به نام خداوند بخشنده مهربان

عنوان پروژه

دسته بندی نودها در گراف دیتاست obgn-product

عنوان درس

داده کاوی

استاد

دکتر محمد کیانی ابری

گردآورنده

سید حسین حسینی دولت‌آبادی

تابستان ۱۴۰۴

دانشکده مهندسی کامپیوتر

دانشگاه اصفهان

فهرست مطالب

۱ .مقدمه

این مستندات به تحلیل جامع پروژه، از مشخصات اجرایی و معماری مدل گرفته تا نتایج و تحلیل دقیق عملکرد می‌پردازد. هدف این پروژه، پیاده‌سازی و ارزیابی دو مدل پایه از شبکه‌های عصبی گراف (GNN) برای وظیفه **طبقه‌بندی گره‌ها** است. گراف مورد استفاده، شبکه محصولات آمازون (ogbn-products) است که در آن گره‌ها نمایانگر محصولات و یال‌ها نشان‌دهنده خرید همزمان دو محصول هستند. هدف نهایی، پیش‌بینی دسته هر محصول (گره) بر اساس ویژگی‌های خود محصول و ساختار ارتباطی آن با محصولات دیگر در گراف است. برای این منظور، دو مدل **GCN** و **GraphSAGE** پیاده‌سازی، آموزش داده شده و نتایج آن‌ها با یکدیگر مقایسه می‌شود.

۱ .مشخصات سیستم و معماری

۱.۱ .مشخصات سیستم اجرایی(Execution System Specifications)

در این بخش، مشخصات سخت‌افزاری و نرم‌افزاری سیستمی که کد بر روی آن اجرا شده است، آورده می‌شود.

- **پردازنده مرکزی (CPU):** 16 * AMD EPYC™ 7003 Series Processors
- **حافظه اصلی (RAM):** 16 * 16 GB per processor
- **پردازنده گرافیکی (GPU):** 1 * NVIDIA H200 Tensor Core GPU
- **حافظه پردازنده گرافیکی (VRAM):** 141GB VRAM
- **سیستم‌عامل (OS):** Ubuntu (Virtual Environment : Conda)
- **کتابخانه‌های اصلی:** PyTorch, PyTorch Geometric, OGB, Scikit-learn, Matplotlib

۱.۲ .خلاصه‌ای از معماری و فرآیند آموزش

هدف این پروژه، طبقه‌بندی محصولات در شبکه خرید آمازون (ogbn-products) است که یک وظیفه **طبقه‌بندی گره (Node Classification)** محسوب می‌شود. در این پروژه، دو معماری کلاسیک شبکه‌های عصبی گراف (GNN) با ساختار مشابه پیاده‌سازی شده‌اند:

۱ .مدل GCN (Graph Convolutional Network) :

- یک شبکه کانولوشن گراف دو لایه که اطلاعات را از همسایگی‌های مستقیم هر گره جمع‌آوری می‌کند.
- **معماری :**

GCNConv(128 ,ورودی) -> ReLU -> GCNConv(128, خروجی)

۲ .مدل GraphSAGE (Graph SAmple and aggreGatE) :

- یک شبکه دو لایه که برای جمع‌آوری اطلاعات از همسایگی‌ها طراحی شده و قابلیت تعمیم‌پذیری بالایی دارد.
- **معماری :**

SAGEConv(128 ,ورودی) -> ReLU -> SAGEConv(128, خروجی)

هر دو مدل با بهینه‌ساز **Adam** و با روش **آموزش تمام-دسته (Full-batch)** برای **۱۰۰ ایپاک** آموزش داده شدند.

۲ .نتایج و تحلیل‌ها

۲.۱ .نتایج نهایی مدل‌ها

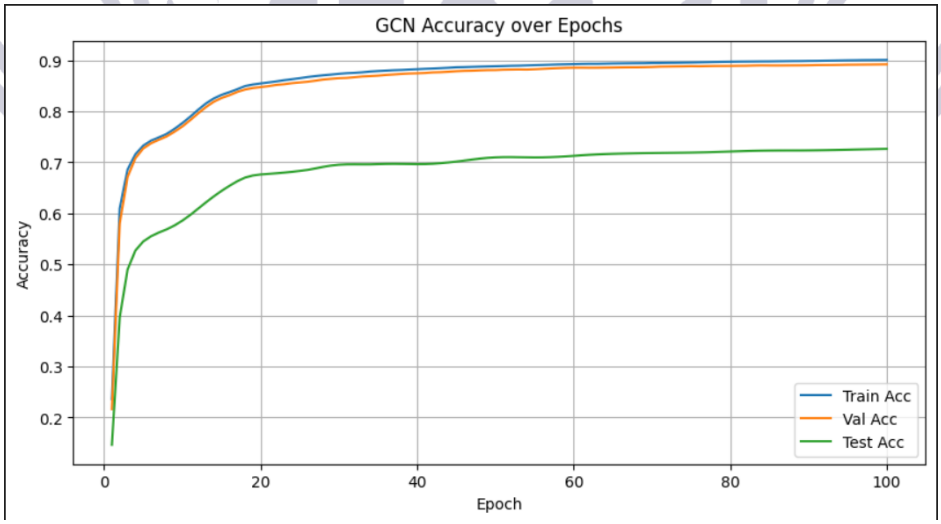
پس از ۱۰۰ ایپاک آموزش، دقت (Accuracy) و امتیاز F1-Score نهایی برای هر دو مدل به شرح زیر است:

مدل (Model)	دقت اعتبارسنجی (Validation Acc)	دقت آزمون (Test Acc)	F1-Score اعتبارسنجی (Val F1)	F1-Score آزمون (Test F1)
GCN	۸۹.۲۲٪	۷۲.۶۶٪	۰.۴۷۷۶	۰.۲۹۶۷
GraphSAGE	**۸۹.۸۹٪**	**۷۳.۸۶٪**	**۰.۴۹۵۵**	**۰.۳۱۸۳**

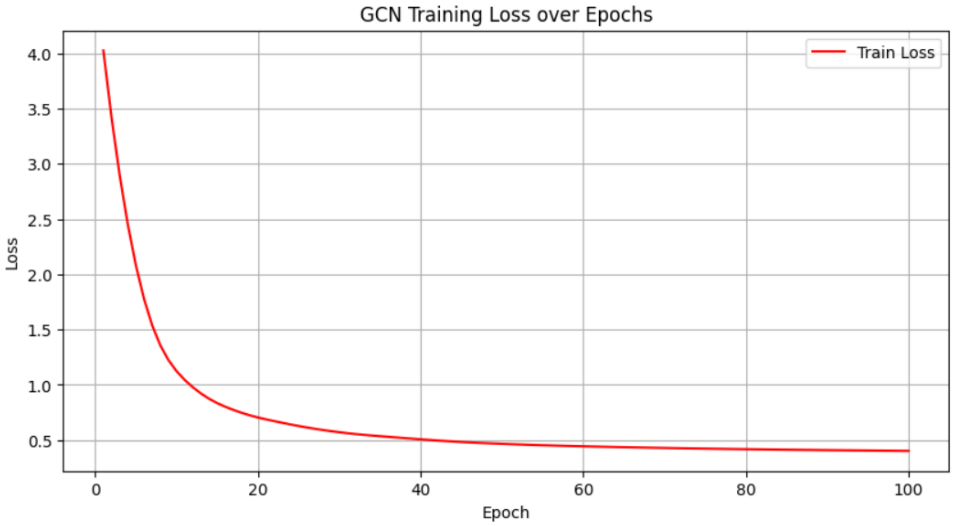
همانطور که مشاهده می‌شود، مدل **GraphSAGE** در تمام معیارها عملکرد بهتری داشته است.

۲.۲ .تحلیل دقیق عملکرد و نمودارها (برای پیشگیری از اشغال فضا یک نمودار در داک است)

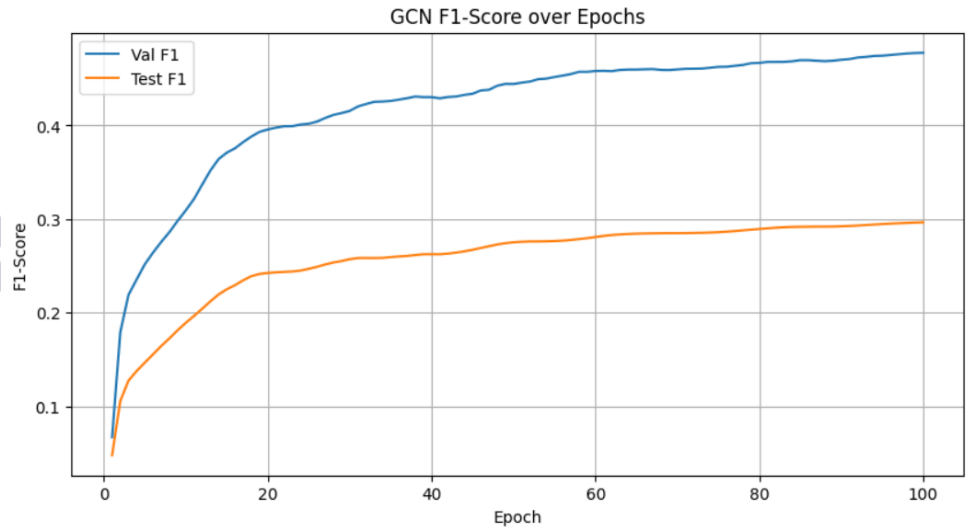
- **نمودار دقت (Accuracy Plot):** هر دو مدل روندی صعودی و همگرا را در نمودار دقت نشان می‌دهند. دقت آموزش به بالای ۹۰٪ می‌رسد، در حالی که دقت اعتبارسنجی و آزمون در سطوح پایین‌تری قرار دارند. این شکاف طبیعی است و نشان می‌دهد که مدل‌ها الگوهای داده‌های آموزش را به خوبی یاد گرفته‌اند. مهم‌تر اینکه دقت اعتبارسنجی نیز به طور پیوسته افزایش یافته و در ایپاک‌های پایانی به پایداری می‌رسد که نشان‌دهنده عدم وقوع بیش‌برازش (Overfitting) شدید است .



- **نمودار هزینه(Loss Plot) :** نمودار هزینه آموزش برای هر دو مدل یک روند نزولی سریع در ایپاک‌های اولیه و سپس کاهش ملایم را نشان می‌دهد. این رفتار کلاسیک، تأییدی بر همگرایی موفقیت‌آمیز مدل‌هاست.



- **نمودار F1-Score :** روند این نمودار نیز مشابه نمودار دقت است و بهبود عملکرد مدل‌ها در طول زمان را تأیید می‌کند. امتیاز F1 به دلیل عدم توازن داده‌ها، معیار مهم‌تری نسبت به دقت است و برتری GraphSAGE در این معیار نیز مشهود است.



۲.۳ .مقایسه عمیق‌تر GCN وGraphSAGE

گرچه هر دو مدل عملکرد بسیار خوبی داشتند، برتری جزئی **GraphSAGE** قابل توجه است. این برتری را می‌توان به تفاوت در نحوه پردازش اطلاعات گره‌ها نسبت داد:

- **GCN:** در هر لایه، بازنمایی جدید یک گره از طریق یک میانگین وزنی نرمالایزشده از ویژگی‌های گره‌های همسایه و خود گره به دست می‌آید. این یک عملیات مبتنی بر رویکرد طیفی (Spectral) است که به طور همزمان اطلاعات همسایه‌ها و خود گره را ترکیب می‌کند.
- **GraphSAGE:** این مدل فرآیند را به دو مرحله مجزا تقسیم می‌کند:

1. **جمع‌آوری(Aggregate) :** ابتدا ویژگی‌های گره‌های همسایه با یک تابع جمع‌کننده (مانند میانگین) ترکیب می‌شوند.
2. **به‌روزرسانی(Update) :** سپس، بازنمایی به‌دست‌آمده از همسایه‌ها با بازنمایی خود گره ترکیب می‌شود تا بازنمایی نهایی گره در آن لایه ایجاد شود.

این جداسازی فرآیند، به مدل انعطاف‌پذیری بیشتری می‌دهد تا روابط پیچیده‌تری بین یک گره و همسایگانش را یاد بگیرد، که احتمالاً منجر به عملکرد بهتر در این وظیفه شده است.

۳ .جزئیات پیاده‌سازی

۳.۱ .آماده‌سازی محیط و داده‌ها

اولین گام در پروژه، نصب کتابخانه‌های ضروری و بارگذاری داده‌هاست. کتابخانه‌های کلیدی شامل PyTorch به عنوان فریمورک اصلی یادگیری عمیق، PyTorch Geometric برای پیاده‌سازی مدل‌های گراف و OGB برای بارگذاری آسان مجموعه داده‌های استاندارد گراف است.

مجموعه داده ogbn-products با استفاده از کلاس NodePropPredDataset از کتابخانه OGB بارگذاری می‌شود. این مجموعه داده شامل موارد زیر است:

- **گراف:** ساختار ارتباطی بین محصولات که شامل بیش از ۲.۴ میلیون گره (محصول) و ۱۲۳ میلیون یال است.
- **ویژگی‌های گره(Node Features) :** هر محصول با یک بردار ویژگی ۱۰۰ بعدی توصیف می‌شود.
- **برچسب‌ها(Labels) :** دسته مربوط به هر محصول که مدل باید آن را پیش‌بینی کند.
- **ایندکس‌های تقسیم داده:** داده‌ها به سه مجموعه **آموزش(Train)** ، **اعتبارسنجی (Validation)** و **آزمون (Test)** تقسیم شده‌اند.

این اطلاعات در یک شیء Data از کتابخانه torch_geometric ذخیره می‌شوند که یک ساختار استاندارد برای مدیریت داده‌های گراف است.

۳.۲ .معماری مدل‌ها: جزئیات فنی

برای این پروژه، دو معماری GNN با ساختار دو لایه پیاده‌سازی شده‌اند تا قابلیت یادگیری الگوهای پیچیده‌تر از همسایگی‌های یک و دو مرحله‌ای را داشته باشند.

- **GCN (Graph Convolutional Network):**

  GCN یکی از اولین و پایه‌ای‌ترین معماری‌های GNN است که با تعمیم عملیات کانولوشن بر روی داده‌های گراف کار می‌کند. در هر لایه، بازنمایی یک گره با میانگین‌گیری از ویژگی‌های گره‌های همسایه (و خود گره) به‌روزرسانی می‌شود.

- **GraphSAGE (Graph SAmple and aggreGatE):**

GraphSAGE یک معماری پیشرفته‌تر است که با هدف مقیاس‌پذیری برای گراف‌های بزرگ طراحی شده است. ایده اصلی آن "نمونه‌برداری و جمع‌آوری (SAmple and aggreGatE) "از همسایگی‌هاست. این مدل به جای استفاده از کل همسایگی، می‌تواند از زیرمجموعه‌ای تصادفی از همسایه‌ها برای جمع‌آوری اطلاعات استفاده کند.

۳.۳ .فرآیند آموزش و ارزیابی

- **تابع آموزش(train_model) :** این تابع حلقه اصلی آموزش را مدیریت می‌کند. در هر ایپاک، مدل در حالت آموزش قرار گرفته، خروجی برای کل گراف محاسبه می‌شود، و تابع هزینه **Cross-Entropy Loss** تنها بر روی گره‌های مجموعه آموزش محاسبه می‌گردد. سپس با استفاده از الگوریتم **Adam** به عنوان بهینه‌ساز، وزن‌های مدل به‌روزرسانی می‌شوند.
- **تابع ارزیابی(evaluate_model) :** پس از هر ایپاک آموزش، این تابع برای سنجش عملکرد مدل بر روی هر سه مجموعه داده فراخوانی می‌شود. معیارهای ارزیابی شامل **دقت (Accuracy)** و **امتیاز F1-Score (با میانگین 'macro')** است.

۴ .نتیجه‌گیری و پیشنهادات

۴.۱ .جمع‌بندی نهایی

این پروژه با موفقیت نشان داد که معماری‌های GNN مانند GCN و GraphSAGE قادرند به طور مؤثری وظیفه طبقه‌بندی گره در گراف‌های بزرگ را انجام دهند. هر دو مدل توانستند به دقت‌های خوبی دست یابند، اما مدل GraphSAGE به دلیل معماری انعطاف‌پذیرتر خود، نتایج اندکی بهتری ارائه داد. این پروژه به عنوان یک نمونه کاربردی، قدرت شبکه‌های عصبی گراف در تحلیل داده‌های ساختاریافته و رابطه‌ای را به خوبی به تصویر می‌کشد.

۴.۲ .پیشنهادات برای کارهای آینده

برای بهبود نتایج و ادامه کار، می‌توان موارد زیر را پیشنهاد داد:

- **بهینه‌سازی هایپرپارامترها :**تنظیم دقیق پارامترهایی مانند نرخ یادگیری، ابعاد لایه پنهان، و weight decay می‌تواند به بهبود عملکرد منجر شود.
- **استفاده از مدل‌های پیشرفته‌تر :**معماری‌هایی مانند **GAT (Graph Attention Network)** که به یال‌ها وزن‌های متفاوتی اختصاص می‌دهند، می‌توانند نتایج را بهبود بخشند.

- **آموزش مبتنی بر نمونه‌برداری(Sampling-based Training) :** برای گراف‌های بزرگ‌تر، استفاده از روش‌های آموزش مینی-بچ مانند **Neighbor Sampling**(که GraphSAGE برای آن طراحی شده)ضروری است تا مشکل حافظه برطرف شود.

- **افزودن تکنیک‌های تنظیم‌گری(Regularization) :** استفاده از Dropout در لایه‌های GNN می‌تواند از بیش‌برازش جلوگیری کرده و به تعمیم‌پذیری بهتر مدل کمک کند.

۵ .منابع برای مطالعه بیشتر

برای درک عمیق‌تر مفاهیم و تکنیک‌های استفاده‌شده در این پروژه، مطالعه منابع زیر پیشنهاد می‌شود:

۵.۱ .کتاب:Graph Representation Learning

- **نویسنده:**William L. Hamilton
- **توضیحات:** این کتاب یکی از جامع‌ترین و پایه‌ای‌ترین منابع برای یادگیری نحوه بازنمایی داده‌های گراف است. این کتاب به طور کامل مفاهیم اساسی مانند تعبیه گره(Node Embedding) ، شبکه‌های عصبی گراف (GNNs) و کاربردهای آن‌ها را پوشش می‌دهد. فصل‌های مربوط به **GCN** و **GraphSAGE** در این کتاب، درک بسیار عمیقی از مبانی نظری این مدل‌ها ارائه می‌دهند.

۵.۲ .دوره آنلاین: CS224W - Machine Learning with Graphs (دانشگاه استنفورد)

- **مدرس:**Jure Leskovec
- **توضیحات :**این دوره یکی از معتبرترین و شناخته‌شده‌ترین دوره‌های آموزشی در زمینه یادگیری ماشین بر روی گراف‌هاست. این دوره تمامی مباحث از الگوریتم‌های کلاسیک تحلیل گراف تا پیشرفته‌ترین مدل‌های یادگیری عمیق مانندGCN ، GraphSAGE و GAT را پوشش می‌دهد. اسلایدها و ویدئوهای این دوره به صورت رایگان در دسترس عموم قرار دارند و منبعی عالی برای یادگیری عملی و نظری هستند.

۵.۳ .مقاله اصلی GraphSAGE

- **عنوان:** Inductive Representation Learning on Large Graphs
- **نویسندگان:** William L. Hamilton, Rex Ying, Jure Leskovec
- **توضیحات:** این مقاله، که مدل GraphSAGE را معرفی می‌کند، یکی از تأثیرگذارترین مقالات در حوزه یادگیری ماشین بر روی گراف‌هاست. مطالعه این مقاله نه تنها به درک عمیق‌تر معماری GraphSAGE کمک می‌کند، بلکه بینش بسیار خوبی در مورد چالش‌های کار با گراف‌های بزرگ و روش‌های **یادگیری استقرایی (Inductive Learning)** ارائه می‌دهد. این مقاله تفاوت کلیدی بین مدل‌های استقرایی (مانند GraphSAGE) و مدل‌های ترادکتیو (مانند GCN در پیاده‌سازی‌های اولیه) را به خوبی روشن می‌کند.

۶. لینک فیلم توضیحات تکمیلی پروژه

https://drive.google.com/file/d/1uZj7LpgXsA4oJsRIlqVGpeaLTfbtbM1t/view?usp=sharing

موفق باشید 😊😅