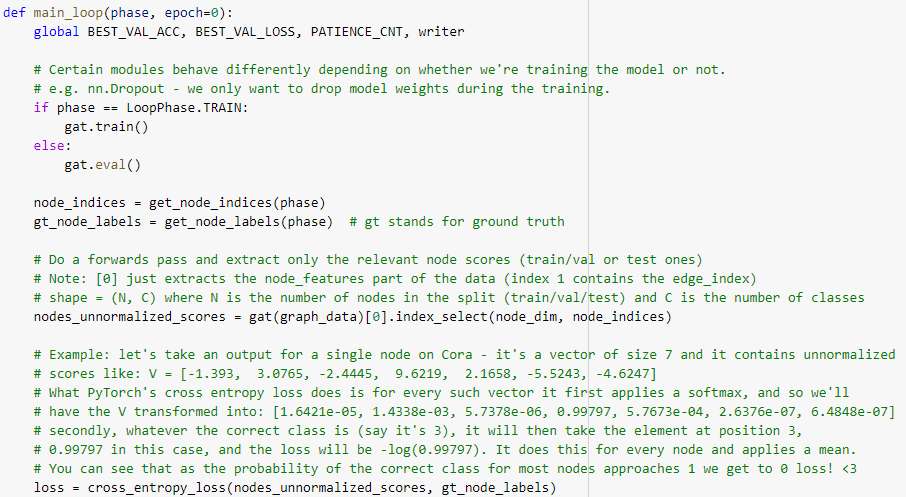




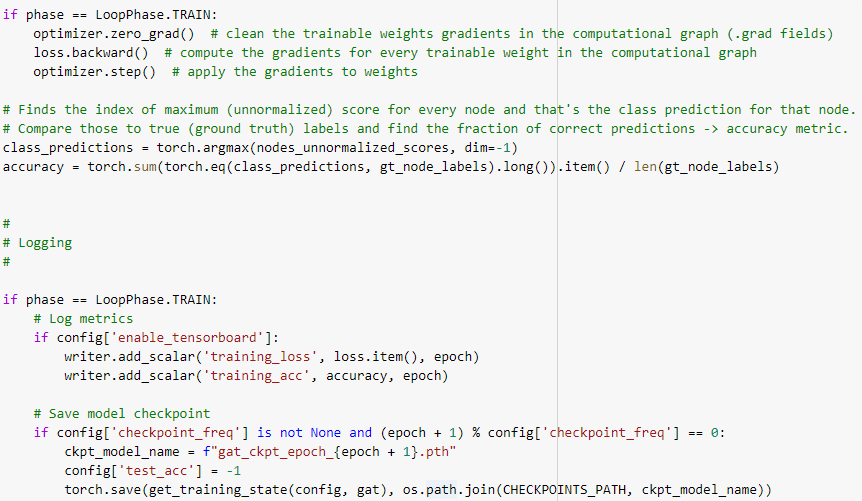
در تابع train\_gat و با استفاده از تابع get\_main\_loop مدل آموزش می‌بیند و عملکرد آن روی داده‌های آموزش، اعتبارسنجی و آزمون محاسبه می‌شود. به ازای هر دور، یکبار تابع main\_loop برای آموزش مدل، و یکبار هم در فاز اعتبارسنجی فراخوانی خواهد شد. وظیفۀ این تابع، آموزش یا اعتبارسنجی مدل GAT و محاسبۀ هزینه براساس امتیازات خروجی مدل است.

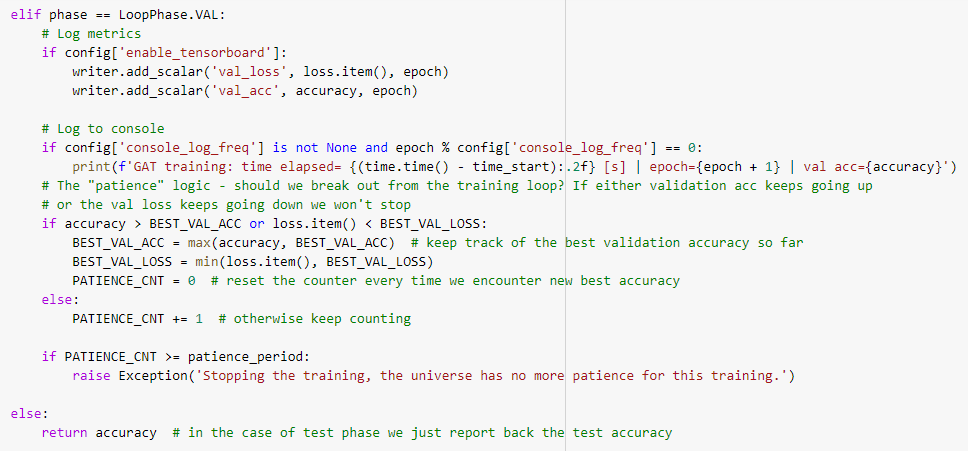






همانطور که گفته شد خروجی نهایی مدل GAT به شکل امتیازهای نرمال‌نشده است. این امتیازها به تابع ضرر داده شده و هزینۀ نهایی محاسبه خواهد شد. همچنین در تنظیمات Transductive از بین امتیازهایی که برای هر راس بدست آمده، شاخص مربوط به امتیاز بیشینه برابر کلاس پیش‌بینی شده برای آن راس خواهد بود. در تنظیمات Inductive نیز از آنجا که از تابع هزینه Sigmoid استفاده شده است، اگر امتیاز خروجی بیشتر از 1 باشد یعنی که Sigmoid ارزش بیشتر از 0.5 را داشته و کلاس مربوطه، کلاس 1 و در غیر اینصورت کلاس 0 خواهد بود.





**3) مجموعه‌های داده**

در مقالۀ اصلی GAT [2] چهار مجموعه داده مورد آزمایش قرار گرفته که دو مورد از آن به عنوان نمونه‌هایی از رویکرد یادگیری Transductive و Inductive جهت بررسی صحت عملکرد مدل پیاده‌سازی شده مطابق با نتایج مقاله مورد ارزیابی قرار گرفته است. مشخصات این دو مجموعه داده به شرح زیر است:

**3-1) Cora**

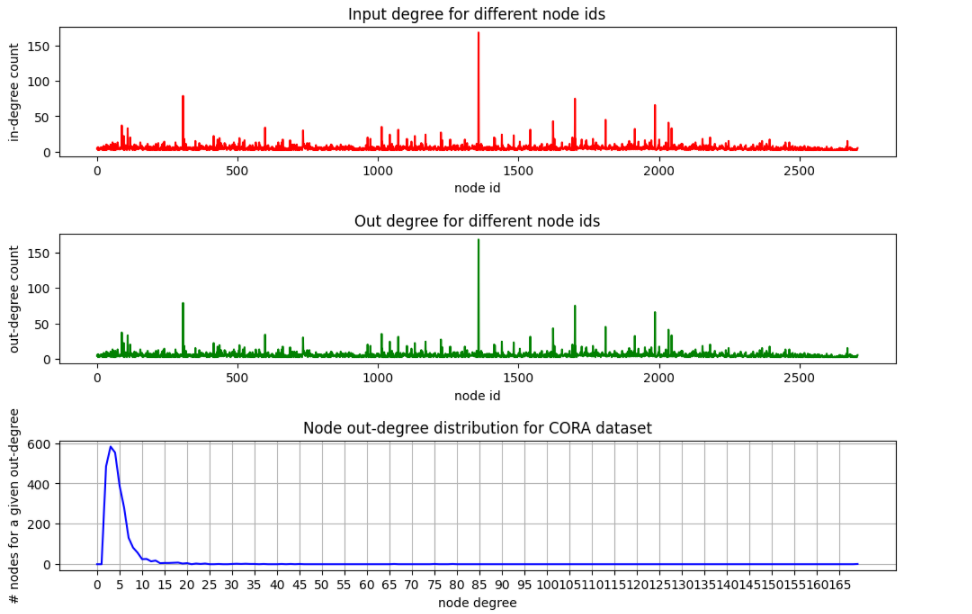
مجموعه داده Cora شامل مقالات مربوط به یادگیری ماشین است. این مقالات به هفت کلاس طبقه بندی می‌شوند. در این مجموعه داده، گره‌ها به اسناد یا مقالات و یال‌ها به استناد‌[[7]](#footnote-7) (بدون جهت) مربوط هستند. ویژگی‌های گره مطابق با اجزای نمایش BOW[[8]](#footnote-8) یک مقاله هستند. پس از تبدیل کلمات به ریشه‌ی آن‌ها[[9]](#footnote-9) و حذف کلمات توقف با یک مجموعه‌ی واژگان شامل 1433 کلمه منحصر به فرد مواجه هستیم. مقالات به گونه‌ای انتخاب شده‌اند که در پیکره نهایی هر مقاله حداقل یک مقاله دیگر به آن استناد کند یا از آن استناد شود. در کل پیکره 2708 مقاله وجود دارد.

هر گره دارای یک برچسب کلاس است. ما اجازه می‌دهیم فقط 20 گره در هر کلاس برای آموزش استفاده شود - با این وجود ، طبق به تنظیمات Transductive، الگوریتم آموزش به بردار ویژگی همه گره‌ها دسترسی دارد.

آمار کلی این مجموعه داده مطابق با جدول زیر است:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **تعداد راس‌ها** | **تعداد یال‌ها** | **اندازۀ ویژگی هر راس** | **نمونه‌های آموزشی** | **نمونه‌های اعتبارسنجی** | **نمونه‌های آزمون** |
| 2708 | 5429 | 1433 | 140 | 500 | 1000 |

نمودارهای میزان درجه ورودی هر راس، درجه خروجی هر راس و توزیع درجات خروجی در شکل 1-1-3 نمایش داده شده است. همانطور که مشاهده می‌شود، دو نمودار اول مشابه با همدیگر بوده و این به دلیل بدون جهت بودن گراف است (با اینکه به طور طبیعی باید به عنوان یک گراف جهت‌دار مدل شود). بعضی از گره‌ها دارای تعداد زیادی یال هستند (قله در وسط)، اما بیشتر گره ها دارای یال‌های بسیار کمتری هستند. در نمودار سوم نیز به خوبی قابل مشاهده است که رئوسی با درجات بسیار بالا، نادر هستند و پراکندگی در تعداد یال‌های کمتر، بیشتر است.



**شکل 1-1-3:** نمودار جزئیات مجموعه داده Cora

**3-2) Protein-Protein Interaction (PPI)**

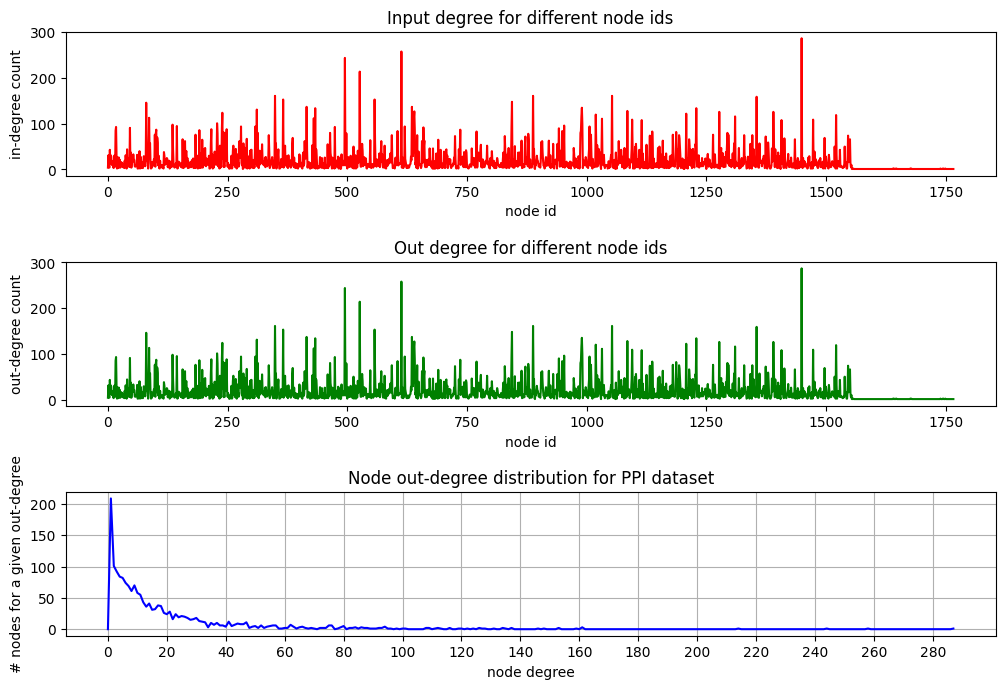
در مسئله‌ای که برای این مجموعه داده در نظر گرفته می‌شود بجای اینکه یادگیری به منظور کسب دانشی درباره ساختار جامعه انجام گیرد بر نقش راس‌ها در آن تاکید دارد، و هدف اصلی دستیابی به عمومیت[[10]](#footnote-10) بر روی گراف‌ها است. در اینجا قصد بر این است تا نقش پروتئین‌ها در گراف‌های مختلف تعاملات پروتئینی-پروتئینی (PPI)، که هر گراف نمایانگر یک بافت متفاوت انسانی است، رده‌بندی شود [3].

مجموعه ژن‌های موضعی[[11]](#footnote-11)، ژن‌های موتیف[[12]](#footnote-12) و امضاهای ایمونولوژیک[[13]](#footnote-13) به عنوان ویژگی و مجموعه آنتولوژی ژن‌ها[[14]](#footnote-14) به عنوان برچسب (مجموعاً 121تا) در نظر گرفته شده است، که همگی از دیتابیس Molecular Signatures Database [4] استخراج شده‌اند.

در مجموع 24 گراف در این مجموعه داده موجود است که 20 گراف آن به منظور آموزش مدل استفاده شده و میانگین امتیازات F1 نیز بر روی 2 گراف آزمون گزارش شده است (2 گراف دیگر نیز به منظور اعتبارسنجی مورداستفاده قرار گرفته است). میانگین درجه هر راس برابر با 28.8 است و به هر راس می‌توان از میان 121 کلاس ممکن، چندین کلاس نیز نسبت داد (رده‌بندی چند-برچسبی[[15]](#footnote-15)).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **تعداد راس‌ها** | **تعداد یال‌ها** | **اندازۀ ویژگی هر راس** | **نمونه‌های آموزشی** | **نمونه‌های اعتبارسنجی** | **نمونه‌های آزمون** |
| 56944 | 1644208 | 50 | 44906 | 6514 | 5524 |

لازم به ذکر است که به دلیل حجم بالای داده در این مجموعه داده، گراف‌ها به شکل دسته‌های دوتایی به مدل داده شده و آموزش پیش خواهد رفت. در نمودارهای زیر، میزان درجات ورودی، درجات خروجی و توزیع آن‌ها برای یکی از گراف‌ها نمایش داده شده است. همانطور که از نمودار اول و دوم شکل 1-2-3 مشاهده می‌شود بسیاری از راس‌ها در مقابله با مجموعه داده Cora دارای تعداد یال بیشتری هستند. همچنین طبق نمودار سوم می‌توان گفت که اکثر راس‌ها دارای 1 تا 20 یال هستند.



**شکل 1-2-3:** نمودار جزئیات مجموعه داده PPI

علاوه بر این، مدل پیاده‌سازی شده بر روی دو مجموعه داده جدید نیز مورد آزمایش قرار گرفته است که جزئیات هرکدام از آن‌ها به شرح زیر است:

**3-3) Reddit**

شبکه اجتماعی Reddit یکی از بزرگترین انجمن‌های اینترنتی است که کاربران به انتشار محتوا و نظرات خود پیرامون بسیاری از موضوعات در آن می‌پردازند. در مسئله‌ای که برای این شبکه مطرح می‌شود سعی بر این است تا پیش‌پینی کنیم که هر پست Reddit به کدام جامعه (موضوع) تعلق دارد.

در سال 2014 یک مجموعه داده گرافی از پست‌های این شبکه ساخته شد. برچسب‌های هر راس در این گراف، جامعه یا subreddit ای است که پست بدان تعلق دارد. گراف ساخته شده به شکل پست-به-پست بوده و دو پست در صورتی با یک یال به یکدیگر متصل هستند که کاربر مشابهی بر روی هر دوی آن‌ها دیدگاه خود را ثبت کرده باشد [5].

در مجموع این مجموعه داده شامل 232965 پست با میانگین درجه 492 است که در مقایسه با سایر مجموعه داده‌های گرافی میزان درجه قابل توجهی به شمار می‌آید. داده‌های مربوط به 20 روز نخست به عنوان مجموعۀ آموزشی و سایر روزها به عنوان آزمایش (30% آن به عنوان اعتبارسنجی) در نظر گرفته شده است.

بردار ویژگی هر راس مطابق با بردارهای کلمات 300 بعدی GloVe CommonCrawl ساخته شده است. به ازای هر پست، ویژگی‌های زیر با یکدیگر الحاق شده‌اند و بردار نهایی را تشکیل داده‌اند:

* میانگین تعبیۀ عنوان پست
* میانگین تعبیۀ تمام دیدگاه‌های پست
* امتیاز پست
* تعداد دیدگاه‌های پست

آمار کلی این مجموعه داده مطابق با جدول زیر است:

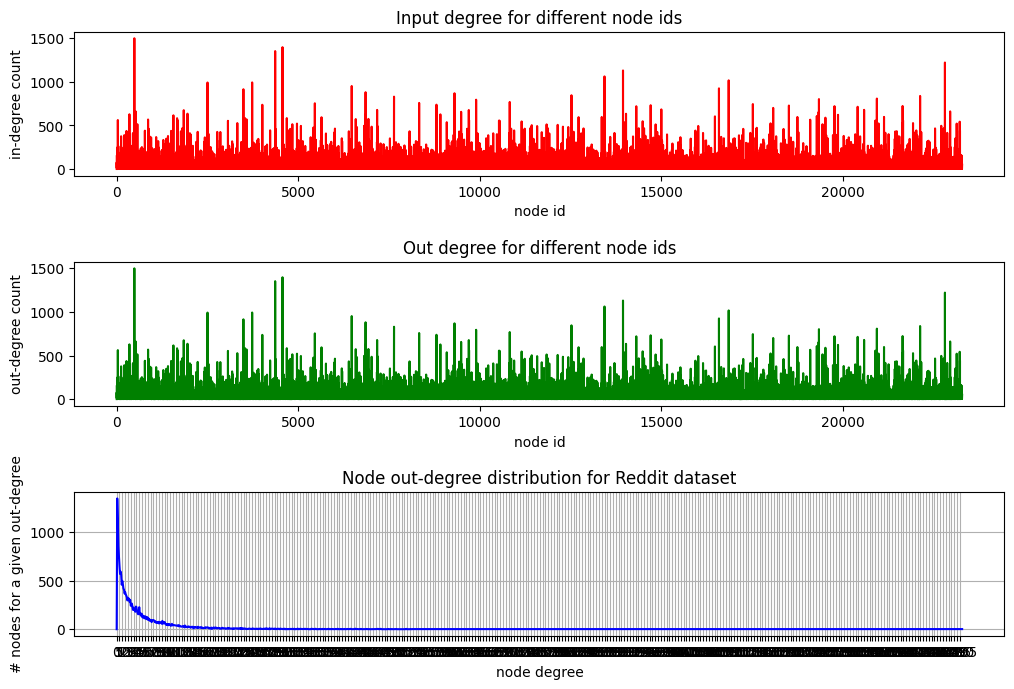
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **تعداد راس‌ها** | **تعداد یال‌ها** | **اندازۀ ویژگی هر راس** | **نمونه‌های آموزشی** | **نمونه‌های اعتبارسنجی** | **نمونه‌های آزمون** |
| 232965 | 114615892 | 602 | 153431 | 23831 | 55703 |

با این حال، به دلیل اندازۀ بسیار بزرگ مجموعۀ داده و عدم دسترسی به سخت‌افزار کافی برای آموزش مدل بر روی این حجم از داده، در آزمایش‌های صورت گرفته تنها به ده درصد آن اکتفا شده و آمار آن به شرح موجود در جدول زیر است:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **تعداد راس‌ها** | **تعداد یال‌ها** | **اندازۀ ویژگی هر راس** | **نمونه‌های آموزشی** | **نمونه‌های اعتبارسنجی** | **نمونه‌های آزمون** |
| 23296 | 1052560 | 602 | 15343 | 2383 | 5570 |

باید توجه داشت که در برخی مقالات پیشین که بر روی این مجموعه داده نیز آزمایش خود را انجام داده‌اند از رویکرد یادگیری Inductive استفاده شده است. این بدین معنی است که در طول فرآیند آموزش مدل، رئوس مربوط به نمونه‌های آزمایشی به کلی نادیده گرفته می‌شوند. با این حال از آنجا که در این آزمایش مجموعه داده کوچک شده، آموزش مدل به شکل Transductive و مشابه با آنچه که بر روی مجموعه Cora اتفاق افتاد، انجام خواهد شد.

نمودارهای میزان درجه ورودی هر راس، درجه خروجی هر راس و توزیع درجات خروجی در شکل 1-3-3 نمایش داده شده است. همانطور که مشاهده می‌شود، دو نمودار اول مشابه با همدیگر بوده و این به دلیل بدون جهت بدون گراف است. در نمودار سوم نیز به خوبی قابل مشاهده است که رئوسی با درجات بسیار بالا، نادر هستند و پراکندگی در تعداد یال‌های کمتر، بیشتر است.



**شکل 1-3-3:** نمودار جزئیات مجموعه داده Reddit

**3-4) Wiki\_CS**

مجموعه داده‌ Wiki-CS که در پژوهش [6] معرفی شده است، یک مجموعه داده‌ مبتنی بر ویکی‌پدیا برای آزمودن[[16]](#footnote-16) شبکه های عصبی گرافی است. این مجموعه داده از دسته‌بندی‌های ویکی‌پدیا، به طور خاص 10 کلاس مربوط به شاخه های علوم کامپیوتر[[17]](#footnote-17) ، با اتصال بسیار بالا ساخته شده است. ویژگی‌های گره از متون مقاله‌های مربوطه گرفته شده است. این ویژگی‌ها به عنوان میانگین تعبیه کلمه از پیش آموزش دیده GloVe محاسبه شدند. در نتیجه ویژگی‌های گره 300 بعدی حاصل می‌شود که می‌تواند برای آموزش مدل‌های بزرگ روی GPU یک مزیت باشد. مقالاتی که می‌توانستند به چند کلاس تعلق داشته باشند از این مجموعه داده حذف شده‌اند.

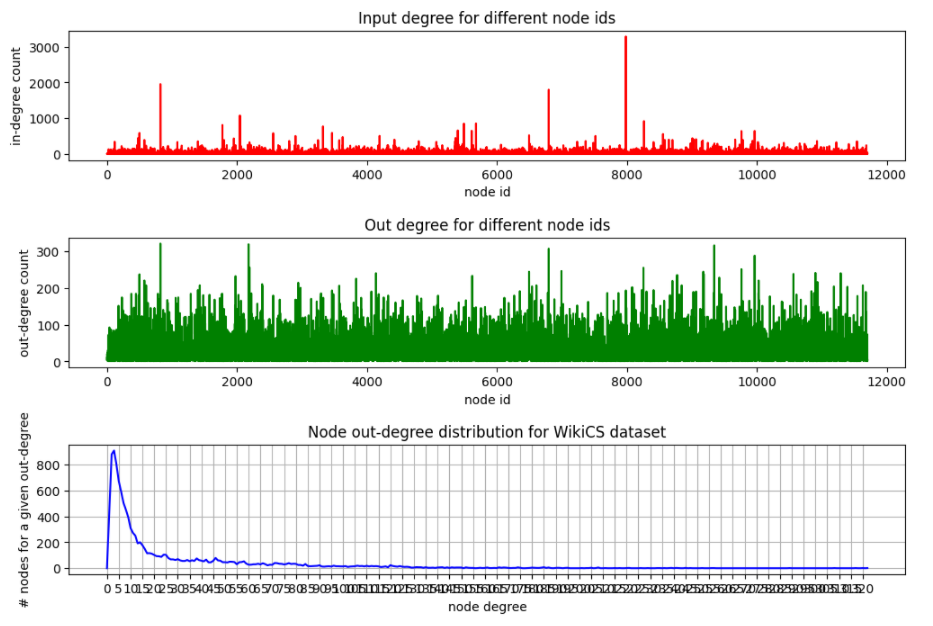
گره‌ها را در هر کلاس به دو مجموعه تقسیم می‌شوند، 50٪ برای مجموعه آزمون و 50٪ به طور بالقوه قابل مشاهده[[18]](#footnote-18). از مجموعه قابل مشاهده ، 20 تقسیم مختلف برای آموزش ، اعتبارسنجی و توقف زودرس[[19]](#footnote-19) ایجاد شده‌است: در هر تقسیم 5٪ از گره‌های هر کلاس برای آموزش ، 22.5٪ برای ارزیابی معیار توقف زودرس مورد استفاده قرار گرفت و 22.5٪ به عنوان مجموعه اعتبارسنجی برای تنظیم هایپر پارامتر استفاده شده‌است.

ما در آزمایشات خود فقط از یکی از 20 تقسیم مختلف برای برای آموزش و اعتبارسنجی استفاده کرده و از داده‌های مربوط به توقف زودرس استفاده نمی‌کنیم.

آمار کلی این مجموعه داده مطابق با جدول زیر است:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **تعداد راس‌ها** | **تعداد یال‌ها** | **اندازۀ ویژگی هر راس** | **نمونه‌های آموزشی** | **نمونه‌های اعتبارسنجی** | **نمونه‌های آزمون** |
| 11701 | 216123 | 300 | 580 | 1769 | 5847 |

نمودارهای میزان درجه ورودی هر راس، درجه خروجی هر راس و توزیع درجات خروجی در شکل 1-4-3 نمایش داده شده است. همانطور که مشاهده می‌شود، نمودار اول و دوم متفاوت هستند و این به دلیل جهت‌دار بودن گراف مربوطه است. در نمودار سوم نیز به خوبی قابل مشاهده است که رئوسی با درجات بسیار بالا، نادر هستند و پراکندگی در تعداد یال‌های کمتر، بیشتر است.



**شکل 1-4-3:** نمودار جزئیات مجموعه داده Wiki\_CS

**4) آزمایش‌ها**

هایپرپارامترهایی که برای آموزش مدل بر روی دو مجموعه داده اصلی Cora و PPI تنظیم شده دقیقاً مطابق با مقاله بوده و برای دو مجموعه داده دیگر نیز تقریباً مشابه با آن‌ها، اما با اندکی تغییر جهت عملکرد بهتر، در نظر گرفته شده‌اند. پارامترهای اصلی تنظیم شده برای هر چهار مجموعه داده در جدول 1-4 خلاصه شده است.

لازم به ذکر است بردار اندازۀ ویژگی هر لایه به شکل ]تعداد کلاس‌ها، ...، تعداد ویژگی‌های ورودی[ مقداردهی می‌شود. به عنوان مثال وقتی این بردار به شکل [602, 8, 41] تنظیم شده است بدین معنی است که لایۀ نخست، بردارهای ویژگی ورودی با اندازه 602 را گرفته و بردارهای ویژگی جدیدی با ابعاد 8 تولید می‌کند. سپس لایۀ دوم نیز بردارهای ابعاد 8 را گرفته و برداری با ابعاد 41 که برابر تعداد کلاس‌های مسئله هست را تولید خواهد کرد.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Cora** | **Wiki\_CS** | **Reddit** | **PPI** |
| **تعداد لایه‌ها** | 2 | 2 | 2 | 3 |
| **تعداد هد به ازای هر لایه** | [8, 1] | [8, 1] | [8, 1] | [4, 4, 6] |
| **اندازه ویژگی هر لایه** | [1433, 8, 7] | [300, 8, 10] | [602, 8, 41] | [50, 64, 64, 121] |
| **Skip connection** | False | False | False | True |
| **بایاس** | True | True | True | True |
| **نرخ حذف تصادفی** | 0.6 | 0.3 | 0.3 | 0.0 |
| **بیشینه تعداد دورها** | 10000 | 10000 | 10000 | 200 |
| **زمان انتظار** | 1000 | 1000 | 1000 | 100 |
| **نرخ یادگیری** | 5e-3 | 5e-3 | 5e-3 | 5e-3 |
| **کاهش وزن** | 5e-4 | 5e-4 | 5e-4 | 0 |

**جدول 1-4:** هایپرپارامترهای تنظیم شده براساس هر مجموعه داده

**5) نتایج**

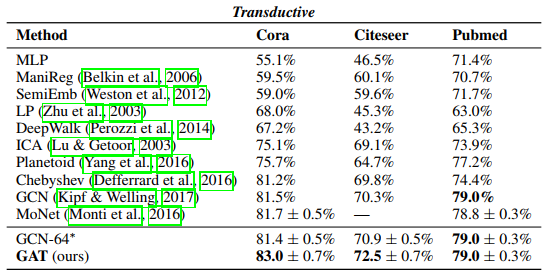
پس از آموزش مدل با توجه به هایپرپارامترهای معرفی شده در بخش پیشین، نتایج حاصل با رویکرد Transductive در جدول 1-5 و با رویکرد Inductive در جدول 2-5 آورده شده است. با مقایسۀ نتایج بدست آمده بر روی دو مجموعه داده Cora و PPI با نتایج مقالۀ اصلی در جدول 3-5 و 4-5 می‌توان به صحت عملکرد مدل پیاده‌سازی شده پی برد. علاوه بر این، در مقایسه عملکرد مدل آموزش داده شده بر روی مجموعه داده Wiki\_CS با نتایجی که مقالۀ اصلی آن منتشر کرده (جدول 5-5)، بهبود حدود 0.6 درصدی دقت آزمون نیز مشاهده می‌شود.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **مجموعه داده** | **آخرین دور** | **زمان (ثانیه)** | **دقت اعتبارسنجی** | **دقت آزمون** |
| Cora | 1101 | 127.08 | 81.60 | 82.20 |
| Wiki\_CS | 1501 | 24.30 | 79.31 | 78.24 |
| Reddit | 5701 | 392.49 | 92.15 | 90.95 |

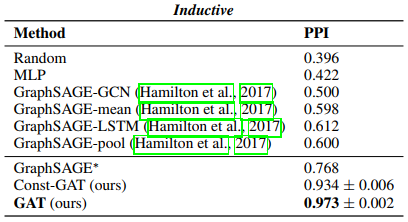
**جدول 1-5:** نتایج آموزش مدل GAT با رویکرد Transductive

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **آخرین دور** | **زمان (ثانیه)** | **F1 آموزش** | **F1 اعتبارسنجی** | **F1 آزمون** |
| 191 | 392.58 | 98.85 | 96.62 | 98.02 |

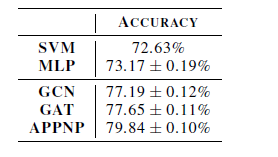
**جدول 2-5:** نتایج آموزش مدل GAT با رویکرد Inductive



**جدول 3-5:** نتایج آموزش مدل GAT با رویکرد Transductive برگرفته از مقالۀ اصلی



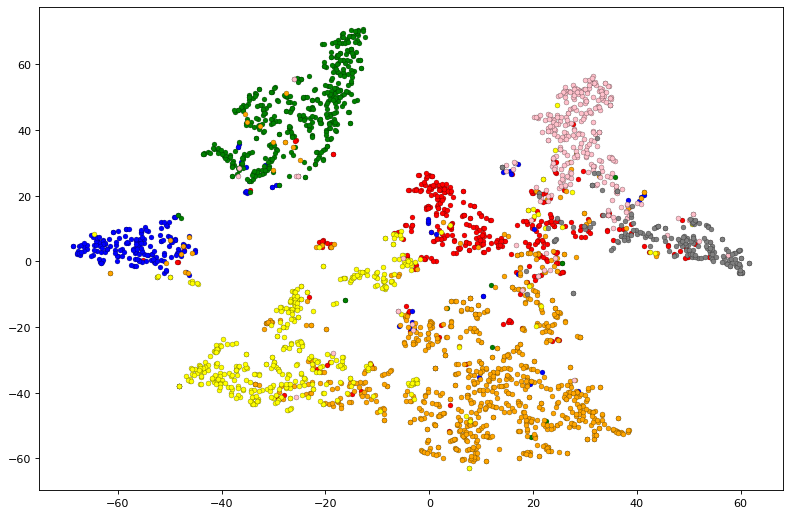
**جدول 4-5:** نتایج آموزش مدل GAT با رویکرد Inductive برگرفته از مقالۀ اصلی



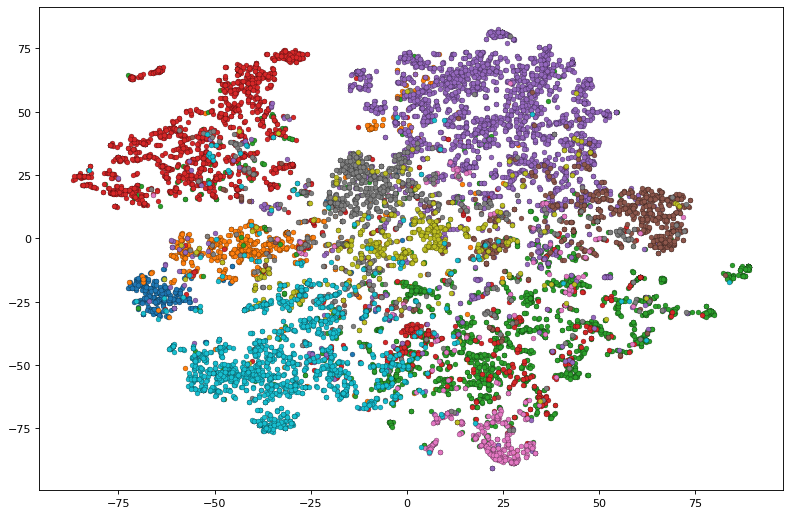
**جدول 5-5:** نتایج بدست آمده بر روی مجموعه داده Wiki\_CS برگرفته از مقالۀ اصلی

**6) نتایج**

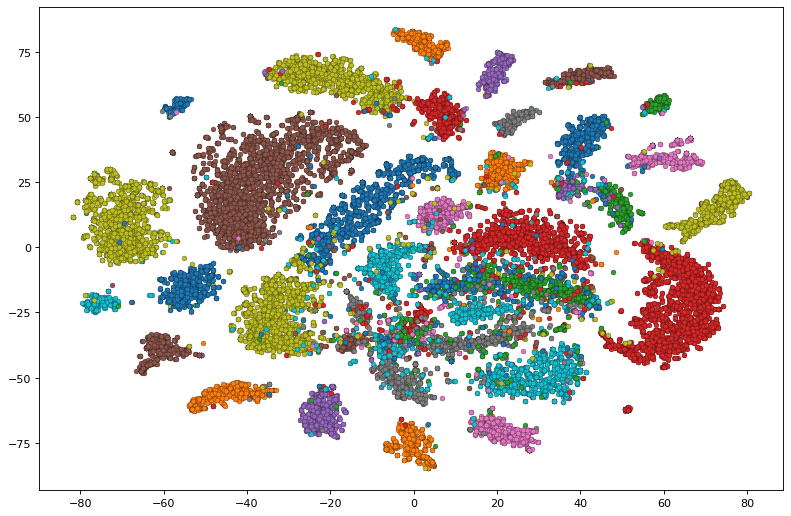
یکی از ویژگی‌های مهم مکانیزم‌های توجه فراهم‌سازی روش‌های مختلف تفسیر مدل است. به کمک روش t-SNE می‌توان بردارهای تعبیه تولید شده برای هر راس را در فضای دو بعدی رسم کرده و به بررسی عملکرد مدل در بازنمایی نزدیک‌تر راس‌های هم‌کلاس بپردازیم. شکل دو بعدی حاصل برای مجموعه داده‌هایی که با رویکرد Transductive آموزش دیده‌اند در شکل 1-6، 2-6 و 3-6 آورده شده است. لازم به ذکر است که رنگ هر راس نمایانگر کلاسی است که بدان تعلق دارد.

****

**شکل 1-6:** نمودار بازنمایی راس‌ها در مجموعه داده Cora به کمک الگوریتم t-SNE

****

**شکل 2-6:** نمودار بازنمایی راس‌ها در مجموعه داده Wiki\_CS به کمک الگوریتم t-SNE

****

**شکل 3-6:** نمودار بازنمایی راس‌ها در مجموعه داده Reddit به کمک الگوریتم t-SNE

* پیاده‌سازی مورداستفاده برگرفته از کدهای [GAT - Graph Attention Network (PyTorch)](https://github.com/gordicaleksa/pytorch-GAT) در گیت‌هاب می‌باشد.

# مراجع

|  |  |
| --- | --- |
| [1] | P. Veličković, "GAT," [Online]. Available: https://github.com/PetarV-/GAT. |
| [2] | P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò and Y. Bengio, "Graph Attention Networks," in *ICLR*, 2018. |
| [3] | M. Zitnik and L. Jure, "Predicting multicellular function through multi-layer tissue networks," in *Proceedings of the 25th International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 2017. |
| [4] | A. T. Subramanian, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gillette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander and J. P. Mesirov, "Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles," in *Proceedings of the National Academy of Sciences*, 2005. |
| [5] | W. L. Hamilton, R. Ying and J. Leskovec, "Inductive representation learning on large graphs," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017. |
| [6] | P Mernyei, C Cangea, "Wiki-CS: A Wikipedia-Based Benchmark for Graph Neural Networks," *arXiv preprint arXiv:2007.02901,* 2020. |

1. Stabilize [↑](#footnote-ref-1)
2. Regularization [↑](#footnote-ref-2)
3. Epochs [↑](#footnote-ref-3)
4. Learning rate [↑](#footnote-ref-4)
5. Patience period [↑](#footnote-ref-5)
6. L2-regularization [↑](#footnote-ref-6)
7. Citation [↑](#footnote-ref-7)
8. Bag-of-words [↑](#footnote-ref-8)
9. Stemming [↑](#footnote-ref-9)
10. Generalization [↑](#footnote-ref-10)
11. Positional gene sets [↑](#footnote-ref-11)
12. Motif gene sets [↑](#footnote-ref-12)
13. Immunological signatures [↑](#footnote-ref-13)
14. Gene ontology sets [↑](#footnote-ref-14)
15. Multi-label classification [↑](#footnote-ref-15)
16. Benchmarking [↑](#footnote-ref-16)
17. Computer Science [↑](#footnote-ref-17)
18. Visible [↑](#footnote-ref-18)
19. Early Stopping [↑](#footnote-ref-19)