



ب ررسی شبکه‌ی گروه تلگرام درس گزارش پروژه‌ی درس علم شبکه

گروه Mr. Reza علیرضا حبیب زاده، یاسمن کاتبی، علی فیاض - پاییز ۱۴۰۲ دکتر مقیمی

لینک پروژه در گیت‌هاب:

<https://github.com/alirezahabib/sut-telegram-network>

مقدمه

در این پروژه شبکه‌ی ارتباطات اجتماعی اعضای درس را با استفاده از شیوه‌ی غالب ارتباطات مجازی افراد یعنی تلگرام بررسی کردیم. البته طبعاً دسترسی به پیام‌های شخصی افراد ممکن نیست و گروه تلگرام درس برای این کار بررسی شده. به نظر ما نماینده‌ی خوبی از ارتباطات درسی اعضای درس است.

جمع‌آوری داده

برای جمع‌آوری باید از API تلگرام استفاده کنیم. یک راه ممکن برای این کار استفاده از API بات تلگرام است. به صورتی که یک بات در گروه مورد نظر اضافه شود و سپس پیام‌ها و اعضای گروه را دریافت کند. اما اضافه کردن بات به گروه جنبه‌ی خوبی ندارد و کار را سخت می‌کند. سوال ما این بود که آیا می‌شود صرفاً همان طوری که یک کلاینت تلگرام پیام‌ها را دریافت می‌کند ما نیز بتوانیم مانند یک کلاینت معمولی و با برای مثال لایکین کردن به اکانت یکی از اعضای گروه پیام‌های گروه را جمع کنیم؟

کد کلاینت شخصی تلگرام

پاسخ: بله! برای این کار کافی است یک کد کلاینت شخصی از تلگرام دریافت کنید. کد معنی این کار این است که من می‌خواهم یک کلاینت برای خودم توسعه دهم. علاوه بر این که کد کلایناتهای رسمی تلگرام منبع باز (open source) هستند، کد API تلگرام (MTProto) نیز منبع باز است و افراد می‌توانند برای تلگرام کلاینت خود را توسعه دهند. اما برای جلوگیری از سرورهای تلگرام و... کلایناتهای غیررسمی باید شناسه‌ی خود را (با اکانت یک فرد که عضو تلگرام است) دریافت کنند.

نکته‌ی جالب این که در جست و جویی که برای کدهای مشابه در گیت‌هاب داشتم یک نفر کد کلاینتش را به اشتباه در ریپابلیت‌وری خود قرار داده بود و من دیدم که کار می‌کند لذا دیگر کدی نگرفتم. (تا وقتی از سرورهای تلگرام سواستفاده نمی‌کنید و حجم زیادی داده بیرون نمی‌کشید این کار هیچ اشکالی ندارد، نهایتاً کد کلاینت آن شخص منقضی یا کند خواهد شد).

Telethon

تلثون یک کلاینت غیررسمی اوپن‌سورس مبتنی بر پایتون برای تلگرام است. این کلاینت به صورت یک کتابخانه‌ی پایتون نوشته شده و با استفاده از آن و یک کلید کلاینت شخصی تلگرام می‌توان به راحتی تقریباً هر کاری که یک کلاینت گرافیکی تلگرام انجام می‌دهد انجام داد.

لینک صفحه‌ی گیت‌هاب تلثون:

<https://docs.telethon.dev/en/stable/>

لینک داکیومنتیشن تلثون:

<https://docs.telethon.dev/en/stable/>

اعضای گروه

با یک کال ساده اعضای گروه را دریافت می‌کنیم. نهایت مواردی که از اطلاعات افراد برای ما حائز اهمیت است این‌ها هستند:

- user_id
- username
- access_hash
- first_name
- last_name
- group_id
- group

پیام‌های گروه

دریافت پیام‌های گروه به صورت یکجا امکان‌پذیر نیست چرا که تعداد و حجم آن‌ها زیاد است. با استفاده از offset در هنگام دریافت پیام‌ها می‌توان پیام‌های قدیمی‌تر را دریافت کرد و هر بار حدود ۵۰ پیام دریافت می‌کنیم. نهایتاً نتیجه را در یک فایل CSV ذخیره کرده‌ایم. فیلدهایی که نگه داشتیم این‌ها هستند:

- message_id
- from_user_id
- reply_to
- pinned
- message
- reactions

ادغام گروه جدید و قدیمی (!?)

نکته‌ی جالبی که در حین ذخیره اطلاعات گروه مشاهده کردیم این بود که تلگرام پس از این که تعداد اعضای گروه از حدی بیشتر می‌شود گروه را به سوپرگروه تغییر می‌دهد. اما این تغییر گروه بدون خونریزی نیست چرا که تا ابد نزد تلگرام گروه قدیمی و پیام‌های آن به صورت یک گروه جداگانه باقی خواهد ماند. البته خوشبختانه تلگرام فیلد شناسه‌ی پیام‌ها (message_id) را برای پیام‌های قدیمی تغییر می‌دهد تا از عدد بزرگی مثل حدود 60000 شروع کنند. در این صورت بین شناسه‌ی پیام‌های گروه جدید و گروه قدیمی تداخل ایجاد نمی‌شود و به راحتی می‌توان بین آن‌ها مثلاً ریپلی زد.

با این تفاسیر تقریباً تنها کاری که برای ادغام این دو قسمت گروه درس لازم است، زیر هم قرار دادن فایل‌های CSV ایجاد شده است که این کار را انجام دادیم.

جمع‌آوری عکس‌های پروفایل

عکس‌های پروفایل افراد گروه که به صورت پابلیک بودند جمع‌آوری شده و در یک پوشش قرار گرفتند. نام هر فایل را شناسه‌ی عددی حساب آن فرد قرار دادیم. از آنجایی که افراد این عکس‌ها را به صورت عمومی قرار داده بودند از نظر ما استفاده از آن‌ها در ارائه و گزارشی که خود اعضای گروه آن را خواهند دید اشکالی ندارد.

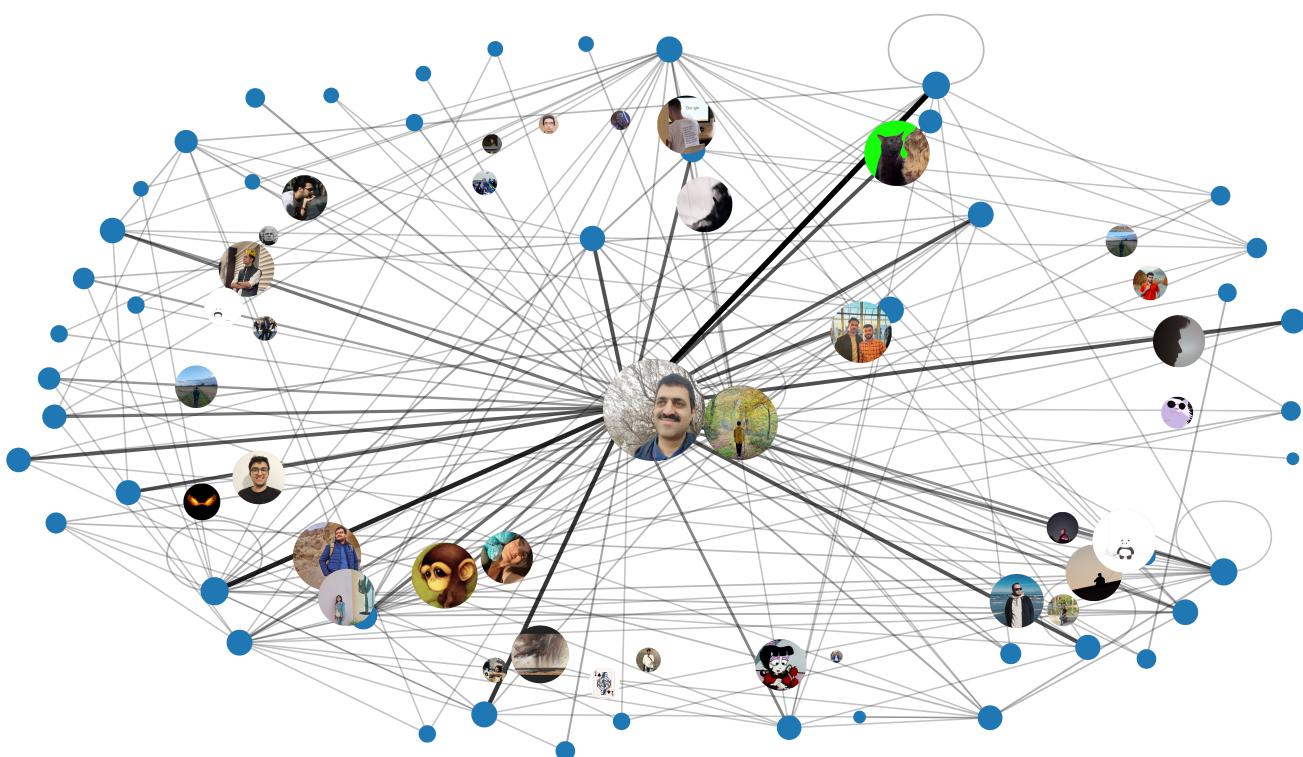
تحلیل داده‌ها

حال که داده‌های گروه را جمع‌آوری کردیم لازم است تا آن را بررسی کنیم. برای تشکیل شبکه‌ی ارتباطات می‌توان مسائل مختلفی را در نظر گرفت. در حالت کلی این شبکه‌ی جهتدار است چرا که ارتباطات افراد جهت دارد:

1. هر پیام A می‌تواند به پیام دیگر B ریپلی شده باشد. در این صورت می‌توان یک یال از فرستنده‌ی A به فرستنده‌ی B اضافه کرد. افراد می‌توانند به پیام خودشان ریپلی بزنند.
2. هر ریکشنی که زده می‌شود یک یال از فرستنده‌ی ریکشن به فرستنده‌ی پیام اضافه می‌کند.
3. هر پیامی که پین می‌شود یک یال از فرستنده‌ی پیام به همه افراد گروه اضافه می‌کند. یادداشت: در پیاده‌سازی نهایی علی‌رغم پیاده‌سازی فنی این بخش و ذخیره‌ی فیلد `is_pinned` در دیتابیس یال‌های این قسمت را اضافه نکردم چرا که تنها استاد درس پیام پین می‌کند و این یک مزیت اضافه برای استاد ایجاد می‌کند و استاد همین‌لان هم حتی از حالت توانی کمی از نظر فعالیت بیرون زده است برای همین این کار نتیجه را خراب می‌کند. ضمناً تعداد زیادی از افراد هیچ فعالیتی در گروه نداشتند و از گراف ارتباطات نهایی به کلی ظاهر ننمی‌شدند، این کار آن‌ها را با تنها چند یال به شبکه می‌آورد و شبکه را الکی شلوغ می‌کند با حذف این کار آن‌ها به صورت خودکار از شبکه حذف شدند (احتمالاً درس را نداشتند یا اشتباهی عضو شدند یا حذف کردند یا ...)

نتایج

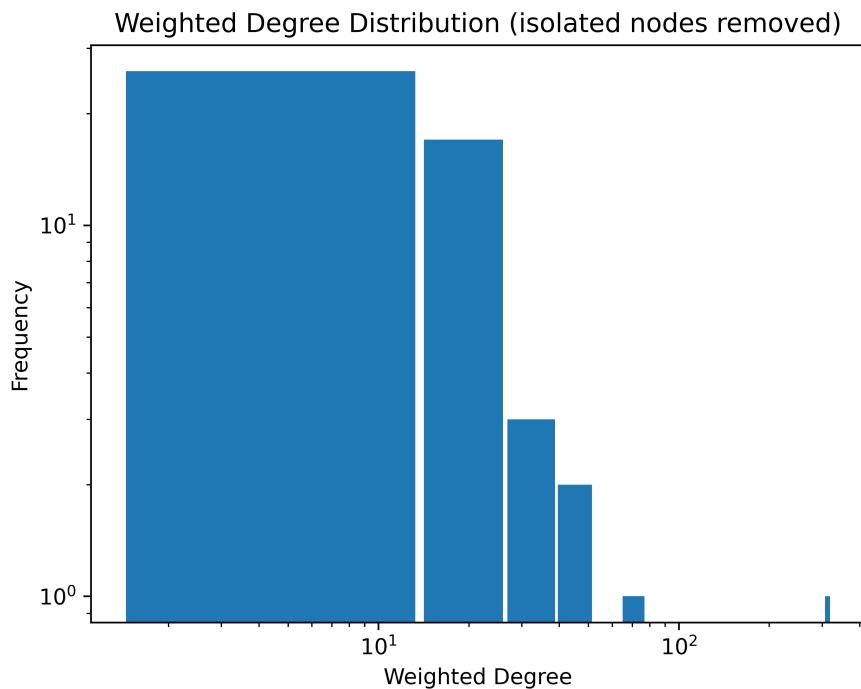
رسم شبکه



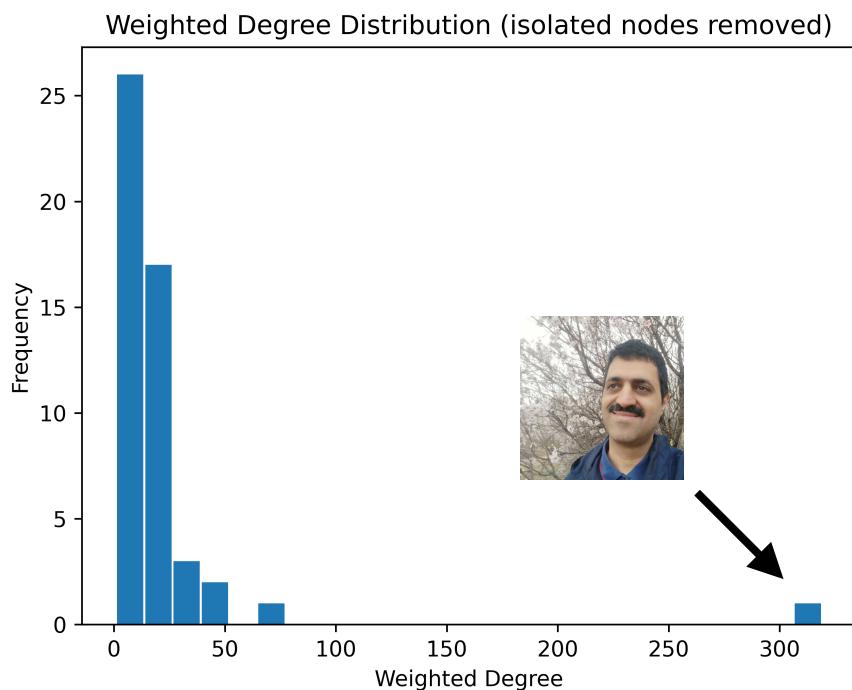
تصویر رسم شده از شبکه. برای رسم از مدل فنری (spring_layout) با پارامترهای `k=3`, `iterations=50` استفاده شده است. همچنین یال‌ها با پارامترهای شفافیت و ضخامت `width=(1 + 3 * edge_weights)`, `alpha=(0.2 + edge_weights * 0.8)` رسم شده‌اند. همچنین اندازه‌ی راس‌ها یک معیار لگاریتمی از درجه‌ی آن‌ها است.
`np.log(degree + 1) * 130`

رفتار توانی

به دو شکل درجه‌ها را بررسی کردیم. یکی درجه‌ی وزن‌دار یعنی جمع تعداد یال‌هایی که هر کس با بقیه دارد. و در حالت دیگر تنها ارتباط داشتن یا نداشتن هر فرد با فرد دیگر را لاحظ کردیم. در هر دو حالت درجه‌ی ورودی و خروجی را جمع زده‌ایم. نتیجه این که در هر دو صورت رفتار توانی با همان توانی که تقریباً انتظار داریم در شبکه مشهود است.

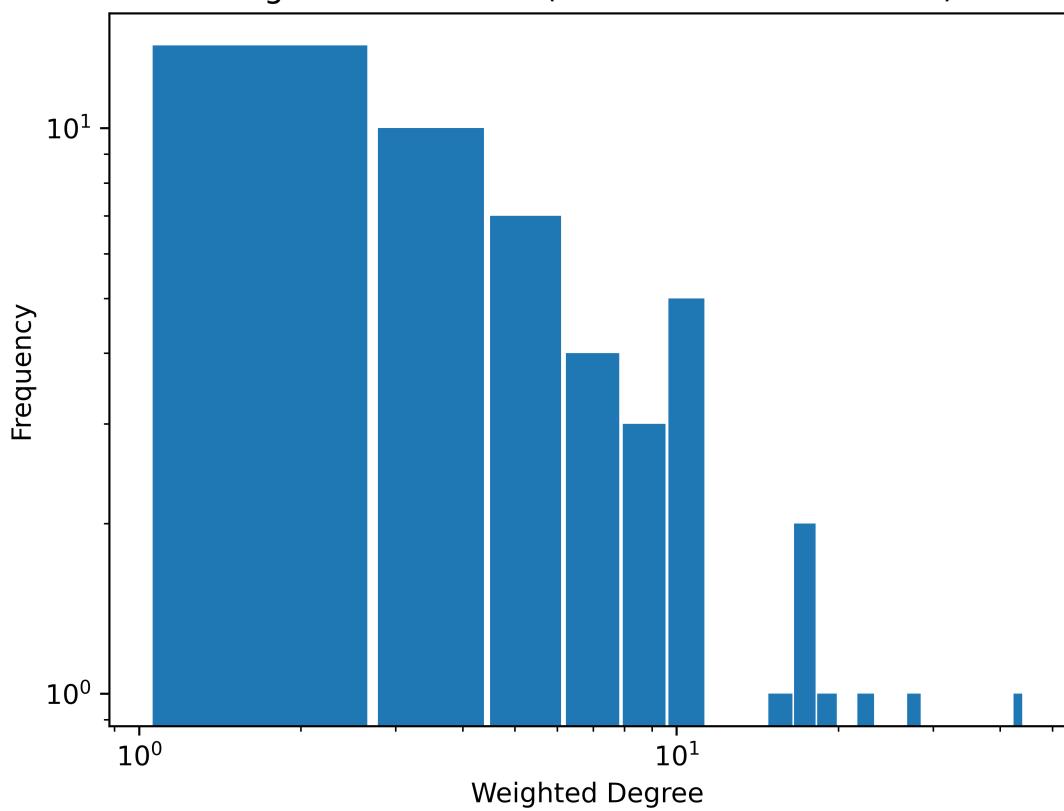


توزیع درجه‌ی وزن‌دار در نمودار لگاریتمی با کیسه‌بندی

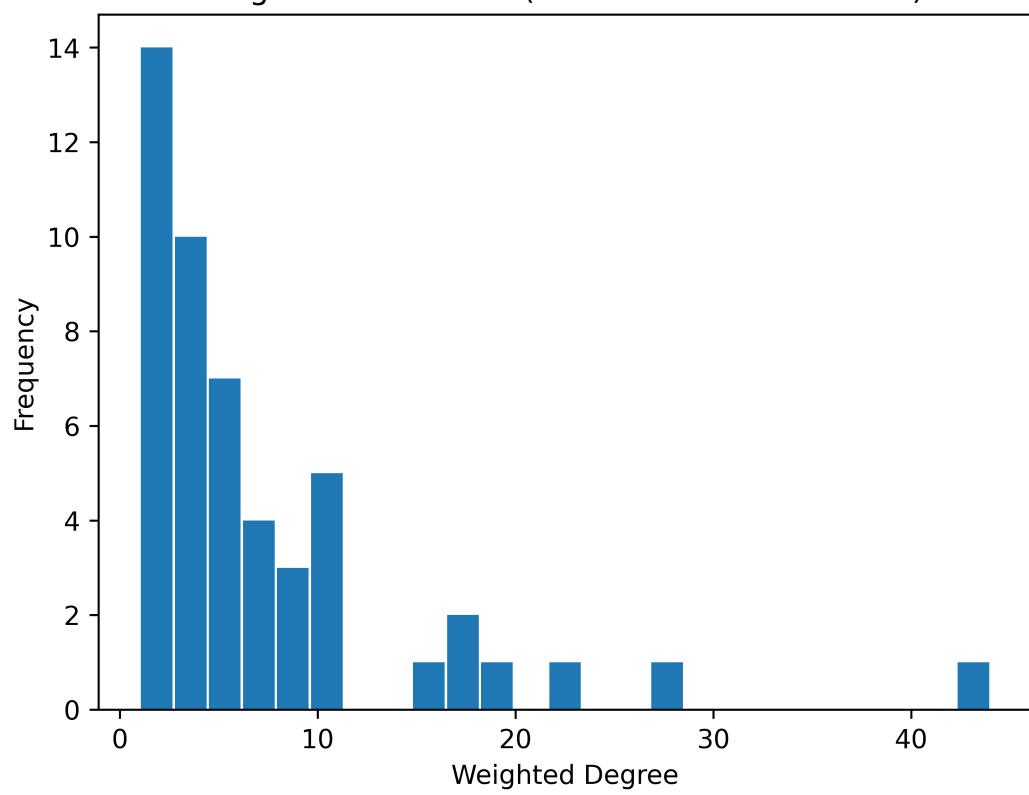


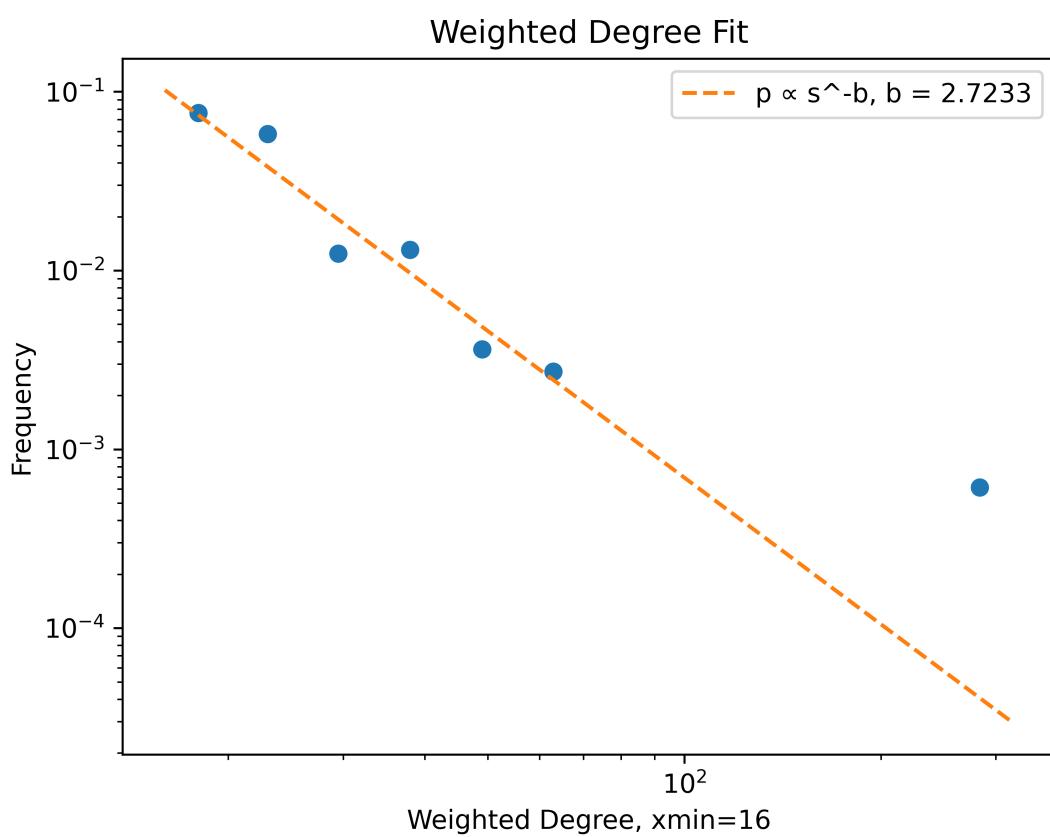
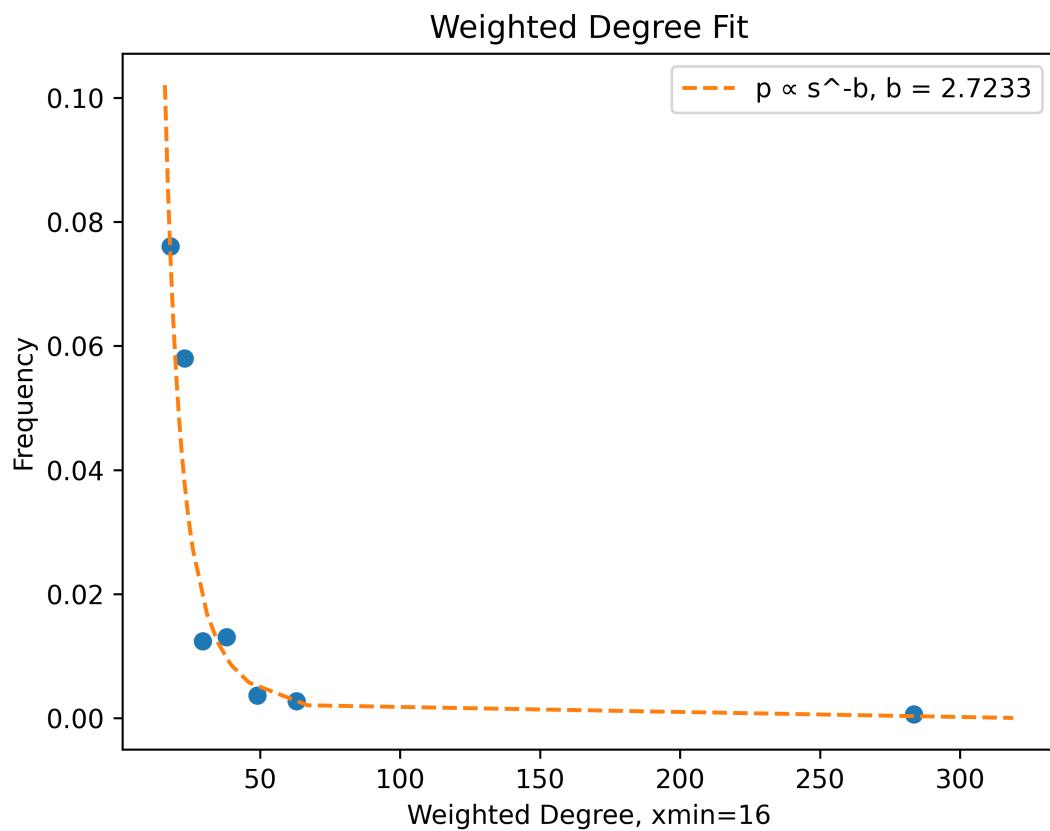
توزیع درجه‌ی وزن‌دار در نمودار خطی با کیسه‌بندی. بخش خوبی از ارتباطات زیاد دکتر مقدمی در شبکه ریکشن‌های زیادی است که پیام‌هایشان می‌خورد. البته در کل هم ارتباطات زیاد است.

Degree Distribution (isolated nodes removed)

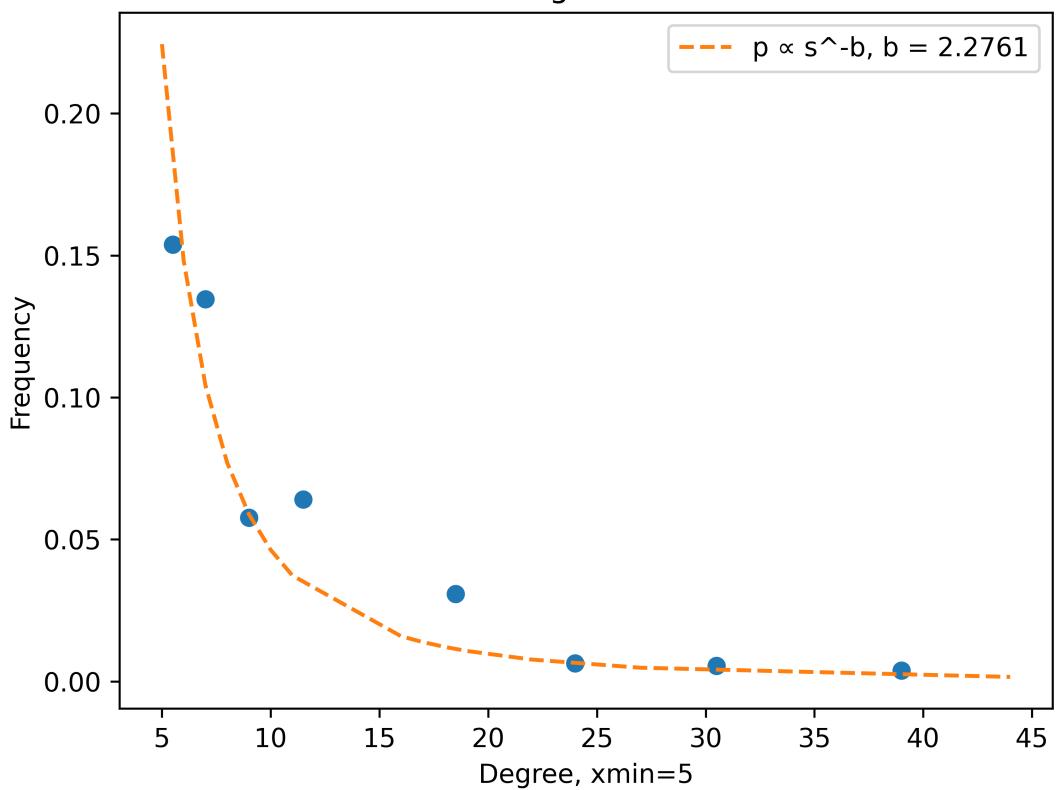


Degree Distribution (isolated nodes removed)

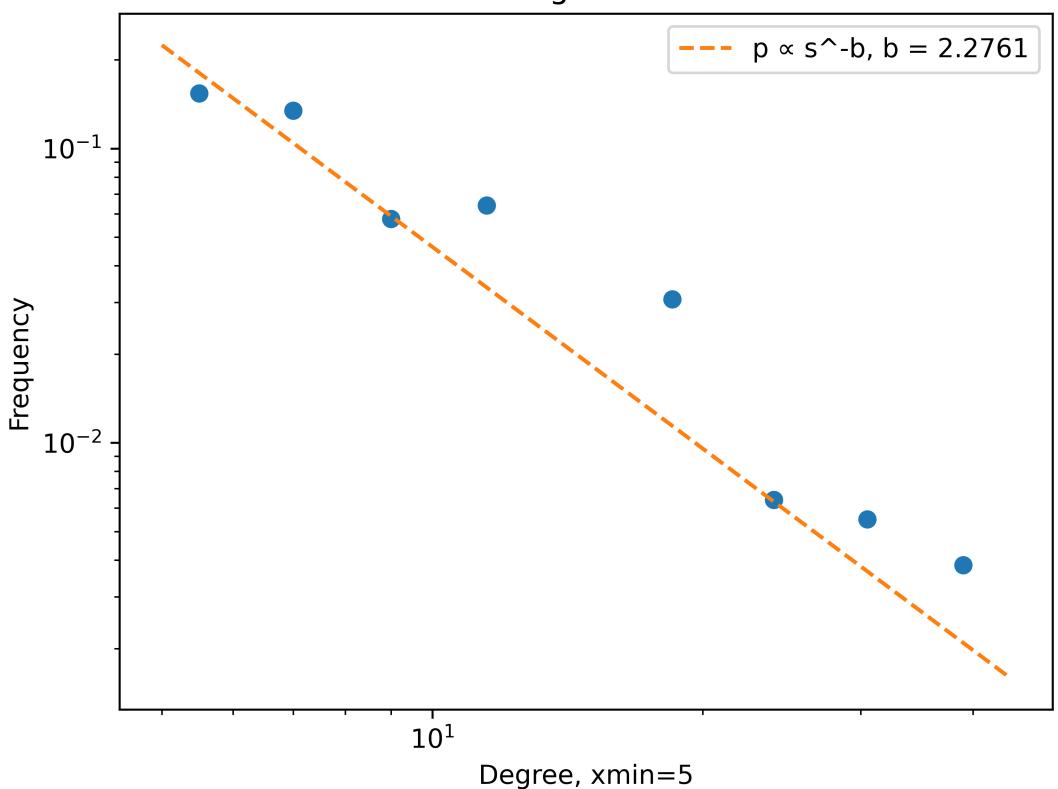




Degree Fit



Degree Fit



برای آینده

برای آینده این موارد را برای پروژه در نظر داریم:

1. در تمام تحلیلها عکس‌ها و نمودارها به صورت فایل png خروجی گرفته شدند. بهتر بود تا جای ممکن همه‌چیز svg باشد تا حجم آن‌ها زیاد نشود و ضمناً کیفیت آن‌ها با زوم کاهش نیابد.
2. می‌توان گروههای بزرگتری را بررسی کرد. ضمناً برای بررسی واقع‌گرایانه‌تری از ارتباطات بین افراد باید بتوان اطلاعات چند گروه را با هم ادغام کرد.
3. بررسی منشن‌ها: برای هر پیامی که در آن یک فرد منشن شده (مثلاً @habibz در یک پیام) می‌توان یک یال از فرستنده‌ی پیام به فرد (افراد) منشن‌شونده اضافه کرد. در پیاده‌سازی اولیه بیخیال این مسئله شدیم چرا که تعداد منشن‌های گروه بسیار کم بود و ضمناً پیاده‌سازی آن بسیار دشوار است. فیلد mentioned تلگرام تنها می‌گوید شما (صاحب اکانت خزنده) در این پیام منشن شده‌اید یا خیر و کاری به منشن شدن سایرین ندارد. برای پیاده‌سازی باید یوزرنیم همه‌ی افراد را در همه‌ی پیام‌ها جست و جو کرد و ضمناً افرادی که یوزرنیم ندارند هم قایل منشن شدن هستند و نمی‌دانم در این صورت در پیام چه اتفاقی می‌افتد.
4. بررسی جداگانه‌ی یالهای ورودی و خروجی: اینجا اطلاعات یالهای ورودی و خروجی را دور ریختیم و آن‌ها را با هم جمع کردیم. شاید بررسی جداگانه‌ی آن‌ها بتواند مفید باشد.
5. ذخیره شبکه: تنها پیامها و افراد را ذخیره می‌کنیم. بهتر است اطلاعات شبکه‌ی حاصل را هم ذخیره کنیم تا بتوان بعداً رو آن تحلیل‌های بیشتری انجام داد.

مواد لازم

این پروژه از کتابخانه‌های پایتون زیر استفاده می‌کند که ضمن تشرک از سازنده‌ی آن‌ها در کنار هر کدام توضیح کارکرد آن را آورده‌ایم.

نام کتابخانه	دلیل استفاده
telethon	برای ارتباط با تلگرام
matplotlib	برای رسم نمودارهای بخش تحلیل
networkx	برای تحلیل و رسم شبکه
pandas	برای ذخیره و لود جدول CSV پیامها و اعضای گروه
pillow	برای ذخیره و لود عکس‌های پروفایل
numpy	برای تحلیل شبکه
powerlaw	برای تحلیل شبکه