



دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

پایان نامه کارشناسی ارشد
گرایش مهندسی نرم افزار

عنوان

استنتاج ساختار شبکه از داده های ناکامل

نگارش
پیام سیری

استاد راهنما
دکتر حمیدرضا ربیعی

تابستان ۱۳۹۲

دانشگاه صنعتی شریف
دانشکده کامپیوتر

رساله کارشناسی ارشد

استنتاج ساختار شبکه از داده‌های ناکامل

نگارش: پیام سیری

امضاء: استاد راهنما: دکتر حمیدرضا ربیعی

امضاء: استاد ممتحن داخلی: دکتر محمد قدسی

امضاء: استاد ممتحن خارجی: دکتر مسعود اسدپور

تقدیم به خانواده عزیزم،
که همیشه مشوق و پشتیبان من بوده‌اند.

سپاسگزاری و قدردانی

از خداوند یکتا سپاسگزارم که توانستم مرحله‌ی دیگری از تحصیلاتم را پشت سر بگذارم. این رساله نه تنها حاصل کار من، بلکه نتیجه تلاش‌های همه دوستان من است که در آن نقشی موثر داشته‌اند. در ابتدا لازم است از زحمات جناب آقای دکتر ریبعی که همواره در طول این پژوهش راهنمای این جانب بوده‌اند، کمال سپاسگزاری را داشته باشم. همچنین از اساتید داور، آقای دکتر قدسی و آقای دکتر اسدپور که با دقت نظر به نقد و بررسی این رساله پرداختند، کمال تشکر را دارم. از آقای دکتر مصطفی صالحی، خانم مطهره اسلامی و آقای حمیدرضا ماهیار که در طول این دوره همراه و راهنمای من بودند، صمیمانه قدردانی می‌کنم و امیدوارم در زندگی و تحصیل همواره موفق باشند.

در انتها سپاسگزارم از خانواده و به ویژه برادر عزیزم، دکتر مسعود سیری که در زندگی همواره یار و یاور من بوده‌اند.

پیام سیری
شهریور ۱۳۹۲

چکیده

در طی یک دهه اخیر مطالعه بر روی جنبه‌های مختلف شبکه‌های پیچیده توسط بسیاری از محققین مورد توجه قرار گرفته است. جمع آوری داده اولین گام در تحلیل این شبکه‌ها محسوب می‌گردد اما با توجه به مقیاس بزرگ این شبکه‌ها، گردآوری اطلاعات کامل از یک شبکه به دلیل سربار زیاد فرآیند جمع آوری، امکان‌پذیر نخواهد بود. لذا تحلیل شبکه برمبنای داده‌های ناکامل حاصل از نمونه‌برداری شبکه انجام خواهد شد. گرچه نمونه‌برداری از شبکه چالش‌های خاص خود را دارد، داده ناکامل باعث می‌شود بسیاری از تحلیل‌ها در مورد شبکه مانند تخمين معیارهای اندازه‌گیری مختلف، دقیق نباشد. بنابراین یکی از چالش‌های اساسی پس از نمونه‌برداری، این است که آیا می‌توان بخش از دست رفته شبکه را پیش‌بینی کرد و ساختار شبکه را با داده‌های حاصل از نمونه‌برداری استنتاج نمود. رویکرد اصلی برای استنتاج ساختار شبکه، رویکرد مبتنی بر مدل است که در آن برای شبکه تحت بررسی، مدلی فرض شده و سعی می‌شود پارامترهای این مدل تخمين زده شود. بسته به داده‌ای که از آن برای استنتاج ساختار استفاده می‌شود، روش‌های استنتاج ساختار ممکن است متفاوت باشند. تکیه روش‌هایی که در این پایان‌نامه ارائه شده‌اند، بر روی ساختار تنک شبکه‌های واقعی است. الگوی ساختار تنک تقریباً در تمامی شبکه‌های واقعی و در هر زمینه مطالعاتی مشاهده می‌شود.

به عنوان مسئله نخست، در این پایان‌نامه به بررسی مسئله بازسازی شبکه زیرین از روی داده‌های انتشار اطلاعات بر روی شبکه می‌پردازم. در مسئله بازسازی شبکه، رئوس گراف و داده انتشار در دسترس بوده و هدف، پیش‌بینی یالهای گراف اولیه است. برای حل این مسئله، روشهای بر اساس چارچوب نمونه‌برداری فشرده ارائه شده است. نمونه‌برداری فشرده یکی از روش‌های جدید در حوزه پردازش و بازسازی سیگنال‌های تنک می‌باشد. نتایج عملی بدست آمده بر روی داده‌های مصنوعی و واقعی، نشان‌دهنده ده درصد بهبود دقت در روش ارائه شده نسبت به روش‌های پیشین حتی در حضور داده‌های بسیار کم است. به علاوه، روش ارائه شده دارای جامعیت بوده و می‌تواند تحت داده‌هایی که خواص فرایند انتشار اطلاعات را داشته باشند، مانند تشخیص لینک‌های تأخیردار در شبکه‌های کامپیوتری، نیز به کار گرفته شود.

مسئله پیش‌بینی لینک به عنوان مسئله دوم در این پایان‌نامه مطرح شده است. این مسئله از پایه‌ای ترین مسائل در حوزه تحلیل شبکه‌ها است و کاربردهای زیادی در سیستم‌های پیشنهاددهنده و شبکه‌های زیستی دارد. ورودی در این مسئله یک گراف ایستا بوده که در آن رئوس و بخشی از یالهای بین آن‌ها مشخص هستند و هدف، پیش‌بینی وجود یا عدم وجود لینک در بین گره‌های شبکه با استفاده از مجاورت مشاهده شده میان رئوس است. برای این مسئله، روشهای سریع و مقیاس‌پذیر تحت چارچوب فاکتورگیری ماتریس نیز، مانند نمونه‌برداری فشرده، از روش‌های جدید و بسیار پرکاربرد در پردازش داده‌های تنک می‌باشد. به طور دقیق‌تر، در روش ارائه شده یک رهیافت احتمالاتی با درنظر گرفتن فرایند پواسون برای مدل‌سازی شبکه مورد نظر به کار گرفته شده و نشان داده می‌شود که این رهیافت منجر به تبدیل مسئله پیش‌بینی لینک به نوعی از مسئله فاکتورگیری ماتریس می‌شود. همچنین نشان داده می‌شود که این مدل‌سازی منجر به استنتاج سریع پارامترها می‌شود. آزمایش‌های عملی بر روی داده‌های واقعی و بزرگ نشان‌دهنده عملکرد تا پانزده برابر سریع‌تر روش ارائه شده نسبت به روش‌های اخیر است. روش ارائه شده در دیگر مسائل مربوط به فاکتورگیری ماتریس نیز کارا بوده و در همین راستا دو گسترش بهینه‌سازی حریصانه و پیاده‌سازی موازی از این روش نیز ارائه شده است که برای کاربردهایی نظیر فیلترینگ اشتراکی مناسب هستند.

کلمات کلیدی: شبکه‌های پیچیده، استنتاج ساختار، نمونه‌برداری از شبکه، بازسازی شبکه، نمونه‌برداری فشرده، مدل شبکه، پیش‌بینی لینک، فاکتورگیری ماتریس

فهرست مطالب

۴۵	تحلیل پیچیدگی زمانی	۴.۲.۳
۴۵	پیچیدگی فضایی	
۴۵	پیاده‌سازی با الگوریتم برآورد-بیشینه‌سازی	۵.۲.۳
۴۶		
۴۷	محاسبه مرحله برآورد	
۴۹	نتایج شبیه‌سازی	۳.۳
۴۹	تنظیمات شبیه‌سازی	۱.۳.۳
۴۹	مجموعه داده‌ها	
۵۰	روش‌های رقیب	
۵۱	تنظیمات	
۵۱	چگونگی انتخاب بعد فضای پنهان	
۵۲	ارزیابی دقت	۲.۳.۳
۵۳	ارزیابی زمان اجرا و مقیاس‌پذیری	۳.۳.۳
۵۴	ارزیابی دقت بر اساس میزان داده از دست رفته	۴.۳.۳
۵۵	کاربرد در فاکتورگیری از ماتریس	۴.۳
۵۷	بهینه‌سازی با انتخاب مؤلفه حریصانه: روش GSCD	۱.۴.۳
۶۱	تحلیل پیچیدگی زمانی و فضایی روش GSCD	۲.۴.۳
۶۱	آیا انتخاب مؤلفه حریصانه در پیش‌بینی لینک ممکن است؟	۳.۴.۳
۶۳	نتایج شبیه‌سازی	۴.۴.۳
۶۳	مجموعه داده‌ها	
۶۴	روش رقیب	
۶۴	مقیاس‌پذیری در فاکتورگیری از ماتریس	
۶۴	پیش‌بینی لینک با انتخاب مؤلفه حریصانه	
۶۶	پیاده‌سازی موازی روش GSCD	۵.۴.۳
۶۹	نتایج عملی میزان تأثیر موازی سازی بر عملکرد GSCD	
۷۰	۴ جمع‌بندی و نتیجه‌گیری	
۷۰	بازسازی شبکه تحت نمونه‌برداری فشرده	۱.۴
۷۰	پیش‌بینی لینک مقیاس‌پذیر با استفاده از فاکتورگیری ماتریس	۲.۴
۷۲	کتاب‌نامه	
۷۹	واژه‌نامه	
۸۲	پیوست آ: مقالات استخراج شده	

لیست تصاویر

۳	۱.۱ شبکه اینترنت [۳]
۶	۲.۱ یک مثال از مسئله بازسازی شبکه
۱۷	۱.۲ نمونه‌ای از انتشار اطلاعات و درخت متاظر با آبشار ایجاد شده
۲۰	۲.۲ سیستم خطی متاظر با اجرای m آبشار
۲۲	۳.۲ تحلیل پارامتر α در روش CS-NetRec
۲۲	۴.۲ تحلیل پارامتر β در روش CS-NetRec
۲۳	۵.۲ مقایسه وابستگی به تعداد آبشار در روش CS-NetRec
۲۴	۶.۲ مقایسه وابستگی به میزان تنک بودن شبکه در روش CS-NetRec
۲۵	۷.۲ مقایسه روش CS-NetRec با روش NetInf
۳۰ [۵۴]	۱.۳ تصویری از یک فضای ویژگی پنهان مشترک میان کاربران و فیلم‌ها
۴۰ [۷۷]	۲.۳ مقایسه درستنمایی برنولی و پواسون برای گراف‌های ساده در
۵۳	۳.۳ مقایسه زمان اجرا با روش SLPMF
۵۴	۴.۳ مقایسه زمان اجرا در برابر میزان AUC با روش SLPMF
۵۵	۵.۳ ارزیابی حساسیت دقت روش SLPMF به میزان داده از دست رفته
۶۴	۶.۳ مقایسه میزان کاهش مقدار تابع هدف در دو روش GSCD و CCD
۶۵	۷.۳ مقایسه رشد AUC در نسخه چرخشی و حریصانه روش SLPMF
۶۷	۸.۳ تأثیر بهینه‌سازی یک مؤلفه بر روی مؤلفه‌های دیگر
۶۹	۹.۳ بررسی میزان تأثیر موازی بر عملکرد GSCD

لیست جداول

۱.۱	جدول آماری برای نمایش ساختار تنک موجود در شبکه‌های واقعی	۳
۱.۲	برخی مطالعات مرتبط با کاربردهای نمونه‌برداری فشرده در تحلیل شبکه‌ها	۱۵
۱.۳	برخی مطالعات مرتبط با کاربردهای تکمیل ماتریس و فاکتورگیری ماتریس در تحلیل شبکه‌ها و پیش‌بینی لینک	۳۷
۲.۳	مجموعه‌داده‌های انتخاب شده برای ارزیابی عملکرد الگوریتم SLPMF	۵۰
۳.۳	مقایسه مقادیر AUC با روش SLPMF	۵۲
۴.۳	مجموعه‌داده‌های انتخاب شده برای ارزیابی عملکرد الگوریتم GSCD	۶۳

فصل ۱

مقدمه

۱.۱ معرفی موضوع تحقیق

ارتباطات، بخشی جدانشدنی و حیاتی در جوامع امروزی هستند. تعاملات بشر، از روابط دوستی ساده تا دادوستد اطلاعات در دنیای دیجیتالی، همگی نقشی مهم در شکل دهی و پیشبرد زندگی انسان‌ها دارند. از این رو، بسیاری از شاخه‌ها و گرایش‌های علوم، جنبه‌های مختلفی از ارتباطات و تعاملات در جوامع گوناگون را مورد بررسی قرار داده‌اند. از این بین می‌توان به مواردی چون جامعه‌شناسی، روان‌شناسی، اقتصاد و علوم سیاسی در حوزه علوم انسانی و آمار، فیزیک و علوم کامپیوتر در حوزه علوم پایه و فنی اشاره کرد. یکی از عمومی‌ترین رویکردها برای تحلیل و بررسی ارتباطات در یک محیط خاص، استفاده از ایده شبکه - یک الگوی ارتباطی میان مجموعه‌ای از عناصر تشکیل دهنده - است [۱]. به عبارتی دیگر، نمایش گرافی (مجموعه‌ای از رئوس و یال‌ها) عمومی‌ترین نوع نمایش ارتباطات است.

در طی دوهه اخیر، مطالعات بسیاری بر روی جنبه‌های مختلف شبکه‌ها انجام شده است. هر چند با پیشرفت فناوری و ظهرور اینترنت و به دنبال آن شبکه جهانی وب و نیز شبکه‌های اجتماعی برخط که ساختار شبکه‌ای بنیادی دارند، این مطالعات اهمیتی ویژه پیدا کرده‌اند. بسیاری از سامانه‌ها و جوامع در دنیای امروزی مانند شبکه اینترنت، صفحات وب و تعاملات اجتماعی می‌توانند به صورت شبکه‌ای از گره‌های به هم پیوسته مدل‌سازی شوند. یال‌های میان گره‌ها که در واقع نمایان‌گر ارتباطات میان گره‌ها است، با توجه به زمینه مطالعه تعریف می‌شود. به عنوان مثال در شبکه وب، لینک‌های صفحات به یکدیگر، یال‌های شبکه را می‌سازند (شکل ۱.۱) و یا در یک شبکه اجتماعی برخط،

رابطه دوستی میان دو شخص می‌تواند به وجود یال مابین دو گره مربوط به آن دو شخص در شبکه منجر شود. یکی از وضعیت‌هایی که ممکن است هنگام مطالعه این نوع شبکه‌ها با آن برخورد شود، این است که با وجود شناخته شده بودن گره‌ها، الگوی ارتباطی میان گره‌های گراف ناقص است. در چنین شرایطی، ارائه روشی که قادر به بازسازی ساختار شبکه باشد، از اهمیت ویژه‌ای برخوردار است. این مسئله، استنتاج ساختار شبکه و یا تکمیل شبکه^۲ نام دارد.

شبکه‌هایی که در دنیای واقعی وجود دارند، اغلب دارای ساختاری تنک هستند. به این معنی که تعداد یال‌های موجود در گراف شبکه از تعداد تمامی یال‌های ممکن بسیار کمتر است. این حقیقت فارغ از این که شبکه‌ها در چه کاربردی مطرح می‌شوند، تقریباً در تمامی شبکه‌های واقعی مشاهده می‌شود. جدول ۱.۱ شاهدی بر این ادعا است. میزان تنک بودن در این جدول برابر یک منهای خارج قسمت تعداد یال‌ها به تعداد کل یال‌های ممکن در نظر گرفته شده است.

یکی از پایه‌ای‌ترین مفاهیم در علوم مهندسی، باور بر وجود مؤلفه‌هایی کوچک و ساده در یک سامانه پیچیده و بزرگ است که نقش اصلی را در شکل‌گیری رفتار کلی دارند. روش‌هایی مانند تحلیل مؤلفه‌های اصلی^۳، تحلیل مؤلفه جداساز پنهان^۴، تحلیل مؤلفه‌های مستقل^۵، قسمت‌بندی برداری^۶ و نمونه‌برداری فشرده^۷ از نمونه روش‌هایی در یادگیری ماشین و پردازش سیگنال هستند که از این مفهوم استفاده می‌کنند [۲]. هدف ما در این پایان‌نامه بهره‌گیری از این روش‌ها برای بهبود دقت و سرعت در استنتاج ساختار شبکه‌ها است.

۲.۱ اهمیت و کاربردهای موضوع تحقیق

جمع آوری داده اولین گام در تحلیل یک شبکه محسوب می‌گردد. اغلب شبکه‌های امروزی، ساختاری بزرگ و پیچیده دارند؛ بدین معنی که تعداد گره‌ها و ارتباطات میان آن‌ها بسیار بزرگ است. به عنوان مثال، شبکه فیسبوک^۸ در سال ۲۰۱۲ دارای بیش از یک میلیارد کاربر بوده است [۱۱]. به همین دلیل، گردآوری اطلاعات کامل از یک شبکه به دلیل سربار زیاد فرآیند جمع‌آوری، امکان‌پذیر

Network Completion^۹

Principal Component Analysis^{۱۰}

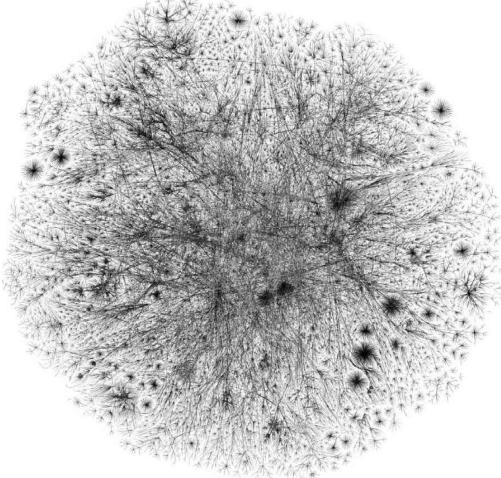
Latent Discriminant Analysis^{۱۱}

Independent Component Analysis^{۱۲}

Vector Quantization^{۱۳}

Compressed Sensing^{۱۴}

Facebook^{۱۵}



شکل ۱.۱: شبکه اینترنت [۳]

جدول ۱.۱: جدول آماری برای نمایش ساختار تنک موجود در شبکه‌های واقعی

نام شبکه	اینترنت	نوع شبکه	تعداد گره‌ها	تعداد یال‌ها	میران تنک بودن
[۴] Yeast	زیستی	۲,۳۶۱	۶,۹۱۴	۰/۹۹۷۵۲	
[۵] ego-Facebook	اجتماعی برخط	۴,۰۳۹	۸۸,۲۳۴	۰/۹۸۹۱۸	
[۶] GrQc	نویسنده‌گان همکار	۵,۲۴۲	۱۴,۴۹۰	۰/۹۹۸۹۱	
[۷] FacebookL1	اجتماعی برخط	۵,۷۹۳	۳۰,۷۵۳	۰/۹۹۸۱۷	
[۸] AS	اینترنت	۱۱,۴۹۲	۲۳,۴۰۹	۰/۹۹۹۶۵	
[۹] COND-MAT	نویسنده‌گان همکار	۲۳,۱۳۳	۹۳,۴۹۷	۰/۹۹۹۶۵	
[۱۰] P2P	نظیر به نظری	۲۶,۵۱۸	۶۵,۳۶۹	۰/۹۹۹۸۱	
[۱۱] ego-Gplus	اجتماعی برخط	۱۰۷,۶۱۴	۱۳,۶۷۳,۴۵۳	۰/۹۹۷۶۴	
[۱۲] web-Google	وب	۸۷۵,۷۱۳	۵,۱۰۵,۰۳۹	۰/۹۹۹۹۹	
[۱۳] roadNet-CA	راه‌های ارتباطی	۱,۹۶۵,۲۰۶	۵,۵۳۳,۲۱۴	۰/۹۹۹۹۹	
[۱۰] com-Friendster	اجتماعی برخط	۶۵,۶۰۸,۳۶۶	۱,۸۰۶,۰۶۷,۱۳۵	۰/۹۹۹۹۹	

نخواهد بود. لذا تحلیل شبکه برمبنای داده‌های ناکامل حاصل از نمونه‌برداری شبکه انجام خواهد شد. گرچه نمونه‌برداری از شبکه به نحوی که نمونه حاصل نماینده خوبی از تمام داده‌های شبکه باشد، چالش‌های خاص خود را دارد، داده ناکامل باعث می‌شود بسیاری از تحلیل‌ها در مورد شبکه مانند تخمین معیارهای اندازه‌گیری مختلف، دقیق نباشد [۱۲، ۱۳]. در ادامه به نمونه‌هایی از کاربردهای استنتاج ساختار شبکه اشاره می‌کنیم:

- مطالعه انواع سامانه‌های پیشنهاددهنده کالا به کاربران (مانند وبسایت‌های خرید اینترنتی کالا) و یا کاربران به کاربران (مانند شبکه‌های اجتماعی) ارتباطی تنگاتنگ با مسئله استنتاج ساختار دارد. به این معنی که می‌توان بر اساس ساختار پیش‌بینی شده گراف، کالا یا کاربران دیگر را به کاربری پیشنهاد نمود.

- در مطالعه شبکه‌های زیستی، اگرچه پیشرفت‌های زیادی جهت بهبود ابزارهای شناسایی شبکه تعاملات پروتئینی انجام شده، گزارش‌های زیادی مبنی بر عدم دقت کامل در آن‌ها منتشر شده است [۱۴].
 - در مطالعه بر روی شبکه‌های اجتماعی، به خصوص شبکه‌های اجتماعی برخط، وجود داده ناکامل پس از جمع‌آوری تقریباً اجتناب‌ناپذیر است که این ناشی از دلایل بسیاری چون سربار زیاد فرایند جمع‌آوری، امنیت و یا شخصی بودن حریم کاربران است. تحلیل چنین داده‌ای باعث تخمين نادرست ویژگی‌های مختلف شبکه خواهد شد [۱۵].
 - گاهی هدف تنها بازسازی بخشی از شبکه است. به عنوان مثال در یک شبکه اجتماعی برخط گاهی تنها هدف این است که شبکه دوستان در یک دانشگاه بازیابی شود. در شناسایی دانشجویان چندان مشکلی وجود ندارد، اما شناسایی روابط میان آن‌ها سربار بسیار زیادی را خواهد داشت. بنابراین بازسازی ساختار شبکه نه تنها در شبکه‌های بزرگ، بلکه شبکه‌های با مقیاس‌های به مراتب کوچک‌تر نیز کاری حائز اهمیت است.
 - ارتباط در عملکرد و آناتومی بخش‌های مختلف مغز انسان باعث شده تا مغز به عنوان یک شبکه بزرگ و پیچیده با ساختاری نامعلوم شناخته شود. با استفاده از تئوری گراف در تفسیر این شبکه، می‌توان درکی بهتر از ساختار و عملکرد مغز بدست آورد [۱۶]. بازسازی شبکه مغز از جمله مهم‌ترین مسائل واقعی موجود در بازسازی ساختار شبکه‌ها می‌باشد.
 - موتورهای جستجو همواره با مشکل جمع‌آوری داده در مورد ساختار لینک‌ها بین وبسایت‌های اینترنتی مواجه هستند. پیش‌بینی لینک‌های مابین وبسایت‌ها می‌تواند موجب بهبود در کارایی و سرعت موتورهای جستجو شود.
- بنابراین داده شبکه‌ای معمولاً ناکامل است و ارائه روش‌هایی برای مقابله با این مشکل کاربرد و اهمیت ویژه‌ای را دارد.

۳۰۱ چالش‌ها

پیش‌بینی ساختار شبکه با توجه به مقیاس بزرگ شبکه‌ها، پیچیدگی آن‌ها و اطلاعات ناکافی از ساختار شبکه، کاری دشوار به شمار می‌آید. از جمله چالش‌های موجود در این راستا عبارتند از:

۱. مقیاس بزرگ شبکه‌ها: کار با مجموعه داده‌های بزرگ، سریار پیش‌بینی ساختار را افزایش می‌دهد. زمانی به کارگیری یک راه حل برای پیش‌بینی ساختار مقرن به صرفه است که زمان اجرای قابل قبولی داشته باشد.

۲. ارائه راه حلی فارغ از زمینه مطالعه: ارائه راه حلی که بتوان آن را در زمینه‌های متفاوتی به کار گرفت، می‌تواند تأثیر به سازی بر پرکاربرد بودن آن داشته باشد. چرا که بدون هیچ اطلاعاتی از یک زمینه مطالعاتی خاص و فقط با اعمال تغییراتی اندک برای نگاشت راه حل به آن زمینه، می‌توان شبکه مورد نظر را بازسازی نمود.

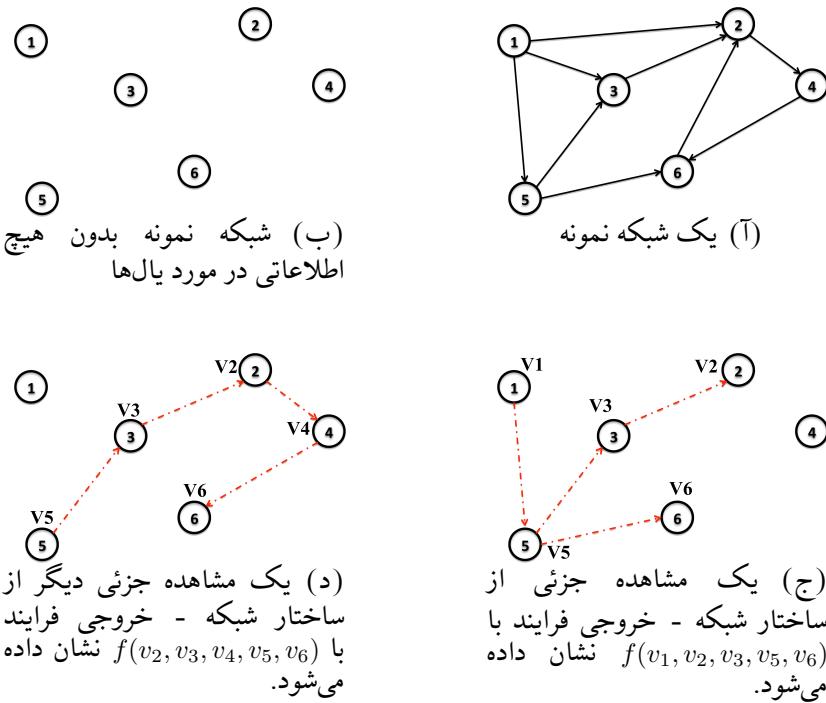
۳. داده مورد استفاده: شناسایی و معرفی این که از چه داده‌ای می‌توان برای پیش‌بینی ساختار شبکه استفاده کرد، از اساسی‌ترین قدم‌ها برای استنتاج ساختار شبکه است. قطعاً راه حل پیشنهادی برای این مسئله تابعی از چگونگی این داده‌ها است.

۴.۱ دستاوردهای پایان‌نامه

در این پایان‌نامه پیش‌بینی ساختار در شبکه در دو مورد خاص، بسته به داده‌های که از شبکه در دسترس است، ارائه شده است.

۱۰.۱ مسئله بازسازی شبکه^۹

مسئله نخستی که ما در این پایان‌نامه بررسی می‌کنیم، مسئله بازسازی شبکه است: یک شبکه با تعدادی گره که یال‌ها در آن نامعلوم می‌باشد، داده شده است. آیا ممکن است که با استفاده از داده‌های استخراج شده حاصل از گره‌ها، ساختار شبکه را پیش‌بینی کرد؟ به عنوان یک مثال، شکل ۱.۲ را در نظر بگیرید. یک شبکه نمونه در شکل ۱.۲(آ) و همان شبکه بدون حضور یال‌ها در شکل ۱.۲(ب) نشان داده شده است که این، وضعیتی مشابه با پیش‌فرض مسئله بازسازی شبکه است. هدف، بازسازی شبکه مورد نظر بر اساس مشاهدات جزئی^{۱۰} در شکل‌های ۱.۲(ج) و ۱.۲(د) است. در این مثال، خروجی‌های حاصل از چندین اجرای یک فرایند مشخص بر روی گراف همان مشاهدات جزئی در نظر گرفته شده‌اند. هر اجرای فرایند یک مقدار مشخص برای هر گره اندازه‌گیری کرده و خروجی



شکل ۲.۱: یک مثال از مسئله بازسازی شبکه

(آ) یک شبکه نمونه. (ب) شبکه نمونه با گره‌های داده شده و یال‌های نامعلوم. این تنها اطلاعاتی است که در مورد توپولوژی شبکه در مسئله بازسازی شبکه وجود دارد. (ج،د) مشاهدات جزئی نمونه (مقادیر مربوط به گره‌ها و خروجی فرایند) بر روی گراف برای ساختار شبکه مورد نظر.

آن، تابعی از این مقادیر خواهد بود. بر اساس مقادیر مربوط به گره‌ها و خروجی فرایند، ساختار شبکه باقیستی بازسازی شود. بسته به زمینه مطالعه، مشاهدات ممکن است پیچیده باشند.

روش CS-NetRec^{۱۱} با بهره‌گیری از چارچوب نمونه‌برداری فشرده، به عنوانی روشی نوین برای مسئله بازسازی شبکه ارائه شده است. برای فرمول‌بندی و ارائه بهتر ایده‌ها، روش مذکور در زمینه شبکه‌های اطلاعاتی و انتشار^{۱۲} بیان شده است. در این روش برای یک شبکه زیرین انتشار اطلاعات^{۱۳} و با استفاده از معیارهای احتمالاتی معرفی شده بر روی شبکه‌های انتشار (مانند احتمال انتشار آبشر اطلاعاتی^{۱۴})، ساختار شبکه زیرین استنتاج می‌شود. به طور دقیق‌تر، پس از در نظر گرفتن اجراء‌های مختلف آبشرهای اطلاعاتی به عنوان واحد اندازه‌گیری، اجرای هر آبشر را به واسطه معیارهای احتمالاتی چون احتمال گذر^{۱۵} و احتمال آبشر به صورت ضرب داخلی دو بردار مدل

Compressive Sensing for Network Reconstruction^{۱۱}

Information and diffusion networks^{۱۲}

Information Diffusion^{۱۳}

Cascade diffusion probability^{۱۴}

Transmission probablity^{۱۵}

می‌کنیم. سپس با اجرای تعدادی از آبشارها، یک دستگاه خطی زیرمعین^{۱۶} تشکیل می‌شود که در آن یال‌های موجود در شبکه به عنوان بردار مجهول در نظر گرفته می‌شود. بنابراین استنتاج ساختار شبکه تبدیل به یافتن پاسخ مناسب برای این دستگاه خطی خواهد شد. برای یافتن چنین پاسخی، از تئوری نمونه‌برداری فشرده استفاده کرده و پس از تعریف یکتابع هدف^{۱۷}، از طریق یکی از روش‌های معمول برای بهینه‌سازی پاسخ مناسب را ارائه خواهیم داد.

ارزیابی‌های انجام شده بر روی مجموعه داده‌های مصنوعی و واقعی نشان می‌دهد که روش ارائه شده قابلیت استنتاج بخش اعظمی از ساختار شبکه حتی در حضور تعداد مشاهدات پایین (در اینجا تعداد آبشارهای پایین) را دارد. به عبارت دیگر روش ارائه شده ساختار شبکه را با تعداد مشاهداتی برابر با حدود نیمی از تعداد یال‌های موجود در شبکه، با دقیقی بالا بازسازی می‌کند. این در حالی است که تعداد یال‌های ممکن برای بازسازی حدود ۵ تا ۱۰ برابر تعداد مشاهدات است. به عنوان بخشی دیگر از ارزیابی‌ها، تأثیر پارامترهای مختلف مشاهدات (طول و سرعت آبشارهای اطلاعاتی) در دقت روش ارائه شده بررسی شده‌اند. به علاوه، تأثیر میزان تنک بودن شبکه و روند افزایشی و یا کاهشی دقت با توجه به آن بررسی شده است. نتایج این بررسی نشان می‌دهد که با افزایش میزان تنک بودن شبکه در یک شبکه با گره‌های ثابت، کارایی روش ارائه شده افزایش می‌یابد. همچنین کارایی روش ارائه شده با یکی از روش‌های کلاسیک استخراج شبکه انتشار به نام NetInf مقایسه شده است. نتایج این مقایسه نشان می‌دهد که در روش ارائه شده تا ۱۰ درصد افزایش دقت در بازسازی شبکه نسبت به NetInf وجود دارد.

در این پژوهش، انتشار اطلاعات به عنوان بستر ارائه روش معرفی شده است. اما روش ارائه شده می‌تواند در زمینه‌های دیگر نیز به عنوان یک راه حل بهینه و کارا مورد استفاده قرار بگیرد. به طور خلاصه، دستاوردهای اصلی این پژوهش عبارتند از:

- ارائه یک روش جدید و عمومی بر اساس چارچوب ریاضی نمونه‌برداری فشرده برای استنتاج ساختار یک شبکه برای نخستین بار، که ویژگی‌هایی چون تنک بودن ساختار شبکه موجب کارایی هر چه بیشتر آن می‌شود.
- نمایش توانایی روش ارائه شده در زمینه شبکه‌های انتشار برای استخراج شبکه زیرین بدون داشتن هیچ گونه اطلاعاتی از ساختار شبکه اصلی.

نتایج مربوط به این پژوهش در مراجع [۱۷، ۱۸] به چاپ رسیده است و همچنین در پیوست آنیز قابل

Under-determined^{۱۹}

Objective Function^{۲۰}

مشاهده است.

۲۰۴۰۱ مسئله پیش‌بینی لینک^{۱۸}

هر چند مسئله پیش‌بینی لینک^{۱۹} تحت نام‌ها و پیش‌فرض‌های زیادی مطرح شده است، فرض ما در این مسئله این است که تنها داده‌ای که از شبکه موجود است، تعدادی از رئوس شبکه و تعدادی از لینک‌هایی است که مابین آن‌ها قرار دارند، است. هدف، پیش‌بینی لینک‌های مابین این رئوس است. به عبارت دیگر، داده در دسترس، ماتریس مجاورت گراف مشاهده شده است.

هر چند روش‌های گوناگونی برای حل این مسئله ارائه شده است، در این پایان‌نامه مسئله پیش‌بینی لینک به عنوان یک مسئله فاکتورگیری از ماتریس مدل شده و روشی کارا برای آن ارائه خواهد شد. به طور دقیق‌تر، با اختصاص یک مدل احتمالاتی پواسون به گراف شبکه، مسئله پیش‌بینی لینک به صورت یک مسئله فاکتورگیری از ماتریس مطرح شده و برای پیش‌بینی لینک، پارامترهای مدل احتمالاتی استنتاج می‌شوند. روش استنتاج بر اساس ماکزیمم درستنمایی^{۲۰} است؛ هر چند مدل ارائه شده برای روش‌های دیگر استنتاج مانند ماکزیمم احتمال پسین^{۲۱} نیز مناسب می‌باشد. برای بهینه‌سازی روش کاهش مؤلفه بلوکی^{۲۲} به کار گرفته شده است. همچنین نشان داده شده که بهینه‌سازی تابع هدف تحت مدل احتمالاتی فرض شده، با استفاده از ساختار تنک شبکه‌ها بسیار سریع‌تر از مدل‌های احتمالاتی مطرح‌تر مانند برنولی-لジستیک^{۲۳} انجام می‌پذیرد. بنا بر دانش ما، پیش‌بینی لینک بر اساس مدل احتمالاتی پواسون برای بالا بردن دقت و سرعت تاکنون انجام نشده است. در این پایان‌نامه، نشان داده شده است که روش ارائه شده در زمانی خطی نسبت به رئوس شبکه انجام می‌پذیرد. همچنین با تغییراتی بر روی روش ارائه شده، نتایجی بهتر از روش‌های ارائه شده در فاکتورگیری از ماتریس تحت تابع خطای KL-Divergence^{۲۴}، بدست آمده است.

نتایج بدست آمده بر روی داده‌های واقعی مربوط به زمینه‌های مختلف (شبکه‌های زیستی، اجتماعی،

Link Prediction Problem^{۱۸}

^{۱۹} در این پایان‌نامه ممکن است دو کلمه "یال" و "لینک" به جای هم به کار برد شوند.

Maximum Likelihood^{۲۰}

Maximum A Posteriori^{۲۱}

Block Coordinate Descent^{۲۲}

Bernoulli - Logistic^{۲۳}

KL - Divergence Generalized^{۲۴}

نویسنده‌گان همکار، اینترنت و نظری به نظیر به ^{۲۵})، نشان‌دهنده دقت بهتر روش مذکور از روش‌های بدون ناظر ^{۲۶} و با ناظر ^{۲۷} رقیب است. روش ارائه شده تا پانزده برابر سریع‌تر از روش‌های با ناظر در مجموعه داده‌های بزرگ می‌باشد. به علاوه نتایج نشان‌دهنده عملکرد مطلوب روش ارائه شده در حضور داده بسیار کم است.

به طور خلاصه، دستاوردهای اصلی این پژوهش عبارتند از:

- ارائه روشی سریع و دقیق برای پیش‌بینی لینک با استفاده از مدل‌سازی پواسون و استفاده از ساختار تنک شبکه‌ها.

- فرموله‌سازی مسئله پیش‌بینی لینک در چارچوب مسئله فاکتورگیری ماتریس که استفاده از روش مذکور را در کاربردهای نظری فیلترینگ اشتراکی ^{۲۸}، اندیس‌سازی معنایی پنهان ^{۲۹} و کاهش بعد ^{۳۰} را ممکن می‌سازد.

نتایج مربوط به این پژوهش در پیوست آ ^{۳۱} ۳۲ نیز قابل مشاهده است.

۱.۵.۱ ساختار پایان‌نامه

ساختار مابقی پایان‌نامه به صورت زیر تنظیم شده است:

در فصل دوم، در ابتدا تعدادی از مطالعات و مسائلی در زمینه تحلیل شبکه‌ها که از نمونه‌برداری فشرده به عنوان ابزار در رهیافت خود استفاده کرده‌اند آورده شده است. همچنین، پس از بیان مسئله بازسازی شبکه و مفاهیم مقدماتی مربوط به شبکه‌های انتشار، مسئله مورد نظر فرموله شده و روش پیشنهادی بر اساس نمونه‌برداری فشرده ارائه می‌گردد.

Peer-to-Peer^{۲۵}

Unsupervised^{۲۶}

Supervised^{۲۷}

Collaborative Filtering^{۲۸}

Latent Semantic Indexing^{۲۹}

Dimensionality Reduction^{۳۰}

^{۲۱} ارسال شده به کنفرانس (ICDE'14) IEEE International Conference on Data Engineering

^{۳۲} ارسال شده به ژورنال (IF: 3.297)

در فصل سوم، ابتدا مسئله پیش‌بینی لینک و تعدادی از مسائل مشابه آن بیان شده است. همچنین، مفاهیم مربوط به تئوری فاکتورگیری و تکمیل ماتریس و تعدادی از پژوهش‌های اخیر در حوزه شبکه‌ها که از این تئوری بهره گرفته‌اند، بررسی شده است. در ادامه روش پیشنهادی و نتایج شبیه‌سازی ارائه خواهد شد.

در فصل چهارم، به جمع‌بندی و بیان کارهای آینده خواهیم پرداخت.

فصل ۲

بازسازی شبکه تحت نمونهبرداری فشرده

۱.۲ مقدمه‌ای بر نمونهبرداری فشرده

نمونهبرداری فشرده، یکی از حوزه‌های تحقیقاتی جدید در پردازش سیگنال و تئوری اطلاعات است که اخیراً مورد توجه بسیاری از محققین قرار گرفته است [۲۰، ۱۹]. ایده اصلی این بحث انجام همزمان نمونهبرداری و فشرده‌سازی می‌باشد. به طور دقیق‌تر، تئوری این بحث، این مطلب را اظهار می‌کند که در یک نمایش مناسب (مانند بردار تنک^۱ یا ماتریس رتبه پایین^۲) داده‌ی نمونهبرداری شده از یک شیء، می‌تواند تمامی اطلاعات در مورد آن شیء را داشته باشد، با این تفاوت که میزان داده نمونهبرداری شده می‌تواند بسیار کمتر از حددهای پایین نمونهبرداری (مانند نرخ نمونهبرداری نایکوییست^۳) باشد. پیشرفت‌های اصلی در زمینه نمونهبرداری فشرده با پژوهش‌های پایه‌ای در مراجع [۲۱، ۲۲، ۲۳] آغاز شد. در این پژوهش‌ها نشان داده شد که ترکیب بهینه‌سازی نرم یک^۴ و ماتریس‌های اتفاقی^۵ می‌تواند موجب بازسازی بهینه بردارهای تنک شود. هر چند نویسنده‌گان در این مراجع این نکته را هم متنذکر می‌شوند که ایده‌های نمونهبرداری فشرده در سطح وسیعی می‌توانند به کار گرفته شوند. فرض کنید بردار $x_{n \times 1}$ یک بردار حقیقی k -تنک باشد، به این معنی که فقط k درایه از آن غیر صفر باشد. اگر این بردار مبین یک سیگنال گستته و متناهی باشد و $\|\cdot\|$ بردار نمونه‌های آن، می‌توان

Sparse vector^۱

Low-rank matrix^۲

Nyquist^۳

l_1 -minimization^۴

Random matrices^۵

دستگاه خطی زیر را حاصل این نمونهبرداری دانست:

$$y_{m \times 1} = A_{m \times n} x_{n \times 1} \quad (1.2)$$

که ماتریس A ماتریس اندازه‌گیری ^۶ نامیده می‌شود. تعداد سطرهای این ماتریس در حقیقت نرخ نمونهبرداری را نشان می‌دهد. اگر شرایط به گونه‌ای باشد که نتوان بدون از دست دادن هیچ گونه اطلاعات، نمونهبرداری را انجام داد، در این صورت $n < m$ خواهد بود. بنابراین دستگاه خطی فوق یک دستگاه زیرمعین بوده و با توجه به نظریه اساسی جبرخطی دارای بی‌شمار جواب است و بازسازی x امکان پذیر نخواهد بود. در حالت کلی، پاسخ این سیستم خطی از مسئله بهینه‌سازی حداقل مربعات زیر حاصل خواهد شد:

$$x^* = \arg \min_x \|Ax - y\|_2^2 \quad (2.2)$$

اضافه کردن قید تنک بودن به بردار x مجموعه جواب‌ها را محدود می‌کند. تنکترین جواب مسئله فوق از حل مسئله بهینه‌سازی زیر حاصل می‌شود:

$$\min_x \|x\|_0 \quad s.t. \quad Ax = y \quad (3.2)$$

که منظور از $\|\cdot\|_0$ نرم صفر بردار (تعداد درایه‌های غیر صفر) است. اما مسئله فوق NP-Hard بوده و حل آن در ابعاد بالا عملی نیست. نزدیکترین تقریب به نرم صفر، نرم یک است. بنابراین رابطه (۳.۲) تبدیل به رابطه زیر خواهد شد:

$$\min_x \|x\|_1 \quad s.t. \quad Ax = y \quad (4.2)$$

با ترکیب روابط (۲.۲) و (۴.۲) می‌توان مسئله بهینه‌سازی زیر را تعریف کرد:

$$\min_x \|x\|_1 + \|Ax - y\|_2^2. \quad (5.2)$$

این تغییر درتابع هدف بهینه سازی (که LASSO نامیده می‌شود [۲۴، ۲۵]) با تبدیل کردن مسئله به یک مسئله بهینه‌سازی غیر مقید، علاوه بر این که ما را قادر به حل آن با روش‌های سریع‌تر

(مانند انواع روش‌های کاهش گرادیان^۷) می‌نماید، این اجازه را به ما می‌دهد که بتوان حتی در حضور نویز و یا مقادیر بریده شده^۸ پاسخی تقریبی برای سیستم خطی پیدا کرد.

برای حل مسئله (۵.۲) راه حل‌های مختلفی ارائه شده است. برای رابطه (۴.۲) راه حل‌هایی به کمک برنامه‌ریزی خطی^۹ ارائه شده است و [۲۳]. همچنین روش‌های حریصانه و تکراری نیز پیش گرفته شده‌اند که به "جستجوی پایه"^{۱۰} و "جستجوی تطابق"^{۱۱} معروفند [۲۶]. همچنین قضایای مختلفی برای حدهای پایین نمونه‌برداری و شکل ماتریس اندازه‌گیری در شرایط گوناگون مطرح شده‌اند [۲۷، ۲۲].

۲.۲ کاربردهای نمونه‌برداری فشرده در تحلیل شبکه‌ها

از آنجا که ما در روش ارائه شده در این پژوهش از تئوری نمونه‌برداری فشرده استفاده می‌کنیم، دسته‌ای از کارهایی که در حوزه تحلیل شبکه‌ها با استفاده از نمونه‌برداری فشرده در سال‌های اخیر انجام شده‌اند را مورد بررسی قرار می‌دهیم. نمونه‌برداری فشرده اغلب در زمینه پردازش سیگنال و عکس مورد استفاده قرار گرفته است [۲۸، ۲۹، ۳۰]^{۱۲} و استفاده از آن در حوزه تحلیل شبکه مراحل اولیه پیشرفت را سپری می‌کند.

استفاده از نمونه‌برداری فشرده در حوزه شبکه، بیشتر در مطالعه شبکه‌های حسگر بی‌سیم انجام شده است [۳۱، ۳۲، ۳۳]. چالش اصلی در این زمینه شامل بازسازی یک سیگنال اطلاعاتی (مانند دما در یک محیط) از داده جمع‌آوری شده از شبکه‌ای از حسگرهای است.

مرجع [۳۴، ۳۵، ۳۶] از نمونه‌برداری فشرده در حوزه مانیتورینگ کارایی شبکه‌های کامپیوترا برهه برده‌اند. به طور دقیق‌تر، در این پژوهش شناسایی لینک‌هایی که موجب تأخیر در یک شبکه کامپیوترا می‌شوند^{۱۲} از طریق اندازه‌گیری‌های انتها به انتها (مانند تأخیر فرستادن یک بسته از یک گره شبکه به گره دیگر) انجام می‌شود. این مسئله تحت عنوان توموگرافی شبکه^{۱۳} مطرح می‌شود. در این پژوهش‌ها با فرض این‌که گراف شبکه به طور کافی متصل است، از قدم‌زن تصادفی برای

Gradient Descent^{۱۴}

Truncated values^{۱۵}

Linear programming^{۱۶}

Basis pursuit^{۱۷}

Matching Pursuit^{۱۸}

Congested links^{۱۹}

Network Tomography^{۲۰}

جمع‌آوری نمونه‌ها استفاده کرده است و حد پایین $O(k \log n)$ را برای تعداد مسیرهای طی شده با قدم‌های تصادفی تعیین کرده است.

همچنین در تحلیل ترافیک شبکه‌های کامپیوتری، می‌توان مراجع [۳۷، ۳۸] را ذکر کرد. در [۳۷] از نمونه‌برداری فشرده برای کاهش هزینه حافظه در روتراها و سوئیچ‌ها استفاده شده است. همچنین در [۳۸] برای تعریف یک مکانیزم درونیابی جهت بازسازی مقادیر از دست رفته در ماتریس ترافیکی شبکه، از تئوری نمونه‌برداری فشرده استفاده شده است.

یکی از محدود کارهایی که در حوزه شبکه‌های نظیر به نظر انجام شده و از تئوری نمونه‌برداری فشرده استفاده کرده است، مرجع [۳۹] است. سوال اساسی در این کار عبارت است از: چطور هر گره شبکه (یک نظیر) می‌تواند دسترسی محلی به اطلاعات کامل شبکه، که بر روی لینک‌های شبکه توزیع می‌شوند، داشته باشد. در این مرجع، با استفاده از قدمزن تصادفی، هر گره شبکه ماتریس اندازه‌گیری خود را با توجه به نمونه‌ها تشکیل می‌دهد و بردار تنک متناظر با اطلاعات کامل شبکه را بازسازی می‌کند.

دو پژوهش دیگر که به این پایان‌نامه بیشتر نزدیک هستند، مراجع [۱۶، ۴۰] هستند. هدف در هر دوی این مراجع، استفاده از نمونه‌برداری فشرده در جهت پیش‌بینی توپولوژی شبکه است، هر چند زمینه مطالعه و پیش‌فرضها در آن‌ها کاملاً با یکدیگر متفاوت است. در [۱۶] یک روش نوین بر اساس برآش خطی با جریمه^{۱۴} برای تخمین همبستگی جزئی تنک^{۱۵} بین نواحی مختلف مغز ارائه شده است. در این کار با استفاده از تئوری نمونه‌برداری فشرده، بازسازی قابل قبولی از شبکه میان نواحی مختلف مغز انجام می‌شود. اگرچه راه حل ارائه شده نمی‌تواند در همه زمینه‌ها عملی باشد. چرا که در این پژوهش و زمینه مربوط به آن، امکان دسترسی به کل شبکه مورد نظر (اگرچه با ساختاری نامعلوم) وجود دارد. بنابراین این امکان وجود دارد تا معیار همبستگی قابل محاسبه باشد. اما در اکثر شبکه‌ها مانند شبکه‌های اجتماعی دسترسی به چند نمونه از شبکه وجود ندارد.

در [۴۰] یک روش برای پیش‌بینی توپولوژی شبکه بر اساس داده‌های یک بازی تکاملی^{۱۶} با استفاده از نمونه‌برداری فشرده ارائه شده است. ایده اصلی راه حل ارائه شده در رابطه بین استراتژی بازیکنان (گره‌های شبکه) و سود^{۱۷} آن‌ها است. تعاملات بین بازیکنان می‌تواند با یک ماتریس $N \times N$ نمایش داده شود و تنک بودن همسایگی (تعداد گره‌های مجاور) یک گره باعث می‌شود که

Penalized linear regression^{۱۴}

Sparse partial correlation^{۱۵}

Evolutionary game^{۱۶}

Payoff^{۱۷}

استفاده از نمونهبرداری فشرده در این مسئله عملی باشد.

جدول ۱.۲ خلاصه‌ای از آن چه در این بخش گفته شد را نمایش می‌دهد.

جدول ۱.۲: برخی مطالعات مرتبط با کاربردهای نمونهبرداری فشرده در تحلیل شبکه‌ها

پیش‌فرض‌های اساسی	مسئله موردبررسی	مراجع	زمینه‌مطالعه
—	بازسازی یک سیگنال اطلاعاتی از داده جمع‌آوری شده از شبکه‌ای از حسگرها	[۳۱، ۳۲، ۳۳]	شبکه‌های حسگر بی‌سیم
تأثیرهای تجمعی، متصل بودن گراف	شناسایی لینک‌های تأخیردار، نرخ گم شدن بسته‌ها	[۳۴، ۳۵، ۳۶]	مانیتورینگ کارایی شبکه‌های کامپیوتری
—	کاهش هزینه حافظه در روتراها و سوئیچ‌ها، بازسازی مقادیر از دست رفته در ماتریس ترافیکی شبکه	[۳۷، ۳۸]	تحلیل ترافیک شبکه‌های کامپیوتری
تشکیل ماتریس اندازه‌گیری برای هر گره	چگونگی دسترسی محلی هر گره به اطلاعات کامل شبکه	[۳۹]	شبکه‌های نظریه‌نظری
تعریف همبستگی و امکان دسترسی به چند نمونه از شبکه	بازسازی از طریق برآش خطی با جریمه برای تخمین همبستگی جزئی تنک	[۱۶]	تشخیص توپولوژی شبکه مغز
وجود داده بر اساس بازی تکاملی بر روی شبکه	نمایش تعاملات به صورت ماتریسی و استفاده از تنک بودن همسایگی یک گره	[۴۰]	تشخیص توپولوژی شبکه

۳.۲ بیان رسمی مسئله

۱۰۳.۲ معرفی مسئله

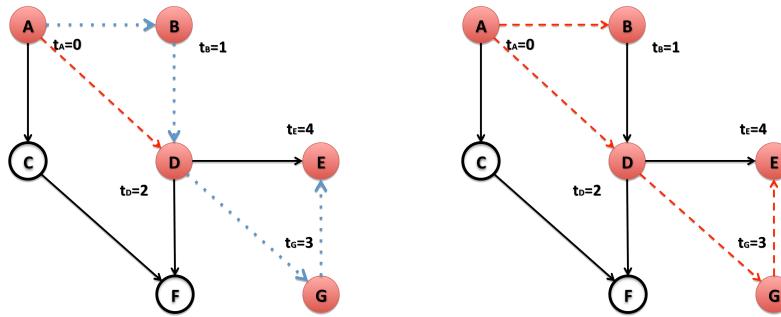
گراف جهت دار ایستا^{۱۸} مانند $G(V, E)$ با $|V| = n$ گره و مجموعه یال های E را در نظر بگیرید. فرض می کنیم که از وجود همه رئوس آگاه هستیم و هیچ یالی را نمی شناسیم. بنابراین برای بازسازی شبکه، باید درایه های ماتریس مجاورت شبکه پیش بینی شوند. پس، در این مسئله به دنبال گرافی مانند G^* هستیم که $\epsilon \leq \|G - G^*\|$ که معيار فاصله ای مناسب بین دو گراف است. فرض اساسی ما، وجود یک فرایند خارجی بر روی گراف است که دارای شرایط زیر باشد:

- این فرایند به تعداد دلخواه بر روی شبکه انجام پذیر است.
- فرایند مورد نظر با هر بار اجرا به گره هایی که ملاقات می کند مقداری معین اختصاص داده و برای هر اجرای خود، یک خروجی عددی تولید می کند که تابعی از مقادیر اختصاص داده شده به گره هاست.

ممکن است فرایندهای زیادی دارای این ویژگی ها باشند؛ اما فرایندی که ما در این پژوهش در نظر گرفتیم، فرایند انتشار اطلاعات (مانند ویروس، شایعه و یا بیماری) است. خروجی این فرایند به ازای هر گره، برچسب زمانی آلوده شدن آن گره است. برای مثال در یک شبکه از وب سایت ها، انتشار یک خبر میان وب سایت ها، نوعی انتشار و زمان پوشش آن خبر توسط هر وب سایت، زمان آلوده شدن آن وب سایت (گره شبکه) است. هر بار اجرای فرایند انتشار، یک آبشار اطلاعاتی^{۱۹} در شبکه به وجود می آورد. شکل ۱.۲ (آ) این فرایند را بر روی یک شبکه نمونه نمایش می دهد. هدف ما این است که از داده هایی که فرایند انتشار تولید می کند، استفاده کرده و لینک های شبکه را استنتاج کنیم. به وضوح فرایند انتشار اخبار، ویژگی اول فرایند خارجی مورد نظر ما را برآورده می کند. برای تشریح ویژگی دوم، در بخش بعدی پیش زمینه ای از شبکه های انتشار بیان شده و سپس ویژگی دوم برای فرایند انتشار شرح داده خواهد شد.

^{۱۸} مقصود از ایستا، عدم اضافه شدن یال یا گره جدید است.

Information cascade^{۱۹}



(ب) محتمل‌ترین درخت سازگار با آبشر انتشاریافته (مشخص شده با یال‌های نقطه‌ای)

(آ) یک آبشر انتشاریافته بر روی شبکه نمونه (مشخص شده با یال‌های متمایز)

شکل ۱.۲: نمونه‌ای از انتشار اطلاعات و درخت متناظر با آبشر ایجاد شده

۲۰۳۰۲ شبکه‌های اطلاعاتی و رفتار آبشری

در هر مرتبه از به وقوع پیوستن انتشار در یک شبکه اطلاعاتی، یک آبشر اطلاعاتی ایجاد می‌شود. به عبارتی دیگر، یک آبشر اطلاعاتی، یک دنباله از گره‌های انتشار است که توسط فرایند انتشار ملاقات شده‌اند. به طور دقیق‌تر، یک آبشر c مجموعه‌ای از سه‌تایی‌های $(u, v, t_v)_c$ است، به این معنی که آبشر c در زمان t_v از گره u به گره v می‌رسد [۴۱]. برای گره آغازین مانند s ، سه‌تایی $(\phi, s, t_s)_c$ را خواهیم داشت. ما فرض می‌کنیم تنها داده‌ای که می‌توان از فرایند انتشار به دست آورد، زمان رسیدن آبشر به هر گره است که آن را زمان برخورد t^* می‌نامیم. بنابراین تا این مرحله اختصاص مقادیر معین به هر گره تشریح می‌شود. در ادامه، آبشرهای منتشر شده بر روی شبکه، تحت مدل آبشری مستقل [۴۲] در نظر گرفته می‌شوند [۴۲]، بدین معنی که آلدود شدن یک گره و ثبت زمان آلدودگی آن، تنها به اولین همسایه آلدود کننده وابسته است و گره‌های آلدود کننده بعدی تاثیری در آلدود کردن گره مورد نظر نخواهند داشت. با در نظر گرفتن چنین ساختاری برای آبشرها، یک آبشر اطلاعاتی معرف یک درخت به عنوان زیرگرافی از گراف شبکه خواهد بود.

۳۰۳۰۲ محتمل‌ترین درخت آبشر

از آنجایی که ما فقط از زمان آلدودگی گره‌ها آگاه هستیم، ساختار درخت انتشار نمی‌تواند به صورت معین مشخص شود. حال قصد داریم یک تقریب برای شناسایی چنین درختی ارائه دهیم. برای این امر، یک مدل احتمالاتی به آبشرها نسبت داده و سعی می‌کنیم درختی که بیشترین احتمال وقوع را

Hit time^{۲۰}Independent cascade model^{۲۱}

دارد را به عنوان درخت انتشار تعیین کنیم.

احتمال $P_c(u, v)$ را به عنوان احتمال شرطی ملاقات راس v از u به شرط وقوع این ملاقات در آبشار c را در نظر می‌گیریم. به طور شهودی، احتمال انتشار یک خبر از دو گرهی که با فاصله زمانی زیادی از هم آلوده شده‌اند کمتر از زمانی است که این فاصله زمانی کم باشد. ما فرض می‌کنیم این احتمال تحت مدل نمایی تعریف شده باشد [۴۳، ۴۴]؛ به عبارت دیگر:

$$P_c(u, v) = P_c(\Delta_{u,v}) = e^{-\frac{\Delta_{u,v}}{\alpha}} \quad (6.2)$$

که $\Delta_{u,v} = t_v - t_u$ و α پارامتر تنظیم و نرمال‌سازی است. این احتمال می‌تواند با پارامترهای دیگری از فرایند انتشار نیز ترکیب شود. به عنوان مثال، تعداد گره‌های مابین دو گره نسبتی عکس با احتمال انتشار خواهد داشت. بدین معنی که هر چه تعداد گره‌های میانی در ترتیب انتشار مابین دو گره u و v بیشتر باشد، احتمال انتشار مستقیم از طریق یال (u, v) کمتر است.

یک آبشار با احتمال $\beta - 1$ متوقف می‌شود. بنابراین می‌توان درستنماهی آبشار c بر روی درخت T را بر این اساس تعریف نمود:

$$P(c|T) = \prod_{u,v \in E_T} \beta P_c(u, v) \prod_{w,x \in V_T, (w,x) \in E \setminus E_T} (1 - \beta) \quad (7.2)$$

که $T = (V_T, E_T)$ گراف درخت مورد نظر است.

همچنین، احتمال انتشار آبشار c بر روی گراف G به صورت زیر تعریف می‌شود:

$$P(c|G) = \sum_{T \in \tau(G)} P(c|T)P(T|G) \quad (8.2)$$

که $\tau(G)$ مجموعه تمام زیر گراف‌های القایی گراف G تحت رئوس ملاقات شده در آبشار c است. از آنجایی که تعداد درخت‌های القایی یک آبشار بسیار زیاد است، محاسبه (۸.۲) ناممکن خواهد بود. از آن جا که هدف اختصاص مقداری خروجی برای آبشار است به نحوی که در تشخیص یال‌های گراف زیرین موثر باشد، ما احتمال محتمل‌ترین درخت ^{۲۲} را به عنوان خروجی فرایند انتشار و احتمال انتشار آبشار بر روی گراف در نظر می‌گیریم.

$$P(c|G) \approx P(c|T_c^*) \quad (9.2)$$

منظور از T_c^* یا محتمل‌ترین درخت، درختی است که احتمال $P(c|T)$ را ماکزیمم می‌کند. به وضوح یال‌های متناظر با چنین درختی، محتمل‌ترین یال‌ها برای حضور در گراف اصلی نیز خواهند بود. شکل ۱.۲ (ب) یک نمونه از این درخت را نمایش می‌دهد. از آنجایی که مراحل محاسبه احتمال صرفاً

نوعی وزن دهنده به یال‌های محتمل موجود در شبکه است، استخراج درخت با بیشترین احتمال می‌تواند با الگوریتم‌های یافتن درخت پوشای بیشینه^{۲۳} [۴۵] انجام شود.

۴.۲ روش پیشنهادی: CS-NetRec

در روش پیشنهادی برای بازسازی شبکه، چندین آبشار بر روی شبکه تحت مدل آبشاری مستقل اجرا شده و با استفاده از مقادیر اختصاص داده شده به رئوس و محاسبه احتمال‌های محتمل‌ترین درخت‌ها، شبکه مورد نظر بازسازی می‌شود. برای این کار، یک سیستم خطی بر اساس داده‌های احتمالاتی تشکیل شده و پیدا کردن پاسخ این سیستم خطی، به عنوان بازسازی از شبکه مورد نظر برگردانده می‌شود. با فرض این که T_c^* محتمل‌ترین درخت مربوط به آبشار c باشد، با لگاریتم‌گیری از رابطه (۷.۲)

خواهیم داشت:

$$LP(c|T_c^*) = vLP(c|T_c^*)^T \cdot vec(Adj(G)) \quad (10.2)$$

که در آن $LP(c|T) = \log P(c|T)$ نمایش برداری ماتریس مجاورت گراف G (با هر ترتیب دلخواه) است. همچنین $vLP(c|T_c^*)$ نیز بردار احتمال‌های انتشار بر روی یال‌های محتمل است، به طوری که k -امین مولفه این بردار برابر است با:

$$vLP(c|T_c^*)_k = LP_c(i, j) = \log P_c(i, j) \quad (11.2)$$

با چنین نمایشی، ما قادر هستیم که هر اجرای فرایند انتشار را به صورت ضرب داخلی بردار احتمال‌های انتقال و بردار متناظر با ماتریس مجاورت نمایش دهیم. پس می‌توان اجرای یک دسته از آبشارها را به صورت یک سیستم خطی نمایش داد. نمونه‌ای از این سیستم خطی در شکل ۴.۲ نشان داده شده است. یافتن پاسخ این سیستم خطی برابر با ماکزیمم‌سازی درستنمایی توام همه آبشارها خواهد بود، چرا که برای هر آبشار، درخت با بیشترین احتمال درنظر گرفته خواهد شد. از آنجایی که پاسخ این سیستم خطی تنک است، حل مسئله (۵.۲) می‌تواند جواب مطلوبی برای این سیستم باشد. روش پیشنهادی در الگوریتم ۲.۴.۱ قابل مشاهده است.

$$\begin{bmatrix} LP(c_1|G) \\ LP(c_2|G) \\ LP(c_3|G) \\ \vdots \\ LP(c_m|G) \end{bmatrix} = \begin{bmatrix} LP_{c_1}(v_1, v_2) & \dots & LP_{c_1}(v_i, v_j) & \dots & LP_{c_1}(v_n, v_{n-1}) \\ LP_{c_2}(v_1, v_2) & \dots & LP_{c_2}(v_i, v_j) & \dots & LP_{c_2}(v_n, v_{n-1}) \\ LP_{c_3}(v_1, v_2) & \dots & LP_{c_3}(v_i, v_j) & \dots & LP_{c_3}(v_n, v_{n-1}) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ LP_{c_m}(v_1, v_2) & \dots & LP_{c_m}(v_i, v_j) & \dots & LP_{c_m}(v_n, v_{n-1}) \end{bmatrix} \begin{bmatrix} v_{1,2} \\ \vdots \\ v_{i,j} \\ \vdots \\ v_{n,n-1} \end{bmatrix}$$

شکل ۲.۲: سیستم خطی متناظر با اجرای m آبشار**Algorithm 2.4.1** CS-NetRec(C, V)**for** each $c \in C$ **do** $T_c^* \leftarrow$ The Most Probable Tree for c .**for** each $(i, j) \in V \times V$ **do**

$$LP_c(i, j) \leftarrow \begin{cases} \log P_c(i, j) & (i, j) \in T_c^* \\ 0 & o.w. \end{cases}$$

end forCalculate $LP(c|G) = \log P(c|G)$.Form the vector $vLP(c|T_c^*)$ from $LP_c(i, j)$ s.Add $LP(c|G)$ as a row to y .Add $vLP(c|T_c^*)$ as a row to A .**end for**

$x^* \leftarrow \arg \min_x \|x\|_1 + \|Ax - y\|_2^2$

return x^*

۵.۲ نتایج شبیه‌سازی

در این بخش، نتایج شبیه‌سازی بر روی روش پیشنهادی CS-NetRec ارائه می‌شود. پس از معرفی مجموعه داده‌ها و تنظیمات شبیه‌سازی، به معیارهای ارزیابی اشاره می‌کنیم و سپس نتایج را تفسیر خواهیم کرد.

۱۰.۵.۲ مجموعه داده‌ها و تنظیمات شبیه‌سازی

در این بخش تعدادی مجموعه داده مصنوعی و واقعی برای مراحل شبیه‌سازی در نظر گرفته شده‌اند. از سه مدل معروف ER [۴۶] (۱۰۰ گره و ۵۰۰ یال) ، Barabasi-Albert [۴۷] (۱۰۰ گره، ۵ گره اضافه شونده به هر گره) ، Small World [۴۸] (۱۰۰ گره، احتمال حذف-اضافه ۰/۴) و Kronecker Core-Priphery Football US-Top-۵۰۰ [۴۹] (۲۵۶ گره، ۶۰۰ یال به طور میانگین) برای تولید مجموعه داده‌های مصنوعی استفاده شده است. همچنین در مجموعه داده‌های واقعی از گراف شبکه C.elegans [۵۰] با ۳۰۶ گره و ۲,۳۴۵ یال و NetInf [۵۱] با ۱۱۵ گره و ۶۱۵ یال استفاده شده است.

در هر مورد از شبیه‌سازی‌ها، ۱۰۰ مرتبه آبشار تولید شده است. همچنین برای تولید آبشار، روند مشخص در کد روش NetInf [۴۱] استفاده شده است. این روند در واقع همان روش استفاده شده برای مدل‌سازی آبشارها که در بخش پیشین معرفی شد، می‌باشد.

$$\text{برای ارزیابی روش، می‌توان معیارهای دقت } \text{۲۴} \text{ و فراخوانی } \text{۲۵} \text{ را به کار برد که عبارتند از:} \\ \frac{\text{تعداد لینک‌های پیش‌بینی شده درست}}{\text{تعداد کل لینک‌های پیش‌بینی شده}} = \text{دقت} \quad (۱۲.۲)$$

$$\frac{\text{تعداد لینک‌های پیش‌بینی شده درست}}{\text{تعداد کل لینک‌های شبکه انتشار واقعی}} = \text{فراخوانی} \quad (۱۳.۲)$$

دو معیار دقت و فراخوانی به نوعی تاثیر عکس بر روی هم دارند. بدین معنی که فراخوانی بالا احتمال کمتر بودن دقت را افزایش داده و فراخوانی پایین، دقت را افزایش می‌دهد. برای خودداری از این مصالحه میان دقت و فراخوانی از معیار F^{۲۶} استفاده شده است، که در واقع یک میانگین موزون

را بین دقت و فراخوانی بیان می‌کند:

$$F = \frac{\text{دقت} \times \text{فراخوانی}}{\text{فراخوانی} + \text{دقت}} \quad (۱۴.۲)$$

Precision^{۲۴}

Recall^{۲۵}

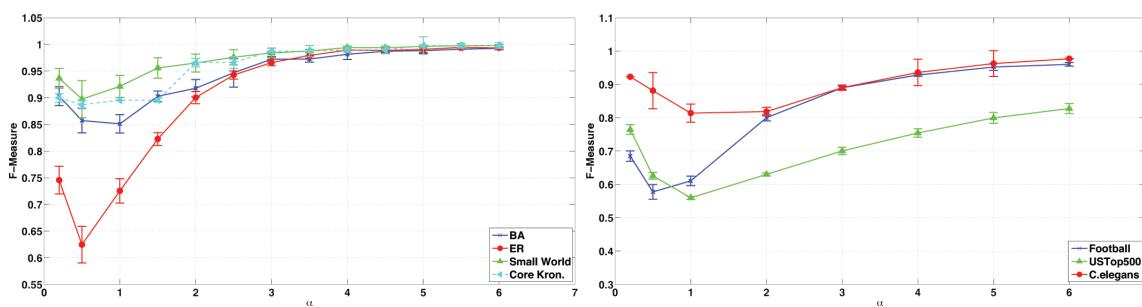
F-measure^{۲۶}

Harmonic mean^{۲۷}

۲.۵.۲ تحلیل پارامترها

ابدا، رفتار روش نسبت به پارامترهای موجود سنجیده شده تا در صورت امکان بتوان مقداری را برای آنها برگزید. پارامتر نخست، α است که در واقع میزان سرعت آبشار را بین گرهها تعیین می‌کند. برای تحلیل این پارامتر، مقدار پارامتر β برابر $5/0$ قرار داده شد. همان‌گونه که در شکل ۲.۵.۲ نشان داده شده است، به نظر می‌رسد برای همه مجموعه داده‌ها مقداری پیرامون 3 ، جایی است که معیار F ایستا می‌شود.

پارامتر دوم که تنظیم کننده طول آبشارها است، β می‌باشد. به وضوح مقدار کم برای β باعث به وجود آمدن آبشارهای کوتاه‌تر خواهد شد. پس در اجرای چند آبشار، اطلاعات کمتری از شبکه بدست خواهد آمد و بدینهی است که تعدادی از یال‌ها نشوند و فراخوانی نهایی کوچک باشد. با توجه به شکل ۲.۵.۲ و ایستا شدن معیار F، مقدار $5/0$ برای این پارامتر برگزیده شد.

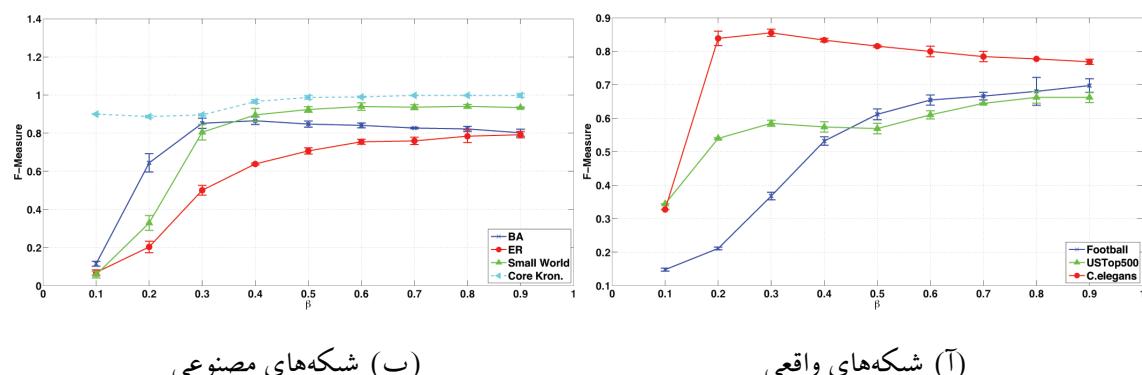


(ب) شبکه‌های مصنوعی

(آ) شبکه‌های واقعی

شکل ۳.۲: تحلیل پارامتر α در روش CS-NetRec

با توجه به ایستا شدن نمودارها، مقدار 3 برای پارامتر α انتخاب شد.



(ب) شبکه‌های مصنوعی

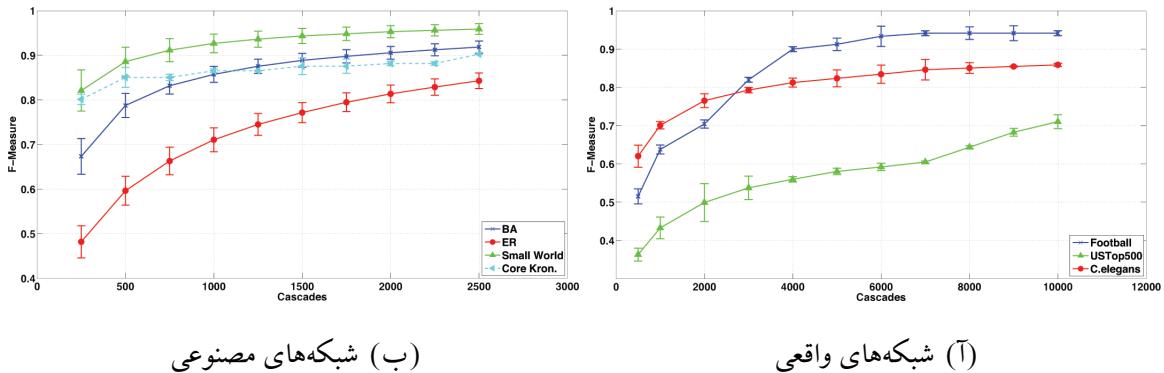
(آ) شبکه‌های واقعی

شکل ۴.۲: تحلیل پارامتر β در روش CS-NetRec

با توجه به ایستا شدن نمودارها، مقدار $5/0$ برای پارامتر β انتخاب شد.

۳.۵.۲ وابستگی به تعداد آبشار

با اجرای تعداد آبشار بیشتر، یالهای بیشتری از گراف دیده خواهد شد. به علاوه، احتمال انتقال یالهای مختلف در دستگاه خطی بیشتر تکرار خواهد شد. بنابراین تاثیر آنها در احتمال آبشار بیشتر شده و با احتمال بالاتری بازسازی خواهد شد. در این آزمایش میزان وابستگی دقت روش به تعداد آبشارها بررسی شد. در شکل ۳.۵.۲ نتایج این آزمایش نمایش داده شده است. برای شبکه‌های مصنوعی، تعداد آبشارها از ۲۵۰ تا ۲۵۰۰ و برای شبکه‌های واقعی از ۲۵۰ تا ۱۰۰۰۰ تغییر یافته‌اند. نتایج نمایان‌گر عملکرد مطلوب روش ارائه شده حتی در حضور تعداد اندازه‌گیری (آبشار)‌های پایین است. در شبکه‌های مصنوعی و واقعی، می‌توان مشاهده کرد که با تعداد آبشار نزدیک به یک چهارم یالهای ممکن، روش ارائه شده توانسته معیار F بالای ۰/۸ کسب کند.

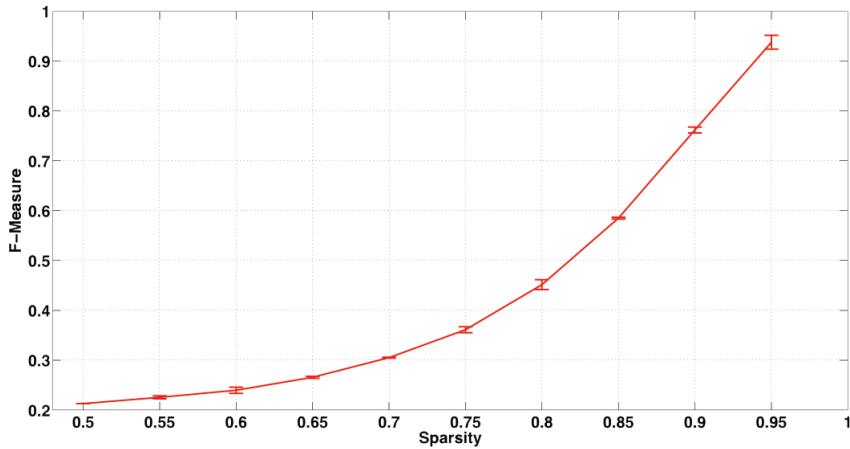


شکل ۳.۵.۲: وابستگی به تعداد آبشار در روش CS-NetRec

نتایج نمایان‌گر عملکرد مطلوب روش ارائه شده حتی در حضور تعداد اندازه‌گیری (آبشار)‌های پایین است. در شبکه‌های مصنوعی و واقعی، می‌توان مشاهده کرد که با تعداد آبشار نزدیک به یک چهارم یالهای ممکن، روش ارائه شده توانسته معیار F بالای ۰/۸ کسب کند.

۴.۵.۲ وابستگی به میزان تنک بودن شبکه

طبق تئوری نمونهبرداری فشرده، هر چه بردار مورد نظر کمتر تنک باشد، اندازه‌گیری‌های بیشتری برای بازیابی آن نیاز است. برای بررسی این پدیده و تاثیر آن بر دقت روش، تعدادی از شبکه‌های مصنوعی بر اساس مدل ER در نظر گرفته شد که پارامتر راس آنها ثابت (۱۰۰ گره) بوده و پارامتر تعداد یال آنها را از ۵۰۰ تا ۵۰۰۰ یال جهت‌دار تغییر دادیم. به عبارت دیگر میزان تنک بودن را از ۰/۵ تا ۰/۹۵ تغییر دادیم. همچنین تعداد آبشارها را برابر روی هر شبکه، ۳۰۰۰ آبشار در نظر گرفتیم.



شکل ۶.۲: وابستگی به میزان تنک بودن شبکه در روش CS-NetRec

هرچه میزان تنک بودن بیشتر باشد، بازسازی بهتری از شبکه بدست می‌آید و برای مقادیر پایین‌تر، اندازه‌گیری‌های بیشتری نیاز است.

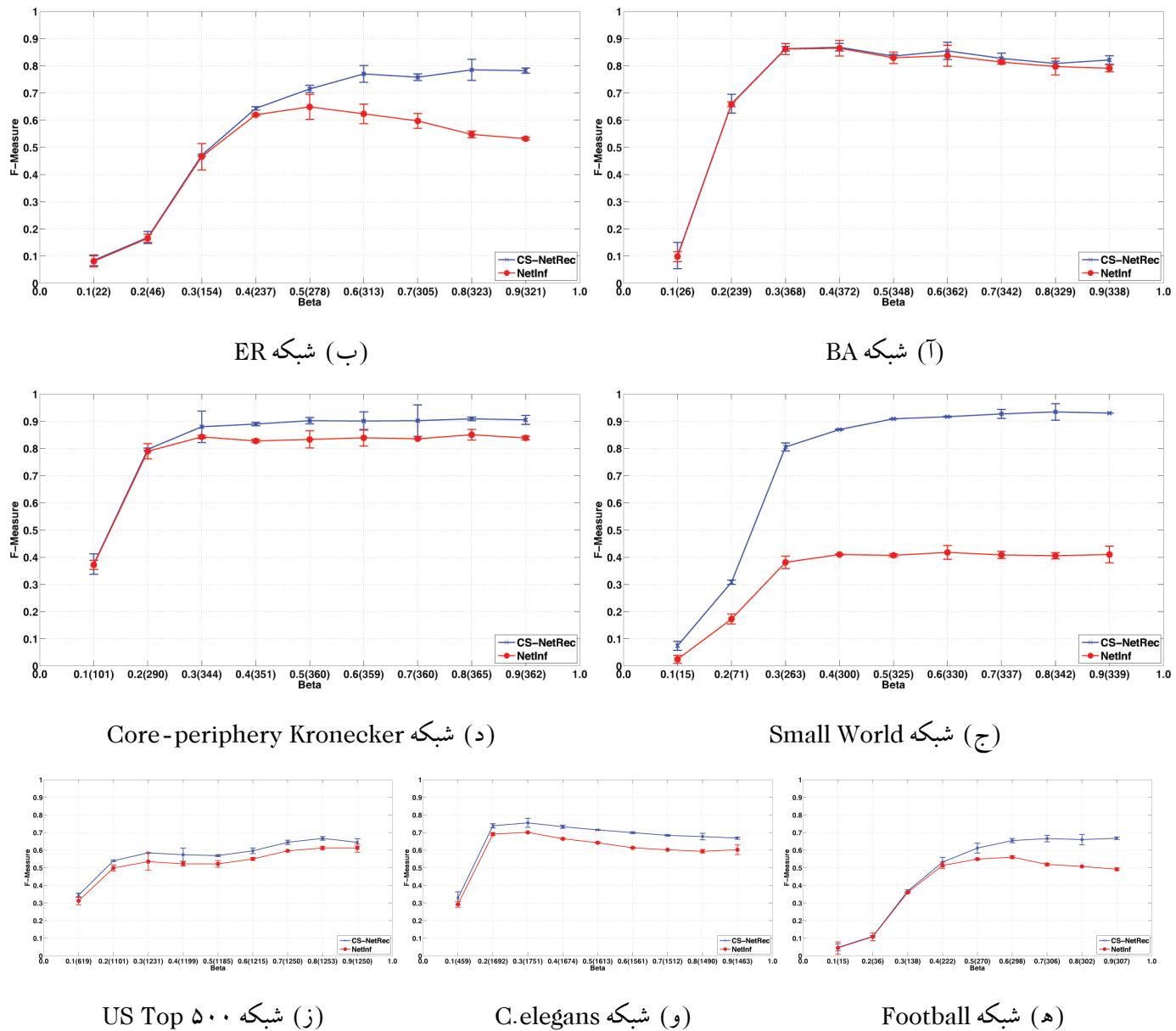
همان‌گونه که در شکل ۴.۵.۲ نمایش داده شده است، هرچه میزان تنک بودن بیشتر باشد، بازسازی بهتری از شبکه بدست می‌آید و برای مقادیر پایین‌تر، اندازه‌گیری‌های بیشتری نیاز است.

۵.۵.۲ مقایسه با روش NetInf

در این بخش، روش ارائه شده با روش NetInf که یکی از روش‌های ارائه شده اخیر در استخراج شبکه انتشار است، مقایسه می‌شود.

برای مقایسه از کد خود نویسنده‌گان استفاده شد. همچنین مقایسه به ازای پارامترهای β متفاوت انجام شد تا عملکرد هر دو روش در حضور اطلاعات محلی‌تر (آبشارهای کوتاه‌تر) و اطلاعات عمومی‌تر (آبشارهای بلند‌تر) نمایش داده شود. همچنین مقدار پارامتر α برابر ۱ درنظر گرفته شد. برای تعداد ثابتی آبشار، روش NetInf به تعداد یال‌هایی که روش ما می‌تواند بازسازی کند، اجرا شد. نتایج در شکل ۷.۲ نمایش داده شده است. در این شکل، اعداد نمایش داده شده در پرانتز که کنار مقادیر مختلف محور β قرار دارند، تعداد یال‌های بازسازی شده را نشان می‌دهند. مشاهده می‌شود که تقریباً در همه شبکه‌ها می‌توان بهبود عملکرد را در روش ارائه شده دید. این بهبود در طول‌های بیشتر آبشارها به مرتب بیشتر و در حدود ۱۰ درصد است. به عنوان دلیل این امر می‌توان به نیاز به تعداد تکرار بیشتر در روش NetInf اشاره کرد. پیچیدگی روش NetInf چه در زمان و چه در فضا می‌تواند کمتر از روش ارائه شده باشد. روش NetInf در مرتبه تعداد یال‌های دیده شده در آبشارها (که لزوماً یال‌های واقعی نیستند) عمل می‌کند اما روش ارائه شده در مرتبه تعداد یال‌های ممکن یعنی $O(n^2)$

کار می‌کند که n تعداد رئوس گراف است. درصورتی که تعداد مشاهدات بالا باشد، پیچیدگی روش NetInf نیز به همان نسبت بالاتر خواهد بود و پاسخ مورد نظر در زمان بیشتری بدست خواهد آمد. هر چند آزمایش‌ها دقت بالاتر روش ارائه شده را نمایش می‌دهد.



شکل ۷.۲: مقایسه روش CS-NetRec با روش NetInf

تقریباً در همه شبکه‌ها می‌توان بهبود عملکرد را در روش ارائه شده دید. این بهبود در طول‌های بیشتر آثارها به مرتب بیشتر و در حدود ۱۰ درصد است.

فصل ۳

پیش‌بینی لینک مقیاس پذیر با استفاده از فاکتورگیری ماتریس

۱۰.۳ مقدمه و مروری بر کارهای پیشین

پیش‌بینی لینک از جمله مسائل پایه‌ای در مطالعه همه انواع شبکه‌های پیچیده است و روش‌های ارائه شده برای آن، می‌تواند زمینه‌های تحقیقاتی بسیاری را شامل شود. برای مثال، مطالعه سیستم‌های پیشنهاددهنده از جمله موارد کاربرد برای این مسئله است؛ چه سیستم زیرین یک شبکه باشد (مانند شبکه‌های اجتماعی برخط) و چه این سیستم بتواند به یک سیستم شبکه‌ای نگاشت شود (مانند سیستم‌های کاربر-کالا که به صورت گراف‌های دوبخشی نگاشت می‌شوند). همان‌طور که در فصل اول ذکر شد، وجود داده ناکامل در سیستم‌های بزرگ نیز انگیزه دیگری برای ارائه راه حل‌های کارای این مسئله است.

روش‌های ارائه شده برای مسئله پیش‌بینی لینک به طور کلی به دو دسته تقسیم می‌شوند [۵۲] :

۱. پیش‌بینی لینک ساختاری^۱ که در آن هدف تشخیص وجود یا عدم وجود لینک در بخش‌های غیر مشاهده شده گراف شبکه است.

۲. پیش‌بینی لینک زمانی^۲ که در آن، ورودی مسئله تعدادی تصویر لحظه‌ای از شبکه مورد نظر در برچسب‌های زمانی گوناگون است و هدف پیش‌بینی اضافه یا کم شدن لینک‌های شبکه در

Structural Link Prediction^۱

Temporal Link Prediction^۲

برچسب زمانی آینده است.

در این بخش ما پیش‌بینی لینک ساختاری را مدنظر قرار می‌دهیم و ترکیب "پیش‌بینی لینک" را به عنوان معرف این مسئله به کار می‌بریم. هر چند ایده‌های مطرح شده در هر کدام از دسته‌های فوق می‌تواند در دیگری نیز به کار گرفته شود. یک مثال ساده، زمانی است که تنها یک تصویر لحظه‌ای از شبکه مورد نظر داشته باشیم.

یکی از اساسی‌ترین چالش‌ها در ارائه راه حلی کارا برای پیش‌بینی لینک، مقیاس‌پذیری روش ارائه شده و قابلیت آن برای اجرا در شبکه‌های بزرگ است. همزمان با این مسئله، بنا به آن‌چه در فصل اول نشان داده شد، داده شبکه‌ای بسیار تنک است؛ بدین معنا که تعداد لینک‌های موجود نسبت به تمامی لینک‌های ممکن بسیار کم است. در بسیاری از روش‌های یادگیری ماشین و پردازش سیگنال، مانند آن‌چه در فصل قبل ارائه شد، نشان داده شده است که می‌توان از ویژگی تنک بودن برای ارائه پاسخ‌های مناسب استفاده کرد.

۱۰.۳ مقدمه‌ای بر فاکتورگیری و تکمیل ماتریس

از آنجا که پیش‌بینی لینک در شبکه‌ها به نوعی می‌تواند موردی خاص از مسئله تکمیل ماتریس شناخته شود، در این بخش مقدمه کوتاهی بر تکمیل ماتریس ارائه خواهیم داد.

تکمیل ماتریس تعییمی از نمونه‌برداری فشرده می‌تواند تلقی شود. بدین معنی که در این مسئله سیگنال مورد نظر دو بعدی بوده و برخلاف فرض تنک بودن در نمونه‌برداری فشرده، عمدتاً با فرض رتبه پایین بودن ماتریس فرموله می‌شود. مسئله تکمیل ماتریس، بدین صورت مطرح می‌شود: یک ماتریس دلخواه در اختیار داریم که تعدادی از درایه‌های آن معلوم و بقیه نامعلوم هستند. آیا می‌توان ماتریس داده شده را تکمیل کرد؟

در حالت عمومی این مسئله قابل حل نیست. چرا که هر یک از درایه‌های ماتریس می‌تواند به دلخواه مقداری داشته باشد و پیش‌بینی این مقادیر ممکن نیست. بنابراین یک فرض یا دانش پیشین در مورد ویژگی‌ها و یا ساختار ماتریس مورد نظر لازم است. فرضی که معمولاً در مطالعات این حوزه درنظر گرفته می‌شود، رتبه پایین بودن ماتریس است. این فرض موجب می‌شود که همه درایه‌های ماتریس به دلخواه تغییر نکنند. برای درک بهتر این موضوع می‌توان درجه آزادی یک ماتریس تصادفی (ماتریسی که هر درایه آن عددی تصادفی است) را با فرض رتبه پایین بودن تحلیل کرد. یک ماتریس تصادفی با ابعاد $n \times n$ دارای درجه آزادی n^2 است. اما اگر این ماتریس دارای رتبه r باشد، می‌توان با

استفاده از تجزیه مقادیر تکین^۳ دریافت که درجه آزادی چنین ماتریسی برابر با $r^2 - nr$ است [۵۳]. بنابراین مرتبه عناصری که به صورت آزادانه می‌توانند مقداری را اختیار کنند، $O(nr)$ به جای ، برابر با $O(n^2)$ خواهد بود .

مسئله تکمیل ماتریس به گونه‌ای که رتبه ماتریس حاصل پایین باشد، می‌تواند به صورت مسئله بهینه‌سازی زیر در نظر گرفته شود:

$$\min_X \text{rank}(X) \quad s.t. \quad X_{ij} = M_{ij} \quad (i, j) \in \Omega \quad (1.3)$$

که در آن، X پاسخ مسئله و M_{ij} درایه‌های ماتریس اولیه و Ω مجموعه درایه‌های معلوم ماتریس M است. مسئله فوق، یک مسئله غیرچندجمله‌ای سخت است که این ناشی از استفاده از تابع رتبه در مسئله بهینه‌سازی است. برای پیدا کردن پاسخی نزدیک به پاسخ مسئله فوق، از تابع نرم هسته‌ای^۴ به جای تابع رتبه استفاده می‌شود و ثابت می‌شود تحت شرایطی پاسخ هر دو مسئله یکی است [۵۳].

بنابراین مسئله (۱.۳) تبدیل به مسئله زیر می‌شود:

$$\min_X \|X\|_* \quad s.t. \quad X_{ij} = M_{ij} \quad (i, j) \in \Omega \quad (2.3)$$

که در آن $\|X\|_*$ نرم هسته‌ای ماتریس X بوده و برابر است با مجموع مقادیر تکین^۵:

$$\|X\|_* = \sum_k \sigma_k(X) \quad (3.3)$$

که امین بزرگترین مقدار تکین ماتریس X است.

مسئله (۲.۳) یک مسئله بهینه‌سازی محدب بوده و روش‌های بسیاری برای حل آن مانند روش‌های برنامه‌ریزی نیمه‌معین^۶ وجود دارد.

روشی که در این فصل ارائه خواهیم داد، ریشه در روش‌های فیلترینگ اشتراکی دارد [۵۴]. به طور خلاصه، فیلترینگ اشتراکی، مطالعه تعاملات میان کاربران و کالاهای خریداری شده و یا رتبه‌بندی شده است و هدف در آن، توسعه روش‌هایی برای پیش‌بینی تعاملات و یا به عبارت ساده‌تر، رتبه‌بندی‌های آینده کاربران به کالاهای دیگر است که این امر فقط بر اساس رفتار گذشته کاربران انجام می‌شود. تعاملات میان کاربران و کالاهای معمولاً در قالب یک ماتریس که سطرها در آن نماینده کاربران و ستون‌ها نماینده کالاهای و درایه‌ها نمایانگر رتبه کاربران به کالاهای هستند، نمایش داده می‌شوند. روش

Singular Value Decomposition (SVD)^۳

Nuclear norm^۴

Singular values^۵

Semi-definite Programming^۶

غالب در این زمینه، بر اساس مدل‌های ویژگی‌های پنهان^۷ است. در این دسته از روش‌ها، هدف استنتاج تعدادی ویژگی پنهان از تعاملات کاربران و کالاها و سپس پیش‌بینی تعاملات آینده بر اساس میزان شbahت این ویژگی‌ها به هم است. موفق‌ترین پیاده‌سازی از مدل‌های ویژگی پنهان، هم از لحاظ دقت و هم از لحاظ مقیاس‌پذیری، در چارچوب فاکتورگیری ماتریس^۸ تعریف شده است. فاکتورگیری از ماتریس در دسته روش‌های کاهش بعد در یادگیری ماشین قرار دارد [۲]. در فاکتورگیری ماتریس، هدف یافتن تقریبی مناسب از یک ماتریس هدف است به طوری که تابع خطابی^۹ خاص را کمینه کند.

پژوهش‌های جدی در ارتباط با فاکتورگیری ماتریس و فیلترینگ اشتراکی از مسئله نت‌فلیکس^{۱۰} مطرح شد [۵۵]. کمپانی نت‌فلیکس برای ارتقای سیستم پیشنهاد فیلم‌های خود، مسئله‌ای را تدوین کرد که در آن، ماتریسی که سطراها در آن نماینده کاربران و ستون‌ها نماینده فیلم‌ها بودند، داده شده بود. درایه‌های این ماتریس رتبه هر کاربر به هر فیلم بود. هدف پیش‌بینی مقادیر رتبه کاربران به بقیه فیلم‌ها و در نهایت پیشنهاد این فیلم‌ها به کاربران بود، به طوری که روش ارائه شده در مجموع ۱۰ درصد بهبود دقت را به همراه داشته باشد. فرضی طبیعی که بر روی این ماتریس شده، این است که فاکتورهای سلیقه‌ای کمی در انتخاب و پیشنهاد فیلم به کاربران تأثیرگذار است. بنابراین می‌توان ماتریس نت‌فلیکس را ماتریسی رتبه پایین در نظر گرفت. به طور کلی در فیلترینگ اشتراکی، این فرض انجام می‌شود که ماتریس کاربران و کالاها یک ماتریس رتبه پایین است. به طور دقیق‌تر، فرض اساسی این است که اگر ماتریس رتبه‌بندی‌ها R باشد (با m کاربر و n کالا)، این ماتریس را می‌توان به صورت ضرب دو ماتریس با رتبه^{۱۱} k تقریب زد به طوری که:

$$R_{m \times n} \approx U_{k \times m}^T V_{k \times n} \quad (4.3)$$

که در آن U و V به ترتیب ماتریس ویژگی‌های پنهان برای کاربران و کالاها هستند. شکل ۳ نمونه‌ای از این فضای ویژگی پنهان مشترک بین کاربران و فیلم‌ها را نشان می‌دهد. بنابراین می‌توان گفت که رتبه کاربرⁱ به کالای^j با ضرب داخلی بردار ویژگی پنهان کاربرⁱ و بردار ویژگی پنهان کالای^j تقریب زده شده است:

$$R_{i,j} \approx U_i^T V_j \quad (5.3)$$

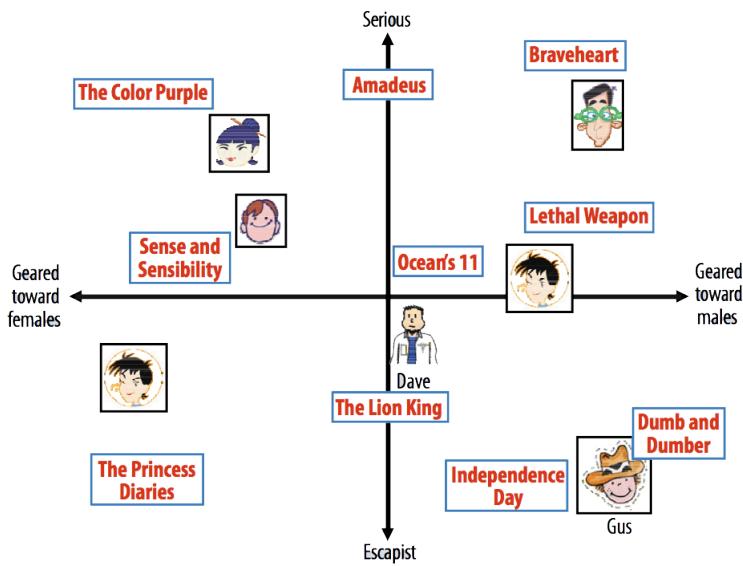
Latent Feature Models^۷

Matrix Factorization^۸

Loss function^۹

Netflix^{۱۰}

Rank^{۱۱}



شکل ۱.۳: تصویری از یک فضای ویژگی پنهان مشترک میان کاربران و فیلم‌ها [۵۴] این فضا هم کاربران و هم فیلم‌ها را به دو زیرگروه مخاطب مرد و زن و زیرگروه جدی و غیرجدی تقسیم‌بندی می‌کند.

به عبارتی دیگر، در فاکتورگیری ماتریس، هدف یافتن ماتریس‌های U و V است به طوری که:

$$(U^*, V^*) = \arg \min_{(U, V)} l(R, L(U^T V)) \quad (6.3)$$

که در آن (\cdot, \cdot) l و (\cdot) L توابع خطا و پیوند^{۱۲} مناسب هستند. علاوه بر این دید، می‌توان مسئله را به صورت احتمالاتی نیز فرموله کرد. در واقع انتخاب تابع خطا و لینک در (۶.۳) بسته به فرض احتمالاتی بر روی ماتریس کاربران و کالاها است. در مرجع [۵۶] تابع احتمال شرطی زیر بر روی این ماتریس تعریف شده است:

$$p(R|U, V, \sigma^2) = \prod_{(i,j) \in O} N(R_{ij}|(U_i^T V_j, \sigma^2)) \quad (7.3)$$

که در آن $N(x|\mu, \sigma^2)$ تابع چگالی توزیع نرمال و O درایه‌های دیده شده است. به راحتی قابل اثبات است که این روش احتمالاتی با تابع خطا مجذور خطا معادل است. همچنین ممکن است توزیع احتمال پیشین بر روی پارامترها وجود داشته باشد و به دنبال آن استنتاج پارامترها^{۱۳} از روش‌هایی مانند ماکریم درستنمایی، ماکریم احتمال پسین، زنجیره مارکوف^{۱۴} و روش‌های مشابه انجام می‌شود. قیود متفاوتی ممکن است به مسئله (۶.۳) اعمال شود. یکی از رایج‌ترین قیدها، نامنفی بودن

Link function^{۱۲}

Inference^{۱۳}

Markov Chain Monte Carlo (MCMC)^{۱۴}

ماتریس‌های U و V است. این زمینه به فاکتورگیری نامنفی ماتریس^{۱۵} نیز شناخته می‌شود. مرجع [۲] بررسی جامعی از مطالعات انجام شده در این حوزه را انجام داده است. از این قید در روش ارائه شده در این فصل برای ساده‌سازی بهینه‌سازی نیز استفاده می‌شود، هر چند دلایلی مانند تفسیر بهتر ویژگی‌های پنهان و یا تنک شدن این ویژگی‌ها در [۲] نیز ذکر شده است.

در این فصل، یک رهیافت احتمالاتی در فاکتورگیری ماتریس دنبال شده است. به طور دقیق‌تر از یک مدل تولید‌کننده^{۱۶} پواسون استفاده شده و نشان داده شده استفاده از این مدل نه تنها باعث کاهش دقت در استنتاج پارامترها و فاکتورگیری ماتریس نخواهد شد، بلکه استفاده از این مدل باعث می‌شود استنتاج پارامتر با مرتبه‌ای خطی نسبت به تعداد رئوس گراف شبکه انجام شود. حال آن که در مدل‌های پیشین، مانند برنولی-لジستیک، این مرتبه از توان دوم تعداد رئوس گراف شبکه است. بنا بر دانش ما، تاکنون روشی از مدل پواسون برای پیش‌بینی لینک استفاده نکرده است. علاوه بر این، الگوریتم ارائه شده، می‌تواند با تغییراتی در زمینه‌های دیگر مانند فیلترینگ اشتراکی و کاهش بعد استفاده شود.

۲۰.۳ استفاده از فاکتورگیری و تکمیل ماتریس در تحلیل شبکه‌ها و پیش‌بینی لینک

پیش‌بینی لینک و یا به طور کلی تکمیل شبکه، وجود اشتراک زیادی با مسئله تکمیل ماتریس دارد. اما معمولاً در تکمیل ماتریس، درایه‌های ماتریس مورد نظر حقیقی فرض می‌شوند. همچنین فرض اساسی دیگر این روش‌ها این است که درایه‌های معلوم به صورت تصادفی و یکنواخت از ماتریس اولیه نمونه‌برداری شده‌اند. در حالی که چنین فرضی در شبکه‌ها نمی‌تواند دقیق باشد، چرا که توزیع درجه در گراف شبکه‌های واقعی معمولاً به شکل توانی^{۱۷} است.

در مرجع [۵۷] روشی برای تکمیل ماتریس تحت نمونه برداری توانی ارائه شده است. اما این تنها ویژگی ماتریس‌های شبکه‌ها نیست؛ بلکه ویژگی‌های دیگری چون تنک بودن ماتریس شبکه‌ها و همچنین دودویی بودن عناصر آنها در نظر گرفته نشده است. در مرجع [۵۸] با استفاده از تئوری تکمیل ماتریس، روشی برای استنتاج علامت یال‌ها در شبکه‌های علامت‌دار ارائه شده که بر پایه دو

فرض تعادل محلی و عمومی^{۱۸} استوار است. به زبانی ساده، تعادل در یک شبکه علامت‌دار، تقسیم شبکه به دو بخش با علامت مخالف هم در طی زمان است. این شبکه‌ها معمولاً برای مدل‌سازی اعتماد بین گره‌های شبکه به کار می‌روند. این در حالی است که فرض تعادل همواره در شبکه‌های دیگر برقرار نیست و البته شبکه‌های دیگر نمی‌توانند به صورت یک شبکه علامت‌دار مدل شوند. در [۵۹] از بهینه‌سازی محدب برای خوشبندی گراف استفاده شده که تا حدی می‌توان آن را به روش‌های تکمیل ماتریس و کمینه‌سازی نرم هسته‌ای مرتبط دانست. این روش تنها خوشبندی‌های جدای از هم را در نظر می‌گیرد و همچنین مشخص نشده آیا می‌توان این روش را در مقیاس‌های بزرگ به کار برد.

پیش‌بینی لینک می‌تواند به عنوان یک مسئله دسته‌بندی^{۱۹} دو کلاسه در نظر گرفته شود و روش‌های مرتبط به آن بناظر و بدون ناظر هستند. در روش‌های بدون ناظر، معمولاً یک کمیت نشان‌گر شbahت دو گره (مانند تعداد همسایگان مشترک) بین زوج‌های نامشخص محاسبه می‌شود و از روی مجموعه این مقادیر، لینک‌های آینده پیش‌بینی می‌شوند (برای مثال با مرتب سازی آن‌ها). بنابراین در این روش‌ها یادگیری انجام نمی‌شود [۵۲]. یکی از نخستین مطالعات در زمینه پیش‌بینی لینک در مرجع [۶۰] مطرح شده که در آن دسته‌ای از کمیت‌های بدون ناظر ارائه و مقایسه شده‌اند. بسیاری از این کمیت‌ها می‌توانند به سرعت بر روی شبکه محاسبه شوند، هر چند کارایی متغیر آن‌ها مانع از استفاده از آن‌ها می‌شود.

تمامی روش‌های دیگر در پیش‌بینی لینک بناظر هستند. مرجع [۵۲] دسته‌بندی کامل و جامعی از این روش‌ها آورده است:

• روش‌های بر پایه ویژگی:

در پیش‌بینی لینک بر اساس ویژگی، معمولاً رفتار شبکه با یادگیری پارامتری مانند θ که حاصل کمینه‌سازی مسئله‌ای مانند (۸.۳) است، تحلیل می‌شود:

$$\min_{\theta} \frac{1}{|O|} \sum_{(i,j) \in O} l(G_{ij}, \hat{G}_{ij}(\theta)) + \Omega(\theta) \quad (8.3)$$

که در آن $\hat{G}_{ij}(\theta)$ مقدار پیش‌بینی شده برای یال (j, i) بر حسب پارامتر θ بوده، $l(\cdot, \cdot)$ و $\Omega(\cdot)$ توابع خطا و پیوند و $\Omega(\cdot)$ یک جمله هموارسازی^{۲۰} برای جلوگیری از بیش‌برازش^{۲۱} مدل است. اگر برای هر گره گراف شبکه مانند n و برای هر یال مانند (j, i) به ترتیب یک بردار ویژگی x_i و z_{ij} داشته باشیم، تعریف \hat{G} بر اساس (۹.۳)، روش‌های بر حسب ویژگی را معرفی

Local and global balance^{۱۸}

Classification^{۱۹}

Regularization^{۲۰}

Over fitting^{۲۱}

خواهد نمود:

$$\hat{G}_{ij}(w, v) = L(f_E(z_{ij}; w) + f_V(x_i, x_j; v)) \quad (9.3)$$

که f_E و f_V بردارهای ویژگی یال‌ها و رئوس هستند و $L(\cdot)$ تابع پیوند مناسب. برای مثال در انتخاب خطی تابع پیوند، $w^T z_{ij} + v_1^T x_i + v_2^T x_j$ برای یال‌ها و $w^T z_{ij}$ برای رئوس را می‌توان در نظر گرفت.

• هموارسازی گراف: در این روش‌ها، با فرض وجود ویژگی‌های رأسی مانند x_i ، یک تابع هسته مانند $K(i, j)$ که نماینده میزان شباهت بین دو گره خاص است، تعریف می‌شود. سپس این فرض در نظر گرفته می‌شود که مقادیر ماتریس مجاورت بر حسب این هسته، به صورت نرم تغییر می‌کنند. این روش و روش‌های مشابه بیشتر در یادگیری نیمه‌نظرارتی^{۲۲} استفاده می‌شود و در قالب مسئله (۸.۳) به شکل زیر خواهد بود:

$$\Omega(\hat{G}) = \frac{\lambda}{2} \sum_{i,i',j,j'} K_{ii'jj'} (\hat{G}_{ij} - \hat{G}_{i'j'})^2 + \frac{\mu}{2} \sum_{(i,j) \notin O} \hat{G}_{ij}^2 \quad (10.3)$$

مدل بالا به عنوان انتشار لینک^{۲۳} شناخته می‌شود [۶۱]. کارایی این روش‌ها به انتخاب تابع هسته وابسته است.

• مدل‌های کلاس‌پنهان^{۲۴}: در این مدل‌ها، هر گره گراف به یک کلاس نسبت داده می‌شود و پیش‌بینی لینک بر اساس میزان عضویت به هر کلاس انجام می‌شود. این مدل‌ها عموماً با تحلیل‌های بیزی بررسی می‌شوند و به صورت مستقیم در قالب (۸.۳) قابل بیان نیستند.

• مدل‌های ویژگی‌پنهان^{۲۵}: در این روش‌ها هدف نگاشت گره‌ها به یک فضای پنهان با بعد مشخص و سپس پیش‌بینی وجود یا عدم وجود لینک با استفاده از ویژگی‌های استخراج شده است. بنابراین در قالب (۸.۳) داریم:

$$\hat{G}_{ij}(F, \Lambda) = L(F_i^T \Lambda F_j) \quad (11.3)$$

که F_i و F_j بردارهای ویژگی رئوس و Λ ماتریسی نماینده چگونگی تعامل بین ویژگی‌ها است.

Semi-supervised^{۲۲}

Link propagation^{۲۳}

Latent class models^{۲۴}

Latent feature models^{۲۵}

• **مدل‌های ساختار پنهان^{۲۶}** : این مدل‌ها ایده مدل‌های اختلاطی^{۲۷} را در داده‌های رابطه‌ای مانند شبکه‌ها پیاده می‌کنند. برای مثال [۶۲] مدل خود را بر این اساس ارائه می‌دهد که کل گراف متشکل از تعدادی خوش است و هر گره از یک خوش در گراف تحت توزیع چندجمله‌ای نمونه‌برداری شده است. مرجع [۶۳] این ایده را گسترش داده و فرض می‌کند هر گره می‌تواند تحت توزیع دریکله^{۲۹} متعلق به چندین خوش نیز باشد. بر همین اساس روش‌های مشابهی برای پیش‌بینی لینک و خوشبندی گراف نیز ارائه شده است. البته در حالت کلی این گونه مدل‌سازی‌های به وجود آمدن لینک در شبکه‌ها به مدل‌سازی بلوکی اتفاقی^{۳۰} معروف هستند [۶۲]. بنا بر مدل بلوکی اتفاقی، به وجود آمدن لینک‌ها از یک دیگر مستقلند به شرط این که تعلق به خوش‌ها برای دو سر لینک داده شده باشد. اگرچه روش‌های ساختار پنهان به روش‌های ویژگی پنهان نزدیک است، این مدل‌ها عموماً بیزی بوده و استنتاج در آن‌ها اکثراً با روش‌های MCMC انجام می‌شود و قابل بیان در قالب (۸.۳) نیستند.

روشی که ما در این فصل ارائه خواهیم داد، در دسته مدل‌های ویژگی پنهان قرار دارد. در ادامه به بیان جزئیات روش‌های اخیر در زمینه مدل‌های ویژگی پنهان خواهیم پرداخت.

در مرجع [۵۲] از فاکتورگیری ماتریس در پیش‌بینی لینک استفاده شده و تابع هدف در آن، مینیمم نمودن AUC^{۳۱} یا سطح زیر نمودار ROC^{۳۲} است. اطلاعات بیشتر در مورد این نمودار در مرجع [۶۴] قابل مشاهده است. دلیل انتخاب این تابع هدف در این کار، دقیقاً همان پدیده تنک بودن شبکه‌ها است. به بیان دقیق‌تر، اگر پیش‌بینی لینک را به صورت یک مسئله دوکلاسه بنگریم، یعنی داده‌ها با زوج‌های رئوس مشخص شده و برچسب‌ها مقدار موجود برای آن زوج در ماتریس مجاورت باشد، بین دو کلاس موجود عدم تعادل زیادی وجود دارد. یعنی تعداد لینک‌ها (مقادیر ۱ در ماتریس مجاورت) بسیار کمتر از تعداد غیرلینک‌ها (مقادیر ۰ در ماتریس مجاورت) است. در چنین شرایطی، معیارهای پیشین به وضوح معرف دقت کلاس‌بندی نیست. به عنوان مثال با در نظر گرفتن معیار صحت^{۳۳} پیش‌بینی همه داده‌های نامشخص به مقدار ۰ صحت بالایی را حاصل خواهد

Latent structure models^{۲۶}

Mixture models^{۲۷}

Multinomial distribution^{۲۸}

Dirichlet distribution^{۲۹}

Stochastic Block Modeling^{۳۰}

Area under ROC curve^{۳۱}

Receiver Operating Characteristic^{۳۲}

Accuracy^{۳۳}

کرد (با توجه به جدول ۱.۱ ، صحت در همه موارد بالای ۹۹٪ خواهد بود) . روشی که در [۵۲] ارائه شده تحت دو سناریوی مختلف، یعنی با داده خارجی و بدون داده خارجی ارزیابی شده است. با وجود این، تابع هدفی که در این روش تعریف شده، نیازمند این است که تمامی زوج‌های لینک و غیرلینک مشخص حداقل یک بار در آن مشاهده شوند. این نیازمندی باعث می‌شود که این روش نتواند مقیاس‌پذیری خوبی به گراف‌های بزرگ داشته باشد. چرا که در نظر گرفتن همه زوج‌های لینک و غیر لینک در یک گراف تنک ($|E| = O(|V|^3)$) پیچیدگی زمانی و فضایی $O(|V|^3)$ خواهد داشت. هرچند روش‌هایی مانند نمونه‌گیری و کاهش گرادیان تصادفی برای رفع این مشکلات در نظر گرفته شده اما هر دوی این روش‌ها می‌توانند در گراف‌های بزرگ همگرایی را به شدت کند و نادقيق نمایند.

مراجع [۶۵، ۶۶] روش‌های احتمالاتی مشابهی را بر اساس فاکتورگیری ماتریس و با مدل تولید کننده برنولی-لジستیک برای پیش‌بینی لینک ارائه داده‌اند. هر دو روش از الگوریتم MM^{۳۴} [۶۷] برای بهینه‌سازی تابع هدف استفاده می‌کنند. در الگوریتم MM صرفاً تابع جایگزین تابع هدف اصلی با شرایطی خاص در نظر گرفته می‌شود و سعی می‌شود این تابع جایگزین بهینه شود. اگر هدف مینیمم سازی تابع هدف باشد، تابع جایگزین بایستی در همه نقاط بزرگتر از تابع اصلی بوده و دارای مینیممی برابر با تابع اصلی باشد. استفاده از روش MM معمولاً محدود به مواردی است که تابع هدف اصلی تابع مشتق‌پذیر و خوش‌رفتاری نباشد. در این فصل پس از ارائه روش پیشنهادی، نشان داده می‌شود که استفاده از مدل برنولی-لジستیک نمی‌تواند به گراف‌های بزرگ مقیاس‌پذیر باشد.

مراجع [۶۸] تلاش می‌کند تا با مدل آزمایش برنولی، مسئله برازش لجیستیک^{۳۵} برای ماتریس‌های مجاورت با روش‌های MCMC را حل کند. روش‌های MCMC به دلیل نیاز به تعداد تکرار بالا و همگرایی کند، نمی‌توانند برای مسائل با مقیاس بزرگ مناسب باشند.

یکی از مهم‌ترین پارامترها در مدل‌های ویژگی پنهان، بعد فضای پنهان است. قرار دادن یک مقدار بهینه برای بعد فضای پنهان بسیار زمان‌گیر است، چرا که ملزم به اجرای کامل برای هر مقدار بعد است. مراجع [۶۹، ۷۰] تلاش می‌کنند که روش‌هایی بدون پارامتر برای تعیین بعد فضای پنهان ارائه دهند که در آن از روش‌های تمام بیزی^{۳۶} و فرایند بوفه هندی^{۳۷} استفاده می‌کنند. هر چند تئوری و نتایج گرفته شده برای این روش‌ها، محدود به مجموعه داده‌های کوچک است.

Minorization-Maximization^{۳۴}

Logistic regression^{۳۵}

Fully Bayesian^{۳۶}

Indian Buffet Process^{۳۷}

دیگر پژوهش‌های مرتبط

در مرجع [۷۱] روشی برای حل مسئله تکمیل شبکه در شرایطی کلی آورده شده است. بدین معنی که در این روش فرض عدم وجود رئوس و یال‌ها پیش گرفته شده است. هر چند مسئله عدم وجود رئوس با روش‌های کلاسیک تخمین اندازه شبکه انجام می‌شود [۷۲]. در این روش فرض شده که گراف زیرین از مدل کرونکر^{۳۸} [۴۹] پیروی می‌کند. برای استنتاج پارامتر در این روش، از الگوریتم EM و نمونه‌برداری گیس^{۳۹} استفاده شده است. همان طور که پیش از این گفته شد، نمونه‌برداری گیس و دیگر روش‌های مشابه MCMC مقیاس‌پذیری قابل قبولی به مجموعه داده‌های بزرگ ندارند. همچنین [۷۳] نیز مسئله تکمیل شبکه را تحت نمونه‌برداری بازدیدی^{۴۰} بررسی می‌کند. در نمونه‌برداری بازدیدی فرض بر این است که برای هر گره انتخاب شده امکان دسترسی به یال‌های مجاور به آن گره نیز فراهم شده است. جدول ۱.۳ خلاصه‌ای از آن چه در این بخش گفته شد را نمایش می‌دهد.

Kronecker graph model^{۳۸}

Gibbs Sampling^{۳۹}

Survey sampling^{۴۰}.

جدول ۱.۳ : برخی مطالعات مرتبط با کاربردهای تکمیل ماتریس و فاکتورگیری ماتریس در تحلیل شبکه‌ها و پیش‌بینی لینک

نقطاط ضعف	پیش‌فرضها	روش‌ها	سال انتشار	مسئله مورد بررسی	زمینه مطالعه
فرض دوبخشی بودن گراف	رتبه پایین بودن ماتریس، نمونه‌برداری غیریکنواخت	انتشار اطلاعات و بیشینه‌سازی تأثیر	۲۰۰۹	تکمیل ماتریس با نمونه‌برداری تصادفی غیریکنواخت [۵۷]	
ناتمام ماندن الگوریتم در حالات خاص	وجود خوش‌های کامل در گراف	بهینه‌سازی محدب، برنامه‌ریزی نیمه معین	۲۰۱۱	خوش‌بندی گراف ناکامل [۵۹]	تکمیل ماتریس
فرض علامت‌دار بودن شبکه	شبکه علامت‌دار، تعادل محلی و عمومی	تکمیل ماتریس، فاکتورگیری ماتریس	۲۰۱۲	استنتاج علامت‌یال در شبکه‌های علامت‌دار [۵۸]	
پیچیدگی زمانی بالا	—	روش‌های MCMC	۲۰۰۹	رگرسیون لجیستیک با ماتریس‌های مجاورت [۶۸]	
بهینه‌سازی کند و نادقيق	—	الگوریتم MM	۲۰۱۱	پیش‌بینی لینک تحت مدل برنولی- لجیستیک [۶۵]	فاکتورگیری ماتریس
پیچیدگی زمانی و فضایی بسیار بالا	در نظر گرفتن زوج‌های لینک و غیرلینک	بهینه‌سازی AUC، کاهش گرادیان تصادفی	۲۰۱۱	پیش‌بینی لینک با فاکتورگیری ماتریس [۵۲]	
پیچیدگی زمانی بالا	—	تحلیل تمام بیزی، فرایند بوفه هندی	۲۰۱۲	مدل‌های ویژگی پنهان بدون پارامتر [۷۰]	
فرض نمونه‌برداری بازدیدی	نمونه‌برداری بازدیدی	یادگیری با ناظر با فرض مدل بلوکی	۲۰۰۹	تکمیل شبکه [۷۳]	تکمیل شبکه
پیچیدگی زمانی بالا	پیروی از مدل کرونکر	الگوریتم EM ، روش‌های MCMC	۲۰۱۱	تکمیل شبکه [۷۱]	

۲۰۳ روش ارائه شده

در این بخش، ابتدا مسئله به صورت رسمی بیان شده و سپس در مورد مدل تولید کننده شبکه مطالعه آورده می‌شود. با فرض ساختار تنک در شبکه‌ها، نشان داده می‌شود که استنتاج پارامتر در مدل یاد شده مناسب شبکه‌های با مقیاس بالا است.

۱۰۲۰۳ بیان رسمی مسئله

گراف بدون جهت ($G(V, E)$) را با ماتریس مجاورت M که V مجموعه رئوس و E مجموعه یال‌های بدون جهت هستند، در نظر می‌گیریم. ماتریس ویژگی‌های پنهان را با F نشان می‌دهیم و نیز F_i را به عنوان $\hat{\alpha}$ مین‌ستون این ماتریس در نظر می‌گیریم. فرض می‌کنیم k نشان‌دهنده بعد فضای پنهان است. هدف بدست آوردن ماتریس ویژگی F^* بهینه است.

در این پایان‌نامه برای ساده‌سازی فرمول‌ها گراف مورد نظر را بدون جهت درنظر می‌گیریم. هرچند روش ارائه شده برای گراف‌های جهت‌دار نیز می‌تواند به کار گرفته شود.

فرض می‌کنیم ماتریس مجاورت M با یک فرایند پواسون مدل شده است و پارامتر توزیع را برای

$$\text{برابر } F_i^T F_j \text{ قرار می‌دهیم.}$$

$$M_{ij} \sim \text{Poisson}(F_i^T F_j) \quad (12.3)$$

همچنین فرض می‌کنیم درایه‌ها هر یک دارای توزیعی مستقل به شرط مشخص بودن ماتریس ویژگی باشند:

$$P(M|F) = \prod_{i,j} \frac{(F_i^T F_j)^{M_{ij}} e^{-F_i^T F_j}}{M_{ij}!} \quad (13.3)$$

این گونه فاکتورگیری از ماتریس برای گراف‌های بدون جهت مناسب است. برای گراف‌های جهت‌دار نیز فاکتورگیری‌هایی نظیر

$$M_{ij} \sim \text{Poisson}(F_i^T \Lambda F_j) \quad (14.3)$$

که Λ یک ماتریس غیر متقارن است، پیشنهاد شده‌اند [۷۴]. در اکثر مقالات فاکتورگیری ماتریس در گراف، هر بعد از این ویژگی‌ها به عنوان نماینده میزان تعلق به گروه یا خوشه‌ای خاص در گراف در نظر گرفته می‌شوند.

ایده استفاده از مدل برنولی برای گراف‌های ساده در مرجع [۶۵] برای پیش‌بینی لینک در شبکه‌ها مورد استفاده قرار گرفته است و نشان داده شده که پیچیدگی روش در زمانی خطی نسبت به تعداد

مشاهدات از گراف شبکه انجام می‌گیرد. هر چند در قسمت‌های بعدی، این امر نشان داده خواهد شد که استفاده از مدل برنولی نمی‌تواند به خوبی به گراف‌های بزرگ مقیاس پذیر باشد. در مقابل، نشان داده خواهد شد استفاده از مدل پواسون به عنوان مدل تولیدکننده زیرین، مانند مدل برنولی درست بوده و همچنین موجب تسريع استنتاج پارامتر از زمانی نمایی به زمانی خطی نسبت به تعداد رئوس گراف (به شرط تنک بودن گراف مذکور) خواهد شد. بنا بر دانش ما، استفاده از مدل تولیدکننده پواسون برای پیش‌بینی لینک تاکنون صورت نگرفته است.

توزيع پواسون معمولاً برای برآش پواسون برای داده‌های شمارشی و جداول احتمال وقوع^{۴۱} (مانند توزيع برنولی در برآش لجیستیک) به کار گرفته می‌شود [۷۵]. اگرچه ماتریس‌های مجاورت را نیز می‌توان به منزله نوعی خاص از داده شمارشی دانست، بسیاری از وقایع که موجب به وجود آمدن یا از بین رفتن لینک در شبکه‌ها می‌شوند، مانند فرستادن درخواست دوستی، پیام خصوصی، ایجاد ارتباط و انتقال داده و بسیاری وقایع دیگر، می‌توانند با فرایند پواسون مدل شوند [۷۶]. علاوه بر این، آزمایش‌های متنوع و جامع در مرجع [۷۷] نشان‌دهنده این حقیقت است که درستنمایی پواسون در مدل‌های بلوکی اتفاقی برای گراف‌های شبکه‌ای ساده، دارای عملکردی مشابه درستنمایی برنولی است و می‌تواند به همان اندازه در مدل‌سازی ساختار شبکه‌ها موثر باشد. در ادامه برای روشن‌تر شدن این موضوع، جزئیات آزمایش در [۷۸] آمده است.

مقایسه دقیق پیش‌بینی لینک با درستنمایی برنولی و درستنمایی پواسون

این بخش، بیان کننده آزمایشی است که مرجع [۷۷] برای انجام این مقایسه ترتیب داده است. بنا بر مدل بلوکی اتفاقی، احتمال به وجود آمدن لینک‌ها به شرط دانستن تعلق رئوس به خوش‌های از هم مستقل است. به طور دقیق‌تر، مدل بلوکی اتفاقی گره‌ها را به مجموعه خوش‌های z تقسیم می‌کند به طوری که احتمال وجود لینک بین دو گروه r و s توسط پارامتر η_{rs} داده شده باشد. این مدل فرض می‌کند عناصر ماتریس مجاورت A_{ij} مستقل توزيع شده‌اند. احتمال رخداد یک گراف با ماتریس مجاورت A به شرط دانستن z و η تحت درستنمایی برنولی برابر است با:

$$P(A|\eta, z) = \prod_{i < j} \eta_{z_i z_j}^{A_{ij}} (1 - \eta_{z_i z_j})^{1 - A_{ij}} \quad (15.3)$$

که در آن z خوش‌های است که رأس i به آن متعلق است. تخمین ماکزیمم درستنمایی برای مدل برنولی، برابر $\hat{\eta}_{rs} = \frac{N_{rs}^+}{N_{rs}^+ + N_{rs}^-}$ خواهد بود که N_{rs}^+ و N_{rs}^- به ترتیب تعداد لینک‌ها و غیرلینک‌ها بین دو گروه r و s است.

همچنین احتمال رخداد یک گراف با ماتریس مجاورت A به شرط دانستن z و η تحت درستنمایی

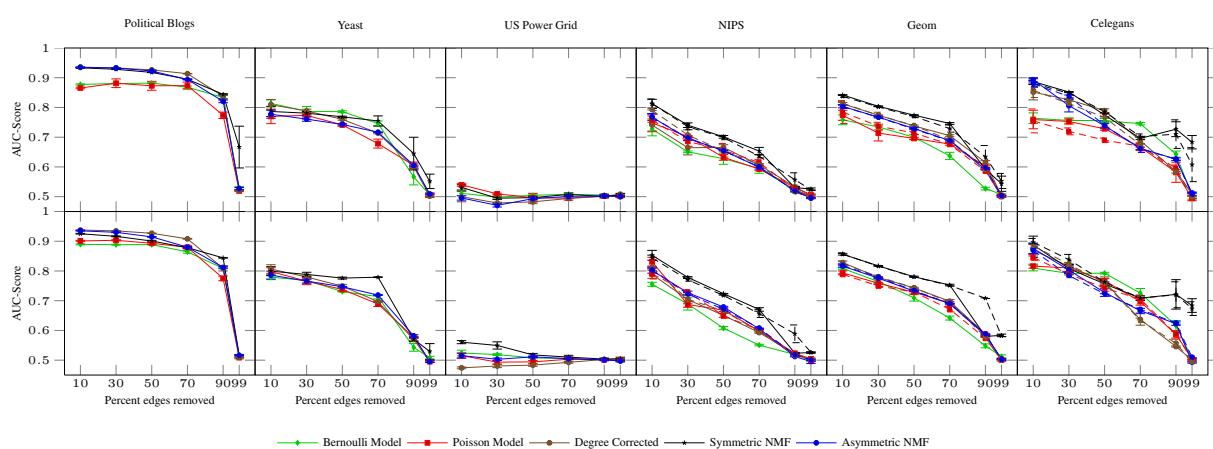
پواسون برابر است با:

$$P(A|\eta, z) = \prod_{i < j} \frac{\eta_{z_i z_j}^{A_{ij}}}{A_{ij}!} \exp(-\eta_{z_i z_j}) \times \prod_i \frac{(\frac{1}{2}\eta_{z_i z_i})^{A_{ii}/2}}{(A_{ii}/2)!} \exp\left(-\frac{1}{2}\eta_{z_i z_i}\right) \quad (16.3)$$

تخمین ماکزیمم درستنمایی برای مدل پواسون، برابر $\hat{\eta}_{rs} = \frac{N_{rs}^+}{n_r n_s}$ که در آن n_r تعداد رئوس گروه r است.

مرجع [۷۷] علاوه بر مقایسه این دو درستنمایی، درستنمایی پواسون با تصحیح درجه و فاکتورگیری ماتریس تحت تابع خطای KL-Divergence تعمیم یافته با روش مرجع [۷۸] را نیز در مقایسه شرکت می‌دهد که از ذکر جزئیات این دو خودداری می‌کنیم.

برای انجام آزمایش، نویسندهای مرجع [۷۷] شش شبکه را انتخاب کرده و پس از مشخص کردن اختصاص هر گره به خوشه‌ای خاص و با حذف درصدهای مختلف از ماتریس مجاورت این شبکه‌ها، مدل‌های فوق را بر روی شبکه‌ها استنتاج می‌کنند. نتایج مربوط به این آزمایش در شکل ۲.۳ نشان داده شده است. همان‌طوری که مشاهده می‌شود دو درستنمایی پواسون و برنولی دارای دقیق نزدیک به هم هستند.



شکل ۲.۳: مقایسه درستنمایی برنولی و پواسون برای گراف‌های ساده در [۷۷]
ردیف بالا، اجرا به ازای تقسیم شبکه به ۵ خوشه و ردیف پایین، اجرا به ازای تقسیم شبکه به ۱۰ خوشه است. همان‌طوری که مشاهده می‌شود تمامی مدل‌ها از دقت یکسانی برخوردار هستند.

با توجه به نتیجه آزمایش فوق و نتیجه نهایی که نویسندهای مرجع [۷۷] با تکیه بر تقریب توزیع برنولی با توزیع پواسون در گراف‌های تنک ذکر می‌کنند، به نظر می‌رسد گراف‌های ساده می‌توانند به

خوبی با فرایند پواسون مدل شوند.
بنا بر این مشاهدات، در روش ارائه شده مدل تولیدکننده پواسون برای گراف‌های تحت بررسی در نظر گرفته خواهد شد.

۲۰۲۰۳ مسئله بهینه‌سازی

برای تخمین پارامترهای مدل مذکور در (۱۳.۳) از روش ماکزیمم درستنمایی استفاده می‌کنیم که در آن به دنبال یافتن پارامترهایی هستیم که درستنمایی ماکزیمم را داشته باشند که به عبارتی محتمل‌ترین پارامترها خواهند بود. ماکزیمم درستنمایی برای مدل مذکور می‌تواند به صورت زیر نوشته شود:

$$\min_F lP(M|F) = \sum_{i,j} -M_{ij} \log(F_i^T F_j) + F_i^T F_j + C \quad (17.3)$$

که در آن lP منفی لگاریتم درستنمایی و C ثابتی مستقل از F است. مسئله (۱۷.۳) می‌تواند به عنوان نوعی از مسئله فاکتورگیری ماتریس مدل شود. بدین معنا که ما در بی‌یافتن F^* هستیم که:

$$F^* = \arg \min_F l(M, L(F)) \quad (18.3)$$

با مقایسه (۱۷.۳) و (۱۸.۳)، منفی لگاریتم درستنمایی و $F^T F$ به ترتیبتابع خطای $(\cdot, \cdot, l(\cdot, \cdot))$ و تابع پیوند $(\cdot, L(\cdot))$ هستند. همچنین ذکر این نکته ضروری است که کمینه‌سازی منفی لگاریتم درستنمایی برای توزیع پواسون معادل کمینه‌سازی تابع خطای KL-Divergence است [۲]. بنابراین $(\cdot, \cdot, l(\cdot, \cdot))$ را می‌توان تابع خطای KL-Divergence تعمیم‌یافته نیز در نظر گرفت.

۳۰۲۰۳ استنتاج پارامترها

برای مرحله استنتاج پارامتر، ابتدا این نکته را مذکور می‌شویم که دامنه مسئله (۱۷.۳) برابر است با مجموعه ماتریس‌های F که:

$$\forall i, j \quad F_i^T F_j > 0 \quad (19.3)$$

برای ساده‌سازی دامنه مسئله، ما فقط یک زیرمجموعه خاص از این دامنه را در نظر می‌گیریم:
 $\forall i \quad F_i > 0 \quad (20.3)$

این ساده‌سازی به علاوه موجب بدست آوردن ویژگی‌های تنک‌تر و قابل تفسیرتر نیز خواهد شد [۲]. مسئله (۱۷.۳) یک مسئله محدب بر حسب تمامی بردارهای ویژگی پنهان نیست. اما اگر فقط یک بردار ویژگی مانند F_i متغیر و بقیه ثابت در نظر گرفته شوند (بدین معنی که بهینه‌سازی فقط بر

اساس F_i انجام شود)، بهینه‌سازی تبدیل به یک مسئله بهینه‌سازی محدب خواهد شد. بنابراین ما از روش کاهش مؤلفه بلوکی استفاده می‌کنیم تا $lP(M|F)$ را کمینه کنیم. هر چند نقطه کمینه نهایی یک کمینه محلی است، اما مشخص نیست مسائل فاکتورگیری ماتریس به صورت کلی دارای کمینه‌ساز عمومی باشند. به طور کلی این مسائل دارای کمینه‌سازهای متعدد و بسیار نزدیک به هم هستند [۲].

بنابراین با ثابت نگهداشتن همه بردار ویژگی‌ها به غیر از F_i داریم:

$$f(F_i) = lP(M|F) \quad (21.3)$$

از آن جایی که f محدب است و البته پاسخ فرم بسته‌ای برای $\nabla f(F_i) = 0$ نمی‌توان یافت، معادله تکراری کاهش گرادیان برای F_i برابر خواهد بود با:

$$F_i^{n+1} = F_i^{(n)} - \eta \nabla f(F_i^{(n)}) \quad (22.3)$$

که در آن η نرخ یادگیری است. $\nabla f(F_i)$ برابر است با:

$$\nabla f(F_i) = 2 \sum_j F_j \left(1 - \frac{M_{ij}}{F_i^T F_j} \right) \quad (23.3)$$

که با ساده‌سازی می‌توان نوشت:

$$\nabla f(F_i) = 2 \sum_j F_j + 2 \sum_j -\frac{M_{ij}}{F_i^T F_j} F_j \quad (24.3)$$

در (۲۴.۳) می‌توان جملاتی را که مطابق درایه‌های صفر در ماتریس مجاورت هستند، از گرادیان حذف نمود و یک مرحله دیگر ساده‌سازی را انجام داد:

$$\nabla f(F_i) = 2 \sum_j F_j + \sum_{(i,j) \in E} -\frac{2}{F_i^T F_j} F_j \quad (25.3)$$

محاسبه گرادیان به شکل معمول، از مرتبه خطی بر حسب تعداد رئوس خواهد بود. اما با یک تغییر کوچک در ترتیب محاسبه گرادیان می‌توان این سرعت را بهبود بخشید. تقریباً این تغییرات که اکثراً با محاسبه از پیش متغیرهای درگیر در بهینه‌سازی انجام می‌شوند، در همه روش‌های مشابه در فاکتورگیری ماتریس دنبال می‌شود [۷۹، ۸۰].

با نگاه به (۲۵.۳) می‌توان دریافت که اگر جمله مجموع اول، از پیش محاسبه شده باشد، محاسبه تمام گرادیان برای هر بردار ویژگی، در مرتبه خطی از تعداد همسایگان رأس متناظر انجام خواهد شد. با توجه به تنک بودن گراف‌های واقعی، این مرتبه بسیار بسیار کمتر از مرتبه قبلی خواهد بود. از این رو، پیش از این که تکرارهای الگوریتم آغاز شود، مجموع همه بردارهای ویژگی را محاسبه کرده و بعد از بهینه نمودن هر مؤلفه، این مجموع را فقط برای این مؤلفه به روز خواهیم کرد. به عبارت دیگر، اگر F_i^{new} و F_i^{old} به ترتیب مؤلفه قبل از بهینه‌سازی و بعد از بهینه‌سازی باشند، مجموع بردارهای ویژگی را

به صورت زیر به روز می‌کنیم:

$$\sum_j F_j \leftarrow \sum_j F_j - F_i^{old} + F_i^{new} \quad (26.3)$$

به این ترتیب گرادیان برای هر مؤلفه در مرتبه خطی از تعداد همسایگان رأس متناظر انجام خواهد شد. برای مدل برنولی-لジستیک (که در مرجع [۶۵] استفاده شده است) می‌توان نشان داد که کاهش گرادیان برای هر مؤلفه نیاز به زمانی خطی نسبت به تعداد تمامی رئوس دارد. در مدل برنولی-لジستیک، تابع درستنمایی برابر است با:

$$P(M|F) = \prod_{i,j} (\sigma(F_i^T F_j))^{M_{ij}} (1 - \sigma(F_i^T F_j))^{1-M_{ij}} \quad (27.3)$$

که در آن σ تابع سیگموید است. پس منفی لگاریتم درستنمایی برابر است با:

$$lP(M|F) = \sum_{i,j} -M_{ij}(F_i^T F_j) + \log(1 + F_i^T F_j) + C \quad (28.3)$$

با قرار دادن $f(F_i)$ ، گرادیان برابر خواهد بود با:

$$\nabla f(F_i) = \sum_j -M_{ij} F_j + \frac{F_j}{1 + F_i^T F_j} \quad (29.3)$$

به وضوح، (۲۹.۳) نمی‌تواند با تکنیکی که در (۲۵.۳) استفاده شد، محاسبه شود و به $O(|V|)$ زمان نیاز دارد.

پس از هر به روزرسانی کاهش گرادیان، مؤلفه مورد نظر (فرض کنیم F_i) را به دامنه مطرح شده در (۲۰.۳) نگاشت می‌کنیم. می‌توان نشان داد که این نگاشت حاصل فرایند کاهش گرادیان مبدایی ۴۲، طبق قید مطرح شده در (۲۰.۳) می‌باشد.

برای کاهش گرادیان، ما روش‌های نیوتون و شبکه‌نیوتون^{۴۳} متنوعی را تست کردیم، اما در روش‌های مطرح شده عدد حالت^{۴۴} ماتریس هسین^{۴۵} برابر عدد بزرگی برای مجموعه داده‌های در نظر گرفته شده می‌شد که این بهینه‌سازی را کندتر و نادقيقتر می‌کند. برای تعیین نرخ یادگیری، ما از جستجوی خطی بازگشتی^{۴۶} استفاده کردیم. می‌توان نشان داد که برای این جستجو، بررسی شرط وولف^{۴۷} می‌تواند به روشنی مشابه محاسبه گرادیان محاسبه شود. شرط وولف برای اطمینان از کاهش به اندازه کافی در

Proximal gradient descent^{۴۸}

Quasi-Newton^{۴۹}

Condition number^{۴۴}

Hessian^{۴۵}

Backtracking line search^{۴۹}

Wolfe^{۴۷}

روش‌های کاهش گرادیان بررسی می‌شود.

یک نرخ یادگیری در شرایط وولف صدق می‌کند، اگر به ازای $0 < \alpha < 1$ داشته باشیم:

$$f(F_i - \eta \nabla f(F_i)) < f(F_i) - \alpha \eta \|\nabla f(F_i)\|^2 \quad (30.3)$$

بسط دادن این شرط و انجام یک سری ساده‌سازی، منجر به نامعادله زیر می‌شود:

$$\sum_{(i,j) \in E} -(\log((F_i - \eta \nabla f(F_i))^T F_j) - \log(F_i^T F_j)) - \eta \nabla f(F_i)^T \sum_j F_j + \alpha \eta \|\nabla f(F_i)\|^2 < 0 \quad (31.3)$$

از آنجایی که گرادیان و جمع بردارهای ویژگی پیش از بررسی این شرط محاسبه شده، بررسی شرط وولف نیز در زمانی خطی بر حسب تعداد همسایه‌های گره متناظر با مؤلفه کنونی محاسبه خواهد شد. خلاصه‌ای از روش ارائه شده در الگوریتم ۱.۲.۳ قابل مشاهده است.

Algorithm 3.2.1 Scalable Link Prediction using Matrix Factorization (SLPMF)

Require: M (Input Matrix),

k (Latent Space Dimension)

```

1: Initialize  $F$  //Initialization
2: Calculate  $\sum_j F_j$ 
3: repeat
4:   for  $i=1$  to  $|V|$  do //Coordinate Descent
5:     repeat
6:       Calculate the gradient  $\nabla f(F_i)$  from (25.3)
7:       Choose the step size  $\eta$  by checking (31.3)
8:        $F_i^* \leftarrow F_i - \eta \nabla f(F_i)$ 
9:        $\sum_j F_j \leftarrow \sum_j F_j - F_i + F_i^*$ 
10:       $F_i \leftarrow F_i^*$ 
11:    until Gradient Descent Convergence
12:  end for
13: until Coordinate Descent Convergence

```

Ensure: F

۴۰۲۰۳ تحلیل پیچیدگی

پیچیدگی زمانی

تا خط (۲) الگوریتم ۱.۲.۳، زمانی از مرتبه $O(k|V|)$ طول می‌کشد تا مقداردهی اولیه و محاسبه مجموع بردارهای ویژگی انجام شود. فرض کنیم ثابت‌های c ، g و l به ترتیب میانگین تکرارهای کاهش مؤلفه، کاهش گرادیان و جستجوی خطی باشند. بدین ترتیب برای هر مؤلفه، خط (۶) زمان $O(kN)$ را خواهد برد که N میانگین تعداد همسایگان رأس متناظر با هر مؤلفه است. همچنین خط (۷) زمان $O(lkN)$ و خط (۸) و (۹) زمان $O(k)$ خواهند داشت، که در عمل k مقداری کوچک خواهد داشت. بنابراین، به طور کلی پیچیدگی زمانی الگوریتم برابر خواهد بود با:

$$O(k|V| + c|V|g(l+1)kN) \quad (۳۲.۳)$$

به دلیل فرض تنک بودن شبکه‌ها، می‌توان مقدار N را ثابت در نظر گرفت. همچنین برای مقادیر c ، g و l نیز آزمایش‌ها مقادیر ناچیزی نشان دادند. بنابراین می‌توان گفت که پیچیدگی الگوریتم فوق $O(|V|)$ است.

روش دیگر برای محاسبه دقیق‌تر پیچیدگی زمانی، جمع بستن هر تکرار حلقه خط (۴) است. پیچیدگی هر بار اجرای حلقه $O(g(l+1)kN_i)$ است. بنابراین جمع همگی این حلقه‌ها $O(g(l+1)k\sum_i N_i) = O(g(l+1)k|E|)$ پیچیدگی زمانی خواهد داشت. پس پیچیدگی الگوریتم فوق به صورت دقیق‌تر برابر است با:

$$O(k|V| + c|E|g(l+1)k) \quad (۳۳.۳)$$

البته با توجه به فرض تنک بودن شبکه‌ها، می‌توان گفت: $E = O(|V|)$. پس پیچیدگی نهایی الگوریتم را می‌توان $O(|V|)$ در نظر گرفت.

پیچیدگی فضایی

یکی از مزیت‌های الگوریتم ارائه شده، عملکرد آن بر اساس تکرار بر روی همسایگان در داخلی‌ترین حلقه الگوریتم است. بنا بر این ویژگی، می‌توان گراف مورد نظر را در لیست مجاورت ذخیره نمود که دارای پیچیدگی $O(E)$ است. برای متغیر F نیز نیاز به $O(k|V|)$ فضا خواهیم داشت، هر چند k مقدار کوچکی خواهد داشت. بنابراین پیچیدگی فضایی نهایی برابر است با:

$$O(k|V| + |E|) \quad (۳۴.۳)$$

که البته این پیچیدگی با فرض تنک بودن شبکه‌ها همان $O(|V|)$ خواهد بود.

۵.۲.۳ پیاده‌سازی با الگوریتم برآورد-بیشینه‌سازی

در این بخش پیاده‌سازی روش ارائه شده را تحت الگوریتم برآورد-بیشینه‌سازی^{۴۸} شرح خواهیم داد. از آنجایی که در روش ارائه شده ماکزیمم درستنمایی در حضور داده‌های ناکامل اعمال می‌شود، می‌توان درستنمایی را بر روی کل داده‌ها، یعنی هم داده‌های مشخص و هم داده‌های نامشخص تعریف کرد. بدین ترتیب اگر O مجموعه درایه‌های مشخص ماتریس مجاورت و U مجموعه درایه‌های نامشخص ماتریس مجاورت باشند، می‌توان منفی لگاریتم درستنمایی بر روی کل داده‌ها را به صورت زیر تعریف کرد:

$$\text{LP}(M|F) = \sum_{(i,j) \in O} -M_{ij} \log(F_i^T F_j) + F_i^T F_j + \sum_{(k,l) \in U} -M_{k,l} \log(F_k^T F_l) + F_k^T F_l \quad (35.3)$$

ذکر این نکته ضروری است که در خود روش ارائه شده، درستنمایی بر روی یک گراف مشاهده شده تعریف شد، بنابراین صفر بودن یک درایه در ماتریس مجاورت برای الگوریتم صرفاً به منزله عدم وجود لینک نخواهد بود.

از آنجایی که M یک ماتریس ناکامل است، نمی‌توان از روش ماکزیمم درستنمایی به صورت مستقیم استفاده کرد، چرا که هنوز نمی‌توان درستنمایی را به خاطر وجود درایه‌های نامشخص محاسبه کرد. برای محاسبه درستنمایی لازم است درایه‌های نامشخص مقدار مناسبی بگیرند. از آن گذشته، اکنون ما با دو مجموعه مجهول مواجه هستیم: یکی پارامتر F و دیگری درایه‌های مجموعه U . اما نکته‌ای که به نظر می‌رسد این است که با داشتن هر کدام از این مجهول‌ها می‌توان دیگری را بدست آورد: با داشتن U و با ماکزیمم درستنمایی می‌توان به F رسید و با داشتن F ، می‌توان نمونه‌های توزیع مربوط به درستنمایی یا به عبارتی درایه‌های U را مشخص کرد، چرا که U از توزیعی تبعیت می‌کند که توسط F کاملاً مشخص می‌شود. این رویه الگوریتم برآورد-بیشینه‌سازی را در ذهن تداعی می‌کند [۸۱]. در این الگوریتم که البته بستری است برای طراحی الگوریتم‌های دیگر، دو مرحله وجود دارد: مرحله اول، تعیین مقادیر نامشخص می‌باشد که به عنوان مرحله برآورد^{۴۹} شناخته می‌شود. مرحله دوم، تعیین پارامترها است که به عنوان بیشینه‌سازی^{۵۰} شناخته می‌شود.

به صورت دقیق‌تر در مرحله برآورد، مقدار منفی لگاریتم درستنمایی برای کلیه داده‌ها محاسبه

Expectation-Maximization^{۴۸}

Expectation: E-step^{۴۹}

Maximization: M-step^{۵۰}

می‌شود که این مقدار، $Q(F|F^T)$ است:

$$Q(F|F^{(t)}) = E_{U|O,F^{(t)}}[lP(O, U|F)] \quad (36.3)$$

که به وضوح $Q(F|F^T) = lP(O, U|F) = lP(M|F)$ مقدار نهایی است با:

$$Q(F|F^{(t)}) = \sum_{(i,j) \in O} -M_{ij} \log(F_i^T F_j) + F_i^T F_j + \sum_{(k,l) \in U} -F_k^T F_l^{(t)} \log(F_k^T F_l) + F_k^T F_l \quad (37.3)$$

به طور خلاصه، در مرحله برآورد، به جای درایه‌های نامشخص، مقدار میانگین مربوطه به آنها جایگذاری می‌شود.

در مرحله بیشینه‌سازی، به دنبال حل مسئله زیر خواهیم بود:

$$F^{(t+1)} = \arg \min_F Q(F|F^t) \quad (38.3)$$

از آن‌جایی که $Q(F|F^t)$ همان منفی لگاریتم درستنایی در مرحله t است، بنابراین مرحله بیشینه‌سازی همان ماکزیمم درستنایی است که چگونگی انجام آن در بخش‌های پیشین شرح داده شد.

پیاده‌سازی روش ارائه شده در چارچوب برآورد-بیشینه‌سازی اگرچه ممکن است دقت بهتری داشته باشد، اما پیچیدگی فضایی و زمانی الگوریتم را افزایش می‌دهد. به طور دقیق‌تر، در مرحله برآورد نیاز به جایگزینی همه مقادیر نامشخص است. چون در اکثر آزمایش‌ها، حدود ۱۰ درصد از درایه‌ها را جایگزین می‌کنیم، پس در مرحله برآورد نیاز به فضا و زمانی از مرتبه $O(|V|^2)$ خواهیم داشت. بنابراین اجرای برآورد-بیشینه‌سازی به نظر نمی‌رسد مقیاس‌پذیری خوبی به شبکه‌های بزرگ داشته باشد. همچنین آزمایش‌ها نشان می‌دهد اجرای برآورد-بیشینه‌سازی در تعداد تکرارهای بسیار کم (۲ تا ۳ تکرار) به همگرایی می‌رسد. از آن‌جایی که هر مرحله برآورد-بیشینه‌سازی برابر با یک بار اجرای ماکزیمم درستنایی است، پس به نظر می‌رسد اجرای یک بار ماکزیمم درستنایی دقت و سرعت خوبی را ارائه خواهد داد.

محاسبه مرحله برآورد

برای ساده‌سازی محاسبه، تعریف می‌کنیم: $U = \{u_1, u_2, \dots, u_{|U|}\}$ که U همان مجموعه درایه‌های نامشخص ماتریس M است. پس طبق فرض استقلال شرطی درایه‌ها خواهیم داشت:

$$P(U|O, F^{(t)}) = P(U|F^{(t)}) = \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \quad (39.3)$$

تعریف می‌کنیم: $\sum_U = \Sigma_{u_1} \dots \Sigma_{u_{|U|}}$ که جمع بروی تمامی مقادیر هر کدام از درایه‌های U است.
بنابراین، می‌توان نوشت:

$$\begin{aligned} Q(F|F^{(t)}) &= \sum_U \left[lP(O, U|F) \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \right] \\ &= \left(\sum_U \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \right) \sum_{(i,j) \in O} -M_{ij} \log(F_i^T F_j) + F_i^T F_j \\ &\quad + \sum_U \left(\sum_{u_y \in U} -M_{u_y} \log(F_{u_y}) + F_{u_y} \right) \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \quad (40.3) \end{aligned}$$

با ساده‌سازی داریم:

$$\begin{aligned} Q(F|F^{(t)}) &= \sum_{(i,j) \in O} -M_{ij} \log(F_i^T F_j) + F_i^T F_j \\ &\quad + \left(\sum_U \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \right) \left(\sum_{u_y \in U} F_{u_y} \right) + \sum_U \left(\sum_{u_y \in U} -M_{u_y} \log(F_{u_y}) \right) \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \\ &= \sum_{(i,j) \in O} -M_{ij} \log(F_i^T F_j) + F_i^T F_j + \sum_{(k,l) \in U} F_k^T F_l + \sum_U \left(\sum_{u_y \in U} -M_{u_y} \log(F_{u_y}) \right) \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \quad (41.3) \end{aligned}$$

با ساده‌سازی آخرین جمله Q خواهیم داشت:

$$\begin{aligned} &\sum_U \left(\sum_{u_y \in U} -M_{u_y} \log(F_{u_y}) \right) \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \\ &= \sum_U (-M_{u_1} \log(F_{u_1})) \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \\ &\quad + \dots \\ &\quad + \sum_U (-M_{u_{|U|}} \log(F_{u_{|U|}})) \prod_{u_x \in U} P(U_{u_x}|F^{(t)}) \quad (42.3) \end{aligned}$$

برای i -امین جمله (۴۲.۳) خواهیم داشت:

$$\begin{aligned} & \sum_U (-M_{u_i} \log(F_{u_i})) \prod_{u_x \in U} P(U_{u_x} | F^{(t)}) \\ &= \left(\sum_{U - \{u_i\}} \prod_{u_x \in U - \{u_i\}} P(U_{u_x} | F^{(t)}) \right) \times \left(\sum_{u_i} -M_{u_i} \log(F_{u_i}) P(U_{u_i} | F^{(t)}) \right) \\ &= \sum_{u_i} -M_{u_i} \log(F_{u_i}) P(U_{u_i} | F^{(t)}) = -F_{u_i}^{(t)} \log(F_{u_i}) \quad (43.3) \end{aligned}$$

بنابراین (۴۲.۳) ساده خواهد شد به:

$$\sum_U \left(\sum_{u_y \in U} -M_{u_y} \log(F_{u_y}) \right) \prod_{u_x \in U} P(U_{u_x} | F^{(t)}) = \sum_{u_x \in U} -F_{u_x}^{(t)} \log(F_{u_x}) \quad (44.3)$$

در نهایت Q به صورت زیر نوشته می‌شود:

$$Q(F|F^{(t)}) = \sum_{(i,j) \in O} -M_{ij} \log(F_i^T F_j) + F_i^T F_j + \sum_{(k,l) \in U} -F_k^T F_l^{(t)} \log(F_k^T F_l) + F_k^T F_l \quad (45.3)$$

۳.۰.۳ نتایج شبیه‌سازی

۱۰.۳.۳ تنظیمات شبیه‌سازی

مجموعه داده‌ها

برای شبیه‌سازی، چندین مجموعه داده‌ی واقعی از کاربردهای مختلف انتخاب شدند. این مجموعه داده‌ها و اطلاعات آماری آن‌ها در جدول ۱.۳.۳ لیست شده‌اند. این مجموعه داده‌ها عبارتند از:

- مجموعه داده Yeast از داده‌های زیستی با ۲,۳۶۱ گره و ۶,۹۱۴ یال [۴].
- از شبکه‌های اجتماعی، یک نمونه از Facebook با ۵,۷۹۳ گره و ۳۰,۷۵۳ یال که مجموعه پروقایل کاربرانی است که در یک شرکت بزرگ IT مشغول به کار بوده‌اند [۵].
- گراف شبکه اینترنت در سطح AS مربوط به منطقه Oregon در سال ۲۰۰۱ که ۱۱,۴۹۲ گره و ۲۳,۴۰۹ یال دارد [۶].

جدول ۲.۳: مجموعه داده‌های انتخاب شده برای ارزیابی عملکرد الگوریتم SLPMF

نام شبکه	تعداد گره‌ها	تعداد یال‌ها	میزان تنک بودن	میانگین درجه
[۴] Yeast	۲,۳۶۱	۶,۹۱۴	۰/۹۹۷۵۲	۲/۹
[۶] GrQc	۵,۲۴۲	۱۴,۴۹۰	۰/۹۹۸۹۱	۲/۸
[۷] FacebookL1	۵,۷۹۳	۳۰,۷۵۳	۰/۹۹۸۱۷	۵/۳
[۸] AS	۱۱,۴۹۲	۲۳,۴۰۹	۰/۹۹۹۶۵	۲/۰
[۹] COND-MAT	۲۳,۱۳۳	۹۳,۴۹۷	۰/۹۹۹۶۵	۴/۰
[۱۰] P2P	۲۶,۵۱۸	۶۵,۳۶۹	۰/۹۹۹۸۱	۲/۵

- شبکه COND-MAT و GrQc که مربوط به نویسنده‌گان همکار در مجموعه مقالات فرستاده شده به پایگاه داده arXiv که دارای ۲۳,۱۳۳ گره و ۹۳,۴۹۷ یال است [۶].
- شبکه نظری به نظری مربوط به ۲۴ آگوست ۲۰۰۲ از شبکه Gnutella ^{۵۱} با ۲۶,۵۱۸ گره و ۶۵,۳۶۹ یال [۸، ۹].

روش‌های رقیب

روش ارائه شده با عنوان SLPMF شناخته می‌شود. روش‌های رقیب دیگر نیز عبارتند از:

- روش‌های بدون ناظر: سه روش بدون ناظر برای مقایسه انتخاب شدند: Adamic-Adar [۶۰] (PA) Preferential-Attachment (SH) Shortest-Path ، (AA)
- NMF-KL [۷۹] : روش‌های بسیاری برای مینیمم‌سازی تابع خطای KL-Divergence تعمیم یافته وجود دارند. ما روش ارائه شده در [۷۹] را به عنوان یک روش اخیر و کارا انتخاب کردیم. همان‌طور که پیش از این ذکر شد، ماکزیمم درستنمایی برای توزیع پواسون به عنوان تولید کننده ماتریس به مینیمم‌سازی تابع خطای KL-Divergence تعمیم یافته می‌انجامد. هر چند روش [۷۹] برای فیلترینگ اشتراکی ارائه شده است. نتایج شبیه‌سازی تفاوت در عملکرد فاکتورگیری‌های مختلف ماتریس را نشان خواهند داد.

- GLFM [۶۵] : این روش از مدل تولید کننده برنولی-لジستیک برای فاکتورگیری ماتریس استفاده می‌کند.

^{۵۱} البته این شبکه در اصل جهت‌دار است که ما با تعویض درایه‌های صفر متناظر با درایه‌های یک، ماتریس مجاورت، این شبکه را بدون جهت نمودیم.

همان‌طور که در بخش‌های پیشین اشاره شد، روش‌های زیادی بر اساس مدل‌های ویژگی پنهان برای پیش‌بینی لینک استفاده شده‌اند. اما روش ارائه شده در [۶۵] از بسیاری از این روش‌ها عملکرد بهتری در سرعت و دقت اجرا داشت. به همین دلیل این روش برای مقایسه انتخاب شد که البته در متداول‌تری کلی بسیار مشابه روش ارائه شده ماست. همچنین روش ارائه شده در [۵۲] نیز همان‌طور که اشاره شد دارای پیچیدگی زمانی بالایی بوده و برای ما امکان گرفتن نتایج در زمانی معقول ممکن نبود.

تنظیمات

برای همه آزمایش‌ها، ۱۰ درصد از درایه‌های ماتریس مجاورت به این ترتیب حذف شده‌اند: ابتدا یک درایه مانند (j, i) به صورت تصادفی انتخاب شده و درایه‌های (j, i) و (i, j) به لیست درایه‌های نامشخص اضافه می‌شوند و سپس اگر درایه (j, i) مقدار یک داشته باشد، مقدار این درایه به صفر تغییر پیدا می‌کند. برای ارزیابی روش از معیار AUC استفاده شد، زیرا این معیار در برابر عدم تعادل کلاس‌ها حساس نیست [۶۴].

ماتریس F به صورت تصادفی مقداردهی اولیه شد. تمامی نتایج به ازای ۱۰ بار اعتبارسنجی چندگانه ۵۲ ارائه شده‌اند. برای پیاده‌سازی از محیط متلب^{۵۳} استفاده شد و برای روش‌های رقیب، برای روش‌های بدون ناظر از پکیج LPMade [۸۲] و برای NMF-KL و GLFM از کد خود نویسنده‌گان استفاده شده است. شبیه‌سازی‌ها بر روی یک کامپیوتر با سیستم عامل ویندوز و پردازنده Intel Core i7 ۲.۶ GHz با ۱۲ گیگابایت حافظه انجام شده‌اند.

چگونگی انتخاب بعد فضای پنهان

به وضوح با بعد بیشتر، درجه آزادی بیشتری را می‌توان پوشش داد. اما ابعاد کمتر به ابعاد بزرگ‌تر ترجیح داده می‌شوند. یک روش ساده برای تعیین ابعاد مناسب این است که یک زیرمجموعه از درایه‌های گراف مشاهده شده را کنار گذاشته و به ازای ابعاد متفاوت الگوریتم را اجرا کنیم. تعداد بعدی که بهترین AUC را داشته باشد، به عنوان بعد ویژگی‌ها می‌تواند انتخاب شود. این روش برای هر سه روش SLPMF، NMF-KL و GLFM اجرا و بعدی که برای هر سه روش بهترین پیش‌بینی را داشت، برای مقایسه انتخاب شد.

جدول ۳.۳: مقایسه مقادیر AUC با روش SLPMF اعداد متمایز با قلم ضخیم نشان‌دهنده بهترین نتایج هستند.

<i>SH</i>	<i>PA</i>	<i>AA</i>	نام شبکه
۰/۶۹۲۵ ± ۰/۰۲۸	۰/۶۶۳۴ ± ۰/۰۲۴	۰/۶۹۱۷ ± ۰/۰۰۵	<i>Yeast</i>
۰/۷۴۱۱ ± ۰/۰۰۳	۰/۷۳۰۲ ± ۰/۰۱۴	۰/۷۵۵۴ ± ۰/۰۲۶	<i>GrQc</i>
۰/۷۸۷۱ ± ۰/۰۱۹	۰/۷۷۵۲ ± ۰/۰۱۱	۰/۸۰۱۴ ± ۰/۰۱۳	<i>FacebookL1</i>
۰/۶۹۳۴ ± ۰/۰۱۱	۰/۶۶۹۷ ± ۰/۰۱۶	۰/۷۱۸۲ ± ۰/۰۱۰	<i>AS</i>
۰/۷۱۲۴ ± ۰/۰۱۳	۰/۷۱۳۲ ± ۰/۰۱۴	۰/۷۳۰۱ ± ۰/۰۱۲	<i>COND-MAT</i>
۰/۶۲۴۹ ± ۰/۰۲۰	۰/۶۲۰۱ ± ۰/۰۰۵	۰/۶۱۱۱ ± ۰/۰۰۵	<i>P2P</i>

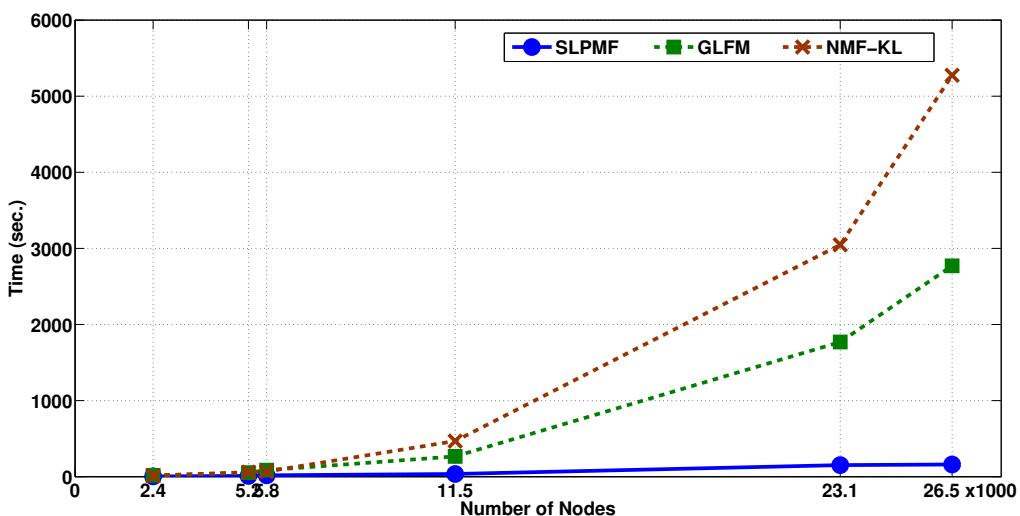
<i>SLPMF</i>	<i>GLFM</i>	<i>NMF-KL</i>	<i>k</i>	نام شبکه
۰/۸۲۳۰ ± ۰/۰۰۵	۰/۷۶۴۸ ± ۰/۰۱۰	۰/۷۵۷۷ ± ۰/۰۱۱	۱۰	<i>Yeast</i>
۰/۸۶۰۰ ± ۰/۰۱۲	۰/۷۳۳۱ ± ۰/۰۲۱	۰/۸۱۱۸ ± ۰/۰۱۷	۱۰	<i>GrQc</i>
۰/۹۲۵۰ ± ۰/۰۰۸	۰/۸۵۳۹ ± ۰/۰۲۳	۰/۹۱۳۵ ± ۰/۰۰۹	۱۰	<i>FacebookL1</i>
۰/۸۸۳۵ ± ۰/۰۱۵	۰/۷۰۱۳ ± ۰/۰۲۱	۰/۷۶۱۰ ± ۰/۰۰۵	۱۵	<i>AS</i>
۰/۹۲۳۸ ± ۰/۰۱۷	۰/۷۳۹۳ ± ۰/۰۱۱	۰/۸۸۶۵ ± ۰/۰۱۶	۲۵	<i>COND-MAT</i>
۰/۷۷۴۴ ± ۰/۰۲۱	۰/۶۹۵۰ ± ۰/۰۱۴	۰/۶۶۴۴ ± ۰/۰۳۱	۳۰	<i>P2P</i>

۲۰۳.۳ ارزیابی دقت

نتایج در جدول ۳.۳ قابل مشاهده هستند. اعداد با قلم ضخیم نشان‌دهنده بهترین نتیجه هستند و k بعد فضای پنهان برای روش‌های باناظر است. همان‌طور که مشاهده می‌شود روش ارائه شده در همه موارد از روش‌های بدون ناظر و باناظر در نظر گرفته شده بهتر عمل می‌کند. دلیل این که GLFM نمی‌تواند با وجود استفاده از مدل برنولی به نتیجه بهتری دست پیدا کند را می‌توان در نحوه بهینه‌سازی آن جستجو کرد. همان‌طور که پیش‌تر ذکر شد، این الگوریتم از روش MM برای تقریب ماتریس هسین استفاده می‌کند، اما بنا بر مشاهدات ما این ماتریس دارای عدد حالت بزرگی در مجموعه داده‌های انتخاب شده بود که موجب کندتر شدن و البته نادقيق شدن بهینه‌سازی می‌شود [۱۳]. AUC در روش NMF-KL به مرتب نزدیکتر به روش ارائه شده است. هر چند این روش برای پیش‌بینی لینک دو ایراد عده دارد: یکی این که این روش برای فیلترینگ اشتراکی ارائه شده و ماتریس‌های فاکتورگیری شده لزوماً ماتریس متقارن تولید نمی‌کنند. ثانیاً پیچیدگی زمانی این روش به مرتب از روش ارائه شده بسیار بیشتر است؛ همان‌طور که نویسندهای در مقاله مربوط می‌نویسند، پیچیدگی زمانی NMF-KL برابر $O(nmkd)$ است که n و m ابعاد ماتریس مورد نظر، k بعد فضای پنهان و d میانگین تعداد تکرارهای الگوریتم کاهش گرادیان برای هر مؤلفه است. بنابراین NMF-KL کندتر از روش ارائه شده همگرا خواهد شد.

۳.۰.۳ ارزیابی زمان اجرا و مقیاس‌پذیری

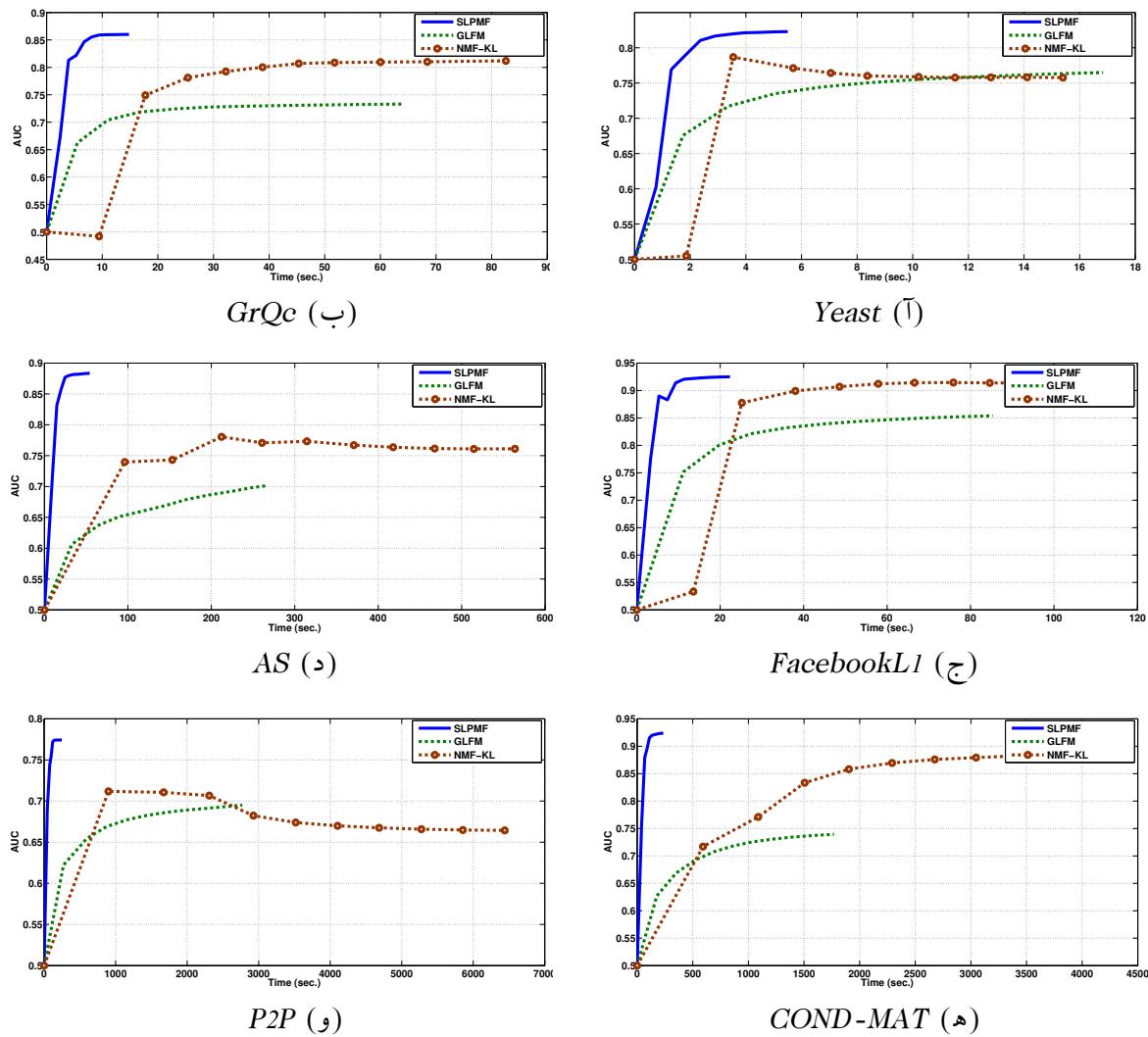
برای ارزیابی سرعت و مقیاس‌پذیری، زمان اجرای سه روش NMF-KL ، GLFM و SLPMF با یکدیگر در مجموعه داده‌های مختلف مقایسه شد. نمودارها در شکل ۳.۳ قابل مشاهده هستند. هر یک از اعداد در محور افقی، نشان‌دهنده تعداد گره‌های مجموعه داده‌ها هستند و مقادیر مربوط برای هر روش، زمان رسیدن به همگرایی (زمانی که AUC رشد کمتر از ۱٪ داشته باشد) با ماکزیمم ۱۰ تکرار برای هر الگوریتم است (در روش ارائه شده، حلقه خط (۴)). همان‌طور که انتظار می‌رفت، زمان اجرای روش ارائه شده به صورت خطی نسبت به تعداد رئوس افزایش پیدا می‌کند ولی در GLFM و NMF-KL این افزایش نمایی است. مطابق با آن‌چه در آزمایش قبلی ذکر شد، اعداد حالت بزرگ در روش GLFM و پیچیدگی زمانی بالاتر روش NMF-KL موجب کندی این دو روش می‌شوند.



شکل ۳.۳: مقایسه زمان اجرا با روش SLPMF

زمان اجرای روش ارائه شده به صورت خطی نسبت به تعداد رئوس افزایش پیدا می‌کند ولی در NMF-KL و GLFM این افزایش نمایی است.

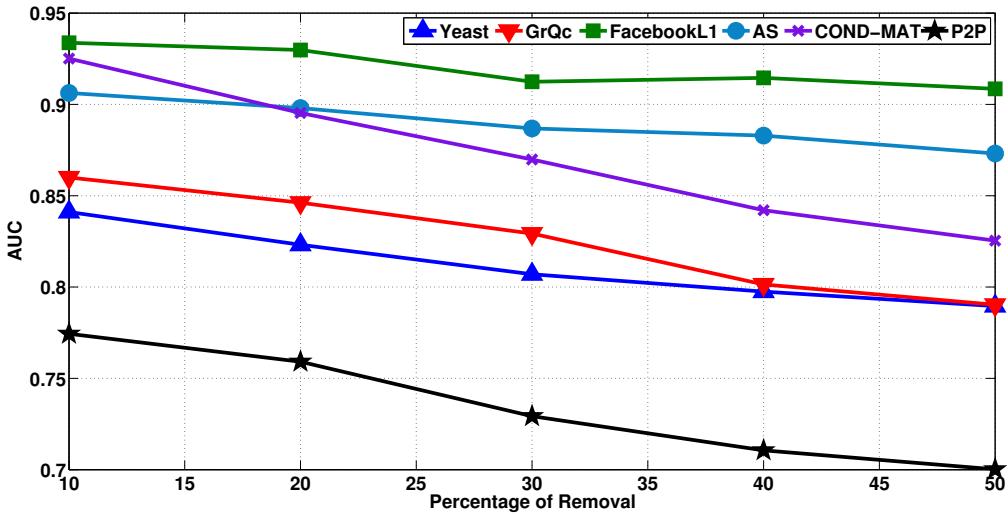
شکل ۴.۳ روند رشد AUC را برای روش‌های مختلف در مجموعه داده‌ها نشان می‌دهد. هر الگوریتم تا ۱۰ تکرار اجرا شده است. مشاهده می‌شود که روش ارائه شده سریع‌تر همگرا شده و در زمان اجرای یکسان، دارای AUC بهتری است.



شکل ۴.۳: مقایسه زمان اجرا در برابر میزان AUC با روش SLP MF و روشهای دیگر. نتایج نشان می‌دهند که روش ارائه شده سریع‌تر همگرا شده و در زمان اجرای یکسان، دارای AUC بهتری است.

۴.۳.۳ ارزیابی دقیق بر اساس میزان داده از دست رفته

کارایی روش ارائه شده در زمانی که میزان داده از دست رفته در حد بالایی باشد نیز مورد بررسی قرار گرفت. در این آزمایش میزان درایه‌های نامشخص را از ۱۰ درصد تا ۵۰ درصد تغییر دادیم. در شکل ۴.۳ مشاهده می‌شود که در همه مجموعه داده‌ها میزان AUC با شیبی کم تحت تأثیر از دست رفتن داده قرار می‌گیرد و با وجود از دست دادن داده تا حد بالای ۵۰ درصد، میزان AUC به طور میانگین تنها ۱۰ درصد کاهش داشته است.



شکل ۵.۳: ارزیابی حساسیت دقت روش SLPMF به میزان داده از دست رفته در همه مجموعه داده‌ها میزان AUC با شیوه کم تحت تأثیر از دست رفتن داده قرار می‌گیرد.

۴.۳ کاربرد در فاکتورگیری از ماتریس

همان‌طور که در بخش‌های پیشین اشاره شد، روش ارائه شده علاوه بر زمینه پیش‌بینی لینک در زمینه تقریب و فاکتورگیری نامنفی ماتریس نیز می‌تواند استفاده شود. در این بخش به بررسی این کاربرد می‌پردازیم.

در فاکتورگیری نامنفی ماتریس برای ماتریس $M_{m \times n}$ ، هدف یافتن دو فاکتور نامنفی $U_{m \times k}$ و $V_{n \times k}$ است به شرطی که یکتابع خطای خاص مانند مجدور خطای مینیمم شود.

در مورد کمینه‌سازی تابع مجدور خطای با فاکتورهای نامنفی پژوهش‌های بسیاری با قیود و پیش‌فرضهای متفاوتی انجام شده است [۲]. اما در این بخش ما تمرکز خود را بر روی تابع خطای KL - Divergence تعیین یافته معطوف می‌کنیم. مسئله فاکتورگیری ماتریس تحت تابع خطای KL -

Divergence تعیین یافته برای M و حاصل ضرب $U^T V$ برابر است با:

$$\arg \min_{(U,V) \geq 0} \sum_{i,j} \log \left(\frac{M_{ij}}{U_i^T V_j} \right) - M_{ij} + U_i^T V_j \quad (46.3)$$

که در آن U_i و V_j به ترتیب ستون i -ام و j -ام ماتریس‌های U و V هستند.

با توجه به روشی که در بخش‌های پیشین ارائه شد، به نظر می‌رسد می‌توان این روش را بر روی این مسئله نیز اعمال نمود. در ادامه تمامی فرمول‌ها برای ماتریس U آورده می‌شود. تمامی مشتق‌ها و فرمول‌ها با روشی مشابه برای ماتریس V نیز می‌توانند محاسبه شوند.

در مرجع [۷۹] روشی برای حل مسئله فاکتورگیری ماتریس تحت تابع خطای KL-Divergence تعیین یافته آورده شده است. در این روش، کاهش مؤلفه به صورت چرخشی^{۵۴} در سطح درایه‌های دو ماتریس U و V انجام می‌شود. به عبارتی دیگر اگر $(s) - h_{ir}$ برابر میزان کاهش تابع خطای پس از کمینه‌سازی مؤلفه ir -ام از ماتریس U و تغییر آن به اندازه s باشد، یعنی $s = U_{ir}^{new} - U_{ir}^{old}$ ، آن‌گاه خواهیم داشت:

$$h_{ir}(s) = \sum_{j=1}^n -M_{ij} \log (U_i^T V_j + s V_{rj}) + s V_{rj} + C \quad (47.3)$$

که C ثابتی مستقل از s است. بنابراین بهترین s برای بهینه‌سازی مؤلفه کنونی عبارتست از:

$$s^* = \arg \max_s -h_{ir}(s) \quad (48.3)$$

مرجع [۷۹] از روش نیوتون برای بهینه‌سازی هر مؤلفه استفاده می‌کند و با استفاده از خاصیت تحدب صریح^{۵۵} $h_{ir}(s)$ ثابت می‌کند این بهینه‌سازی برای هر مؤلفه نیازی به جستجوی نرخ یادگیری نیز ندارد. همچنین در ادامه دو قضیه را برای الگوریتم خود (CCD یا همان NMF-KL در بخش پیشین) مطرح می‌کند:

- هر نقطه حدی استخراج شده از CCD یک نقطه ایستا^{۵۶} است.
- هر نقطه حدی استخراج شده از CCD با اعمال هموارسازی l_1 و l_2 یک نقطه ایستا است.

پیچیدگی زمانی روش CCD برابر $O(nmkd)$ است که n و m ابعاد ماتریس مورد نظر، k بعد فضای پنهان و d میانگین تعداد تکرارهای الگوریتم کاهش گرادیان برای هر مؤلفه است. بخش اعظمی از روشی که در این بخش ارائه می‌دهیم، متکی بر روش SLPMF در بخش پیشین است. به طور دقیق‌تر، کاهش مؤلفه در روش ارائه شده، بر اساس مؤلفه‌های برداری (ونه مانند CCD مؤلفه‌های تک متغیری) است. مؤلفه‌های برداری همان ستون‌های دو ماتریس U و V هستند که در روش SLPMF ستون‌های ماتریس F بودند. بنابراین اگر فرض کنیم با ثابت فرض کردن همه ستون‌های دو ماتریس U و V ، خواهان اجرای کاهش گرادیان بر روی U_i هستیم، تابع هدف برای مؤلفه U_i برابر خواهد بود با:

$$f(U_i) = \sum_j -M_{ij} \log (U_i^T V_j) + U_i^T V_j \quad (49.3)$$

Cyclic^{۵۴}

Strict convexity^{۵۵}

Stationary Point^{۵۶}

بنابراین گرادیان تابع هدف تحت آن برابر خواهد بود با:

$$\nabla f(U_i) = \sum_j -\frac{M_{ij}}{U_i^T V_j} V_j + V_j \quad (50.3)$$

با ساده‌سازی گرادیان داریم:

$$\nabla f(U_i) = \sum_j V_j + \sum_{j: M_{ij} \neq 0} -\frac{M_{ij}}{U_i^T V_j} V_j \quad (51.3)$$

مانند تکنیکی که در روش SLPMF استفاده شد، می‌توان مقدار $\sum_j V_j$ برای کاهش مؤلفه‌های U_i (و $\sum_i U_i$ برای مؤلفه‌های V_j) را از پیش محاسبه کرد و بعد از هر کاهش مؤلفه مقدار جدید را در هر جمع جایگزین مقدار قبلی کرد.

۱۰۴.۳ بهینه‌سازی با انتخاب مؤلفه حریصانه: روش GSCD

به طور خلاصه، کمینه‌سازی تابع هدف را در مسئله فاکتورگیری ماتریس بر حسب مؤلفه‌های برداری، می‌توان به صورت زیر بیان کرد:

$$\min_{(U,V) \geq 0} f(U_1, \dots, U_m, V_1, \dots, V_n) \quad (52.3)$$

برای کمینه‌سازی تابع هدف، نیاز به انتخاب مؤلفه مناسب برای کاهش تابع هدف است. این انتخاب در ساده‌ترین حالت می‌تواند به صورت چرخشی باشد. بدین معنی که روی همه مؤلفه‌ها به تعداد یکسان و با ترتیبی چرخشی گرادیان را انجام می‌دهیم. این رویه در روش SLPMF نیز دنبال شده بود. البته، در روش SLPMF، با استفاده از تنک بودن ماتریس اصلی، زمانی بسیار کمتر صرف به روزرسانی و محاسبه گرادیان هر مؤلفه می‌شود. به علاوه در نامعادله (۳۱.۳) نشان دادیم که می‌توان جستجوی نرخ یادگیری در SLPMF را با همان زمان پیچیدگی کوتاه برای محاسبه گرادیان در ماتریس‌های تنک انجام داد.

برای مقیاس‌پذیری بیشتر، یک انتخاب مناسب برای اعمال کاهش مؤلفه، انتخاب مؤلفه‌ای است که دارای گرادیان با بزرگترین اندازه ^{۵۷} باشد [۸۴، ۸۵]. ما از این ایده برای ارائه روشی حریصانه برای فاکتورگیری ماتریس تحت KL-Divergence Loss، با عنوان GSCD استفاده می‌کنیم. در روش GSCD نیاز است که اعمال زیر تحت داده‌ساختاری مناسب برای ذخیره و بازیابی اندازه گرادیان مؤلفه‌ها به صورت کارا انجام شوند:

- مقداردهی اولیه داده ساختار.

- انتخاب مؤلفه دارای بزرگترین اندازه گرادیان.
- به روز رسانی اندازه گرادیان مؤلفه کنونی و احتمالاً مؤلفه‌هایی که پس از به روز رسانی تحت تأثیر قرار می‌گیرند.
- حذف مؤلفه‌هایی که دیگر نمی‌توان روی آن‌ها بهینه‌سازی انجام داد.

اندازه گرادیان مربوط به هر مؤلفه صرفاً مقداری حقیقی و مثبت است. به نظر می‌رسد داده‌ساختار هرم ^{۵۸} داده‌ساختاری مناسب و کارا برای این منظور باشد. چرا که اگر تعداد عناصر داده‌ساختار N باشد، مقداردهی اولیه داده ساختار در $O(N)$ و انتخاب مؤلفه با بیشترین مقدار در $O(1)$ و حذف یک مؤلفه در $O(\log N)$ قابل انجام است. برای به روز رسانی یک یا چند عنصر در هرم ذکر این نکته ضروری است که در داده ساختار استاندارد هرم چنین عملی گنجانده نشده است [۸۶] ، هر چند می‌توان با داشتن محل عنصر مورد نظر در داده ساختار، این عمل را به عنوان تعمیمی از عمل درج با هزینه $O(\log N)$ انجام داد.

برای ساده شدن توضیحات الگوریتم، جمله دوم گرادیان U که در (۵۱.۳) تعریف شده را ”مانده گرادیان“^{۵۹} نام‌گذاری می‌کنیم.
مراحل الگوریتم :

۱. محتویات داده ساختار هرم غیر از اعداد مربوط به اندازه گرادیان هر مؤلفه چیزی نخواهد بود. در ابتدای الگوریتم گرادیان ابتدا دو مجموع $\sum_i U_i$ و $\sum_j V_j$ که در گرادیان‌ها موثر هستند، محاسبه می‌شوند. سپس مانده گرادیان همه مؤلفه‌های U و V محاسبه شده و اندازه گرادیان‌های مؤلفه‌های U در یک هرم قرار می‌گیرند. فعلاً برای V هرمی تشکیل نمی‌شود. همچنین به همراه اندازه گرادیان‌های U که داده ساختار بر اساس آن شکل می‌گیرد، شماره مؤلفه متناظر نیز همراه با آن نگهداری می‌شود، چرا که در هنگام انتخاب بزرگترین گرادیان لازم است به مؤلفه متناظر آن در ماتریس U رجوع شود.

۲. اگر فرض کنیم یک مؤلفه از U مانند U_i را انتخاب کرده و بر روی آن کاهش گرادیان را انجام داده باشیم، لازم است گرادیان مربوط به U_i جدید را دو مرتبه محاسبه کرده و در هرم قرار دهیم. این کار مانند روش SLPMF با گذر فقط از روی درایه‌های ناصلف ماتریس ورودی و طبق فرمول گرادیان برای مؤلفه‌های U در (۵۱.۳)، در مرتبه تعداد درایه‌های غیر صفر متناظر

با سطر i -ام ماتریس M انجام می‌پذیرد؛ چرا که مجموع $\sum_j V_j$ از قبل محاسبه شده و در طول به روزرسانی ثابت است. پس از محاسبه دوباره گرادیان، دوباره اندازه آن را در هرم مربوط به مؤلفه‌های U قرار می‌دهیم.

۳. مؤلفه U_i به غیر از گرادیان خویش، در مجموع $\sum_i U_i$ نیز تأثیرگذار است. بنابراین پس از به روزرسانی U_i این مجموع را نیز به روز می‌کنیم.

۴. همچنین U_i در مانده گرادیان مؤلفه‌ایی از V که مربوط به ستون درایه‌های غیر صفر متناظر با سطر i -ام ماتریس M ، نیز تأثیرگذار است. بنابراین اگر V_j یکی از این مؤلفه‌ها ($M_{ij} \neq 0$) و $Res(\nabla f(V_j))$ مانده گرادیان تابع هدف بر حسب V_j باشد و همچنین U_i^{new} و U_i^{old} به $Res(\nabla f(V_j))$ ترتیب مؤلفه i -ام ماتریس U بعد و قبل از کاهش گرادیان باشند، به روز رسانی $(Res(\nabla f(V_j)))$ به صورت زیر انجام خواهد شد:

$$Res(\nabla f(V_j)) \leftarrow Res(\nabla f(V_j)) + \frac{M_{ij}}{U_i^{oldT} V_j} V_j - \frac{M_{ij}}{U_i^{newT} V_j} V_j \quad (53.3)$$

در روش GSCD برخلاف روش SCCD بین مؤلفه‌های U و V به صورت گردش-نوبتی عمل نمی‌شود. چرا که پس از هر به روزرسانی مؤلفه‌های U ($\sum_j V_j$) V (بردار مجموع $\sum_i U_i$) دچار تغییر می‌شود و این باعث می‌شود هرم مربوط به اندازه گرادیان‌های V (U) کاملاً تحت تأثیر قرار بگیرد. بنابراین برای حفظ کارایی، انتخاب مؤلفه‌ها را تا زمان همگرایی مناسب یا رسیدن به ماکزیمم تکرار، فقط بر روی مؤلفه‌های U (V) انجام می‌دهیم.

۵. پس از اتمام انتخاب مؤلفه بر روی U ، به سراغ مؤلفه‌های V رفته و در ابتدا مانده گرادیان‌ها و هرم اندازه گرادیان مؤلفه‌های V را تشکیل داده و سپس به همان روال بهینه‌سازی را انجام می‌دهیم.

۶. تنها در یک مورد ممکن است الگوریتم GSCD به بن بست برسد (یعنی پیش از زمان لازم تمام شود) و آن زمانی است که مؤلفه با بیشترین گرادیان نقطه ایستا باشد. در این صورت این مؤلفه با اعمال کاهش گرادیان تغییری نکرده و همواره در بالای هرم باقی می‌ماند. به منظور رفع این بن بست، زمان برخورد با چنین مؤلفه‌ایی، آن‌ها را از هرم حذف می‌کنیم.

همچنین می‌توان برای استفاده از GSCD از سیاست ترکیبی نیز استفاده نمود. برای مثال تکرارهای اولیه کاهش مؤلفه را به صورت چرخشی اجرا کرد تا مقادیر مانده گرادیان مقادیری پایدار باشند. در کل، الگوریتم ارائه شده برای انتخاب مؤلفه حریصانه را می‌توان در الگوریتم ۱.۴.۳ خلاصه کرد.

Algorithm 3.4.1 Greedy Scalable Coordinate Descent for NMF under KL-Divergence Loss

(GSCD)

Require: M (Input Matrix), k (Latent Space Dimension)

```

1: Initialize  $U$  and  $V$  //Initialization
2: Calculate  $\sum_i U_i$  and  $\sum_j V_j$ 
3: Calculate All  $Res(\nabla U_i)$  and  $Res(\nabla V_j)$ 
4: repeat
5:   Initialize and Heapify  $GradientHeap_U$ 
6:   repeat //Coordinate Descent
7:     Pick the Highest Gradient Norm from  $GradientHeap_U$  (e.g.  $U_i$ )
8:      $U_i^{old} \leftarrow U_i$ 
9:     Perform Gradient Descent on  $U_i$  and store it in  $U_i^{new}$ 
10:    if  $\|U_i - U_i^{new}\| \leq \epsilon$  then //Skip Stationary Point
11:      Delete  $U_i$  gradient from  $GradientHeap_U$ 
12:      continue;
13:    end if
14:     $\sum_i U_i \leftarrow \sum_i U_i - U_i^{old} + U_i^{new}$ 
15:    Recalculate  $\nabla f(U_i)$  and update its norm in  $GradientHeap_U$ 
16:    for all  $V_j$  where  $M_{ij} \neq 0$  do
17:       $Res(\nabla f(V_j)) \leftarrow Res(\nabla f(V_j)) + \frac{M_{ij}}{U_i^{old T} V_j} V_j - \frac{M_{ij}}{U_i^{new T} V_j} V_j$ 
18:    end for
19:     $U_i \leftarrow U_i^{new}$ 
20:    until Convergence or Maximum Iterations Reached
21:    Repeat above steps for  $V$  analogous to Line 5 – 20
22: until Coordinate Descent Convergence

```

Ensure: F

۲۰۴۰۳ تحلیل پیچیدگی زمانی و فضایی روش GSCD

دو خط ابتدایی الگوریتم هزینه $O((m+n)k)$ دارند. خط (۳) به دلیل گذر از روی همه درایه‌های غیرصفر ماتریس ورودی M دارای هزینه M_{NZ} (تعداد درایه‌های غیرصفر ماتریس M) خواهد بود. در هر بار اجرای حلقه خط (۴) و برای ماتریس U خط (۵) هزینه $O(m)$ خواهد داشت. در هر بار اجرای حلقه خط (۶)، خط (۷) و (۸) دارای هزینه ثابت و خط (۹) هزینه $O(g(l+1)N_U)$ خواهد بود خواهد که N_U میانگین تعداد درایه‌های غیرصفر سطرهای ماتریس M خواهد بود. شرط خط (۱۰) در صورت اجرا دارای هزینه $O(\log m)$ خواهد بود. خط (۱۴) هزینه ثابت داشته و خط (۱۵) و حلقه خط (۱۶) هزینه $O(\log m + N_U)$ خواهند داشت. با توجه به گذر الگوریتم فقط بر روی درایه‌های غیر صفر در بدنه حلقه خط (۴)، پیچیدگی الگوریتم برای یک بار گذر و بهینه‌سازی ماتریس U از خط (۵) تا (۲۰) برابر خواهد بود با:

$$O(m + c_U(\log m + N_U)) \quad (54.3)$$

که در آن c_U تعداد اجرای حلقه خط (۶) است.

به طور مشابه برای V نیز داریم:

$$O(n + c_V(\log n + N_V)) \quad (55.3)$$

که در آن c_V تعداد اجرای حلقه متناظر با حلقه خط (۶) برای ماتریس V است. در صورتی که تعداد اجرای حلقه خط (۴) ناچیز باشد، بنابراین پیچیدگی کلی الگوریتم GSCD برابر است با:

$$O(m + n + c_U(\log m + N_U) + c_V(\log n + N_V)) \quad (56.3)$$

با توجه به مکانیزم کلی الگوریتم، به نظر می‌رسد هزینه سرشکن شده الگوریتم به صورت ساده برابر $O(m \log m + n \log n + M_{NZ})$ باشد که M_{NZ} تعداد عناصر غیرصفر ماتریس M است. چرا که مکانیزم کلی الگوریتم به صورت گذر از روی درایه‌های غیر صفر است.

۳۰۴۰۳ آیا انتخاب مؤلفه حریصانه در پیش‌بینی لینک ممکن است؟

متاسفانه در مورد پیش‌بینی لینک که فقط یک فاکتور متغیر (F) در بهینه‌سازی وجود دارد، انتخاب مؤلفه حریصانه و به روزرسانی داده‌ساختارها به صورت دقیق ممکن نیست.

برای تشریح این مطلب، گرادیان مؤلفه i -ام از F را در نظر بگیرید:

$$\nabla f(F_i) = 2 \sum_j F_j + \sum_{(i,j) \in E} -\frac{2}{F_i^T F_j} F_j \quad (57.3)$$

همانند آنچه در روش GSVD داشتیم، این گرادیان را به دو بخش جمله مجموع و جمله مانده گرادیان تقسیم می‌کنیم و خواهیم داشت:

$$\nabla f(F_i) = \Sigma + R_i \quad (58.3)$$

که Σ جمله مجموع و ثابت برای همه مؤلفه‌ها و R_i جمله مانده گرادیان مؤلفه i -ام است. به وضوح با به روز کردن هر مؤلفه، جمله مجموع برای همه مؤلفه‌ها و به دنبال آن، گرادیان همه مؤلفه‌ها تغییر می‌کند. هر چند می‌توان با اعمال تغییراتی، یک روش حریصانه تقریبی برای پیش‌بینی لینک نیز ارائه داد.

فرض کنیم در داده‌ساختار هرم، مقادیر $\|\Sigma + R\|^2$ و $\|R\|^2$ نگهداری می‌شود. بنابراین با به روز شدن F_i ، گرادیان بقیه مؤلفه‌ها مانند مؤلفه j -ام به صورت زیر تغییر خواهد کرد:

$$\nabla f(F_j) = \Sigma + \Delta_i + R_j + \Delta_{R_j} \quad (59.3)$$

که در آن $\Delta_{R_j} = \frac{M_{ij}}{F_i^{*T} F_j} F_j - \frac{M_{ij}}{F_i^{*T} F_j} F_j$ برابر با تغییر در جمله مجموع و $\Delta_i = F_i^* - F_i$ برابر با تغییر در جمله مانده گرادیان مؤلفه j -ام است. به وضوح تغییر در مانده گرادیان فقط برای مؤلفه‌هایی اتفاق خواهد افتاد که رأس متناظر با آن‌ها با رأس i -ام همسایه باشد. بنابراین مقدار متناظر با مؤلفه j -ام به صورت زیر تغییر پیدا خواهد کرد:

$$\|\nabla f(F_j)\|^2 = \|\Sigma + R_j\|^2 + 2(\Sigma + R_j)^T (\Delta_i + \Delta_{R_j}) + \|\Delta_i + \Delta_{R_j}\|^2 \quad (60.3)$$

می‌توان این گرادیان را به صورت زیر بازنویسی کرد:

$$\|\nabla f(F_j)\|^2 = \|\Sigma + R_j\|^2 + 2\Sigma^T \Delta_i + \|\Delta_i\|^2 + 2R_j^T \Delta_i + 2(\Sigma + R_j)^T \Delta_{R_j} + 2\Delta_i^T \Delta_{R_j} + \|\Delta_{R_j}\|^2 \quad (61.3)$$

موارد زیر را در مورد این گرادیان می‌توان برشمود:

۱. جمله اول گرادیان یعنی $\|\Sigma + R_j\|^2$ که قبلاً در خود هرم برای همه مؤلفه‌ها محاسبه شده است.

۲. سه جمله آخر این گرادیان یعنی $2(\Sigma + R_j)^T \Delta_{R_j} + 2\Delta_i^T \Delta_{R_j} + \|\Delta_{R_j}\|^2$ مربوط به جملاتی است که زمانی غیر صفر هستند که مانده گرادیان مؤلفه j -ام صفر نباشد، یعنی رأس j -ام همسایه رأس i -ام در گراف اصلی شبکه باشد. با توجه به فرض تنک بودن شبکه‌ها، این سه جمله فقط برای تعداد کمی از مؤلفه‌ها غیر صفر است و می‌توان تغییرات این مؤلفه‌ها را در زمانی کوتاه

اعمال و در هرم به روزرسانی کرد. هزینه این کار $O(N \log |V|)$ خواهد بود که N میانگین تعداد همسایه‌ها در گراف و مقداری کوچک است.

۳. جملات $2\sum^T \Delta_i + \|\Delta_i\|^2$ برای همه عناصر هرم یکی هستند. بنابراین، اضافه کردن آنها به همه عناصر هرم ترتیب عناصر در هرم را تغییر نخواهند داد. پس می‌توان از آنها صرف نظر کرد.

۴. جمله $2R_j^T \Delta_i$ برای هر مؤلفه منحصر به فرد است و باید به همه درایه‌های هرم اضافه شود.

به عنوان تقریبی شهودی، می‌توان از جمله $2R_j^T \Delta_i$ برای همه مؤلفه‌ها صرف نظر کرد و امید داشت که این جمله تأثیر کمی در جابه‌جایی عناصر هرم داشته باشد. نتایج آزمایش‌های تجربی ما نشان می‌دهد که این مقدار البته مقدار کوچکی نسبت به عناصر هرم است.

۴.۴.۳ نتایج شبیه‌سازی

مجموعه داده‌ها

در این بخش از دو مجموعه داده زیر استفاده کردیم:

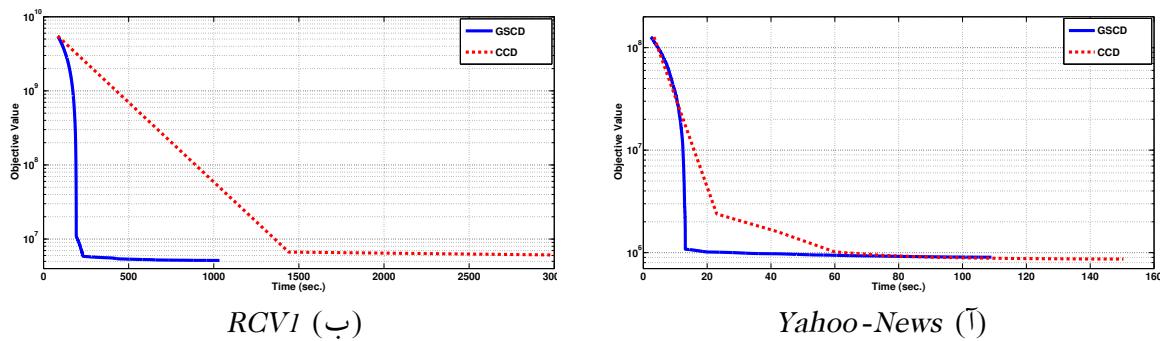
- مجموعه داده Yahoo-News [۷۹]: ماتریس سند-واژه 60 مربوط به تعدادی از اخبار وبسایت Yahoo است.

- مجموعه داده RCV1 [۸۸، ۸۷]: ماتریس سند-واژه مربوط به گروه خبری Reuters که مربوط به مجموعه‌ای از اخبار این گروه در سال ۲۰۰۰ است.

اطلاعات این دو مجموعه داده در جدول ۴.۳ آمده است. در این جدول k نشان دهنده بعد فضای پنهان مشخص شده برای مجموعه داده است.

جدول ۴.۳: مجموعه‌داده‌های انتخاب شده برای ارزیابی عملکرد الگوریتم GSCD

نام مجموعه داده	m	n	k	تعداد درایه‌های غیر صفر	میزان تنک بودن
[۷۹] Yahoo News	۲۱,۸۳۹	۲,۳۴۰	۲۰	۳۴۹,۷۹۲	۰/۹۹۳۱۶
[۸۷] RCV1	۲۳,۱۴۹	۴۷,۲۳۶	۲۰	۱,۷۵۷,۸۰۱	۰/۹۹۸۳۹



شکل ۳.۶.۳: مقایسه میزان کاهش مقدار تابع هدف در دو روش GSCD و CCD روش GSCD در زمان کمتر و با شیبی تندتر از CCD مقدار تابع هدف را کاهش می‌دهد.

روش رقیب

روش CCD [۷۹] به عنوان روشی کارا و جدید برای مقایسه انتخاب شد. مقایسه صورت گرفته فقط در باب مقیاس‌پذیری خواهد بود، چرا که روش CCD از لحاظ دقیق و متداول‌تری بهینه‌سازی (که همان کاهش مؤلفه بلوکی است) تفاوتی اندک با روش ارائه شده خواهد داشت.

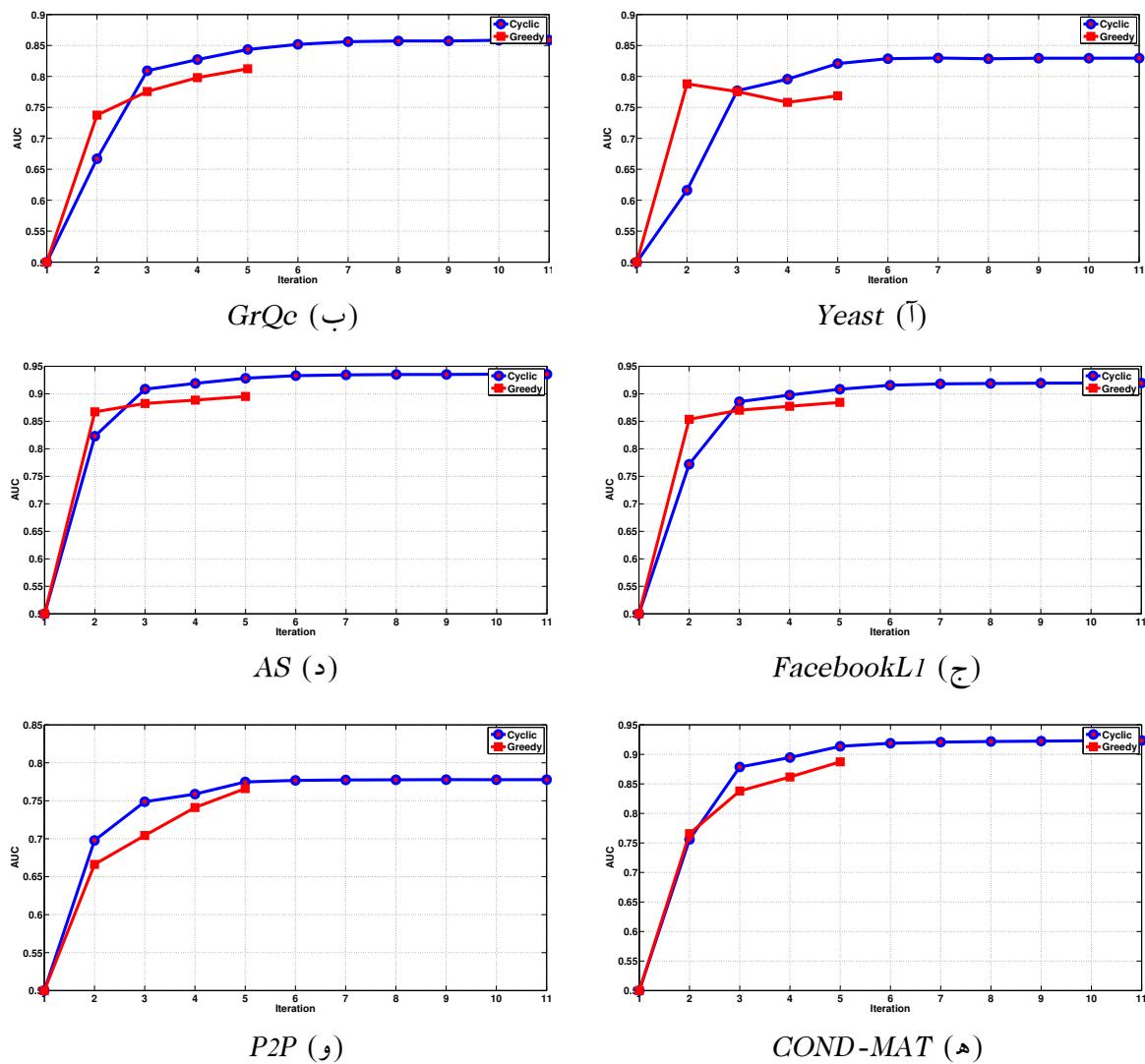
مقیاس‌پذیری در فاکتورگیری از ماتریس

در این بخش مقیاس‌پذیری روش‌های ارائه شده را بر اساس سرعت نزدیک‌شدن به مینیمم محلی در بررسی خواهیم کرد و مقدار تابع هدف KL-Divergence Loss را در برچسب‌های زمانی مختلف اندازه می‌گیریم. برای روش GSCD اولین تکرار کاهش مؤلفه را به صورت چرخشی پیاده کردیم تا مقادیر مانده گرادیان مقادیری پایدار باشند. شکل ۳.۶.۳ مقدار تابع هدف را برای روش‌های CCD و GSCD در برچسب‌های زمانی مختلف نشان می‌دهد. به دلیل تغییرات بزرگ در مقدار تابع هدف، محور عمودی (مقدار تابع هدف) در مقیاس لگاریتمی رسم شده است.

مشاهده می‌شود که در آغاز، هر دو الگوریتم با سرعتی تقریباً برابر پیش می‌روند، هر چند الگوریتم GSCD مقداری از زمان را صرف پیش‌محاسبه مجموعه‌ها و مانده گرادیان‌ها خواهد کرد. به مرور، روش GSCD در زمان کمتر و با شیبی تندتر از CCD مقدار تابع هدف را کاهش می‌دهد. این موجب می‌شود که GSCD تا ۶ برابر سریع‌تر از CCD به حوالی نقطه مینیمم محلی نزدیک می‌شود.

پیش‌بینی لینک با انتخاب مؤلفه حریصانه

اگر چه روشی که در بخش پیشین برای پیش‌بینی لینک با انتخاب مؤلفه حریصانه ارائه شد، شهودی بود، در این بخش به بررسی عملکرد این روش نیز می‌پردازیم. مجموعه داده‌هایی که برای این بخش



شکل ۳.۷.۳: مقایسه رشد AUC در نسخه چرخشی و حریصانه روش SLPMF در مراحل آغازین الگوریتم، نسخه حریصانه رشد بیشتری را در AUC ایجاد می‌کند، اما رفته رفته این رشد کنتر از نسخه چرخشی شده و در نهایت در مقدار AUC کمتری، متوقف می‌شود.

در نظر گرفتیم، همان مجموعه داده‌های معرفی شده برای پیش‌بینی لینک در روش SLPMF هستند. شکل ۳.۷.۳ میزان AUC را برای نسخه چرخشی و حریصانه روش SLPMF آن در تکرارهای مختلف الگوریتم نشان می‌دهد.

همان‌طور که مشاهده می‌شود، در مراحل آغازین الگوریتم، نسخه حریصانه رشد بیشتری را در AUC ایجاد می‌کند. اما رفته رفته این رشد کنتر از نسخه چرخشی شده و در نهایت در مقدار AUC کمتری، به دلیل خالی شدن داده‌ساختار هرم، متوقف می‌شود. به نظر می‌رسد نسخه چرخشی به مینیمم محلی بهتری از نسخه حریصانه می‌رسد. علت این پدیده را مسلماً در تقریب در نظر گرفته شده برای نسخه حریصانه می‌توان عنوان کرد. چرا که این تقریب علاوه بر ناهمگونی در بهینه‌سازی بر

روی مؤلفه‌ها، موجب می‌شود مؤلفه‌هایی که قبلاً بهینه‌سازی شده‌اند به مرور در انتهای الگوریتم بیشتر انتخاب شوند و زمانی نوبت به مؤلفه‌های با ماکزیمم گرادیان برسد که به حد کافی به مینیمم محلی نزدیک شده‌ایم. بنابراین گرچه روش حریصانه نمی‌تواند در کل بهتر از نسخه چرخشی عمل کند، اما می‌توان از روش حریصانه برای انتخاب نقطه شروع بهتر برای الگوریتم چرخشی استفاده کرد.

۵.۴.۳ پیاده‌سازی موازی روش GSCD

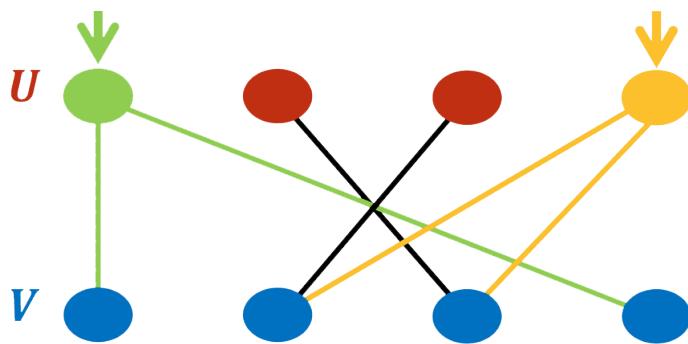
موازی‌سازی الگوریتم GSCD نیازمند این است که بدانیم تأثیر بهینه‌سازی یک مؤلفه بر روی چه بخش‌هایی از داده‌های ذخیره شده است؟ با توجه به الگوریتم روش GSCD، هنگام بهینه‌سازی مؤلفه‌های U به روزرسانی‌ها بر روی سه دسته داده زیر انجام می‌پذیرد:

- جمع ستون‌های U در خط (۱۴) الگوریتم.
- به روز کردن هرم بیشینه با جایگزینی اندازه مؤلفه‌ای که بهینه‌سازی شده است، در خط (۱۵) الگوریتم.
- به روز رسانی مانده گرادیان‌هایی از V که با مؤلفه بهینه‌سازی شده در ارتباط هستند، در خط (۱۷) الگوریتم.

از طرفی می‌دانیم که خود مؤلفه‌های U پس از بهینه‌سازی، تأثیری بر گرادیان یکدیگر ندارند. همچنین، به وضوح مشخص است که به طور مثال با بهینه‌سازی موازی دو مؤلفه از U نمی‌توان جمع ستون‌ها و هرم بیشینه را به طور موازی به روز کرد. چرا که حافظه حاوی جمع ستون‌ها و هرم بیشینه بین دو مؤلفه مشترک است. اما اگر این دو مؤلفه طوری انتخاب شوند که با یکدیگر همسایه مشترکی در مؤلفه‌های V نداشته باشند، یعنی هر دو بر روی یک بردار از V تأثیرگذار نباشند، می‌توان به صورت موازی این دو مؤلفه را بهینه‌سازی و مانده گرادیان‌های V مرتبط با آن‌ها را به روز کرد.

با این تغییر، می‌توان داخلی‌ترین حلقه الگوریتم GSCD که همان بهینه‌سازی مؤلفه‌ها است، را به طور موازی انجام داد. هرچند به روز رسانی جمع مؤلفه‌های U و هرم بیشینه متعلق به U نیازمند به روز رسانی ترتیبی است، تعداد این به روز رسانی‌ها با توجه به فرض تنک بودن تعداد همسایه‌های هر گره، تعدادی ناچیز است و می‌توان هزینه ثابت برای آن‌ها در نظر گرفت.

در مورد انتخاب مؤلفه‌هایی که دارای همسایه مشترک نباشند، بهترین حالت با انتخاب مؤلفه‌هایی که همسایه مشترک نداشته و دارای بزرگترین گرادیان هستند، حاصل می‌شود. هر چند جستجو با چنین قیدهایی به نظر نمی‌رسد در زمان ثابت انجام‌پذیر باشد. بنابراین یک روش ساده برای انتخاب مؤلفه‌ها



شکل ۸.۳: تأثیر بهینه‌سازی یک مؤلفه بر روی مؤلفه‌های دیگر تأثیر بهینه‌سازی یک مؤلفه از U فقط بر روی همسایه‌های آن در V است. بنابراین دو مؤلفه نشان داده شده با پیکان می‌توانند به طور موازی وارد بهینه‌سازی شوند.

می‌تواند انتخاب تصادفی مؤلفه‌ها باشد که با توجه به تنک بودن گراف متناظر با U و V به احتمال زیاد مؤلفه‌هایی را بر می‌گرداند که دارای همسایه مشترک نباشند.

قطعه کد ۲.۴.۳ خلاصه‌ای از ایده موازی نمودن بهینه‌سازی هر مؤلفه را در الگوریتم GSCD نشان می‌دهد.

در مورد پیاده‌سازی موازی برای پیش‌بینی لینک باید گفت که امکان موازی سازی نیز برای پیش‌بینی لینک وجود ندارد؛ چرا که بهینه‌سازی یک مؤلفه در گرادیان تمامی مؤلفه‌های دیگر تأثیرگذار خواهد بود.

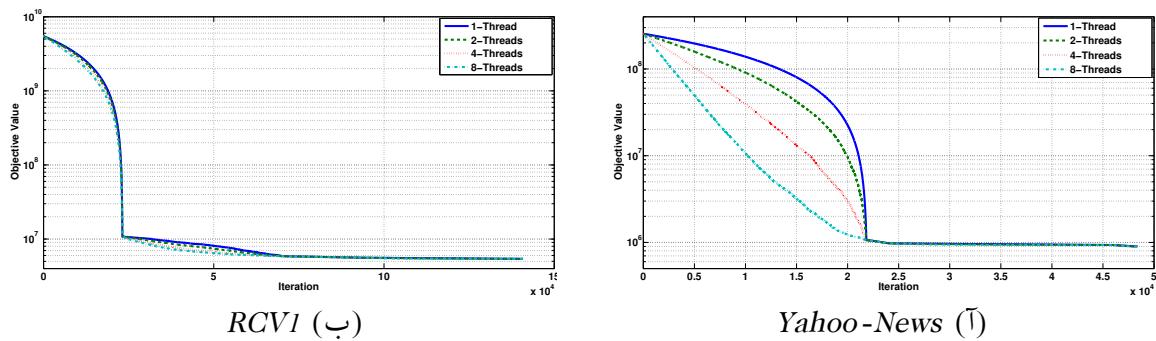
Algorithm 3.4.2 Parallel Greedy Scalable Coordinate Descent for NMF under KL-Divergence Loss (*PGSCD*)**Require:** M (Input Matrix), k (Latent Space Dimension), p (Number of Parallel Instances)

```

1: Initialize  $U$  and  $V$  //Initialization
2: Calculate  $\sum_i U_i$  and  $\sum_j V_j$ 
3: Calculate All  $Res(\nabla U_i)$  and  $Res(\nabla V_j)$ 
4: repeat
5:   Initialize and Heapify  $GradientHeap_U$ 
6:   repeat //Coordinate Descent
7:     Pick the Highest Gradient Norm from  $GradientHeap_U$  (e.g.  $U_i$ )
8:     choose  $p - 1$  other coordinates from  $U$  randomly and check their adjacencies to
       be independent
9:   //PARALLEL FOR ALL CHOSEN COORDINATES  $U_i$ 
10:  Perform Gradient Descent on  $U_i$  and store it in  $U_i^{new}$ 
11:  for all  $V_j$  where  $M_{ij} \neq 0$  do
12:     $Res(\nabla f(V_j)) \leftarrow Res(\nabla f(V_j)) + \frac{M_{ij}}{U_i^{oldT} V_j} V_j - \frac{M_{ij}}{U_i^{newT} V_j} V_j$ 
13:  end for
14:  //END OF PARALLEL COMPUTATION
15:  for all Chosen Coordinates  $U_i$  do
16:    if  $\|U_i - U_i^{new}\| \leq \epsilon$  then //Skip Stationary Point
17:      Delete  $U_i$  gradient from  $GradientHeap_U$ 
18:    end if
19:     $\sum_i U_i \leftarrow \sum_i U_i - U_i^{old} + U_i^{new}$ 
20:    Recalculate  $\nabla f(U_i)$  and update its norm in  $GradientHeap_U$ 
21:  end for
22:  until Convergence or Maximum Iterations Reached
23:  Repeat above steps for  $V$  analogous to Line 5 – 20
24:  until Coordinate Descent Convergence

```

Ensure: F



شکل ۹.۳: بررسی میزان تأثیر موازی‌سازی بر عملکرد GSCD در مجموعه داده RCV1 به دلیل استفاده از لغات پرکاربرد در ماتریس سند-واژه و در نتیجه داشتن مؤلفه‌های با تعداد زیاد همسایه مشترک، موازی‌سازی تأثیر ناچیزی بر سرعت عملکرد روش دارد. در مجموعه داده Yahoo News نیز مشاهده می‌شود موازی‌سازی با تعداد نخ اجرایی بالا می‌تواند تأثیر به سزایی بر روی سرعت بهینه‌سازی داشته باشد.

نتایج عملی میزان تأثیر موازی‌سازی بر عملکرد GSCD

برای بررسی میزان تأثیر موازی‌سازی بر عملکرد GSCD همان مجموعه‌داده‌های Yahoo News و RCV1 استفاده شدند و سرعت کاهش تابع هدف در هر تکرار سنجیده شد. بعد فضای ویژگی پنهان نیز برای هر دو مجموعه داده برابر ۲۰ در نظر گرفته شد. همچنین نتایج با اجرا بر روی یک کامپیوتر با ۸-نخ اجرایی^{۶۱} انجام گرفت. شکل ۹.۳ نتایج را نمایش می‌دهد.

تأثیر بهینه‌سازی بر روی مجموعه داده RCV1 ناچیز است. اگر چه دلیل این موضوع برای ما به طور دقیق مشخص نیست، اما می‌توان این حدس را زد که از آنجا که این مجموعه‌داده یک مجموعه داده متنی پیش‌پردازش شده است، به احتمال زیاد لغات با فرکانس بیشتر در اسناد مختلف در این مجموعه داده گنجانده شده‌اند. این موضوع موجب وجود تعداد زیادی مؤلفه با همسایه مشترک می‌شود و به دنبال آن تأثیر موازی‌سازی را کمتر می‌کند.

نتایج بر روی مجموعه‌داده Yahoo News نیز موید این موضوع است که موازی‌سازی با تعداد نخ اجرایی بالا می‌تواند تأثیر به سزایی بر روی سرعت بهینه‌سازی داشته باشد؛ اگرچه اجرای با بیش از ۸ نخ اجرایی برای ما به دلیل محدودیت سخت‌افزاری ممکن نبود، اما با توجه به شکل ۹.۳، با اجرای ۸ نخ اجرایی تغییر قابل لمسی در سرعت کاهش تابع هدف مشاهده می‌شود.

فصل ۴

جمع‌بندی و نتیجه گیری

۱۰۴ بازسازی شبکه تحت نمونه‌برداری فشرده

در فصل دوم، برای نخستین بار روشنی برای مسئله بازسازی شبکه بر اساس چارچوب نمونه‌برداری فشرده ارائه شد. برای فرموله‌سازی بهتر، این روش در حوزه شبکه‌های اطلاعاتی ارائه شد. با استفاده از مدل احتمالاتی آبشارهای اطلاعاتی، مسئله بازسازی شبکه به صورت حل یک دستگاه خطی مدل شد. از آن جایی که در اکثر اوقات این دستگاه زیرمعین بوده و دارای پاسخی تنک است، از تئوری و روش‌های نمونه‌برداری فشرده برای یافتن پاسخ استفاده شد. نتایج نشان می‌دهد که روش ارائه شده در حوزه شبکه‌های اطلاعاتی دارای عملکردی قابل قبول است. همچنین روش ارائه شده در دیگر حوزه‌ها نیز می‌تواند به کار گرفته شود.

هر چند روش ارائه شده در آزمایش‌ها عملکرد بهتری از روش‌های پیشین داشته است، بهبود پیچیدگی زمانی و فضایی این روش از کارهای آینده مهم در راستای استفاده از نمونه‌برداری فشرده در تحلیل شبکه‌ها است. همچنین استفاده از فرایندهای خارجی دیگر در روش ارائه شده، می‌تواند کارایی این روش را در حوزه‌های دیگر به نمایش بگذارد.

۲۰۵ پیش‌بینی لینک مقیاس‌پذیر با استفاده از فاکتورگیری ماتریس

در فصل سوم، مسئله پیش‌بینی لینک را معرفی کرده و برای پیش‌بینی لینک ساختاری، یک روش مقیاس‌پذیر با استفاده از روش‌های فاکتورگیری ماتریس ارائه دادیم. برای این منظور، بر خلاف کارهای گذشته در پیش‌بینی لینک از مدل تولید کننده پواسون برای مدل‌سازی آماری شبکه استفاده

کردیم و نشان دادیم استفاده از این مدل به جای مدل‌های کلاسیکی چون برنولی-لジستیک، منجر به استنتاج سریع پارامترها مدل آماری می‌شود. طبق آزمایش‌هایی که در کارهای اخیر انجام شده و همچنین شواهد طبیعی برای استفاده از مدل‌های پواسون در شکل‌گیری شبکه‌ها و نیز نتایج این پژوهش، به نظر می‌رسد مدل‌سازی پواسون می‌تواند در تحلیل شبکه‌ها به صورت کارا استفاده شود.

نتایج شبیه‌سازی روش پیشنهادی برای پیش‌بینی لینک نشان می‌دهد روش ارائه شده توانسته است از روش‌های باناظر و بدون ناظر رقیب دقیق‌تر عمل کرده و مقیاس‌پذیری بسیار بهتری را داشته باشد.

در ادامه فصل سوم، روش پیشنهادی برای پیش‌بینی لینک در قالب کاربرد فاکتورگیری و تقریب ماتریس با استفاده از روش‌های کاهش بلوکی مولفه‌ای نیز بسط داده شد. روش پیشنهادی این بار با نحوه انتخاب پارامتر حریصانه ارائه شد. علاوه بر تحلیل رسمی پیچیدگی، نتایج شبیه‌سازی حاکی از برتری روش ارائه شده در سرعت و مقیاس‌پذیری در ماتریس‌های تنک نسبت به روش‌های اخیر است.

در ادامه برای افزایش مقیاس‌پذیری، نسخه موازی روش حریصانه نیز پیشنهاد و نتایج حاصل از اجرای آن در محیط موازی ارائه شد. روش‌های حریصانه و موازی یاد شده، نخستین روش‌های این چنینی در زمینه مطالعه فاکتورگیری ماتریس تحت تابع خطای KL-Divergence تعیین یافته هستند.

همچنین روش بهینه‌سازی حریصانه برای شرایط مسئله پیش‌بینی لینک نیز بررسی شد و یک روش تقریبی حریصانه برای این مسئله ارائه شد. هر چند روش حریصانه کارایی بهتری نسبت به روش اولیه ندارد، اما می‌تواند شروع خوبی برای ارائه الگوریتم‌های موازی و یا حریصانه دقیق‌تر باشد.

مسیرهای زیادی در پژوهش‌های آینده می‌تواند دنبال شود. وجود روش‌های بهتر کاهش مولفه بلوکی، مانند روش‌های کاهش بلوکی مولفه‌ای وفقی^۱ می‌تواند مقیاس‌پذیری و کیفیت پاسخ‌های ارائه شده را بهبود ببخشد. همچنین یکی از مهم‌ترین شاخه‌ها برای بهبود روش‌های کنونی، طراحی و پیاده‌سازی الگوریتم‌های بهینه‌سازی تحت بسترهای توزیع شده مانند Hadoop یا CUDA است که می‌توانند کارایی روش‌ها را در حل مسائل چند برابر کنند.

كتاب نامه

- [1] D. Easley and J. Kleinberg, *Networks, Crowds, and Markets: Reasoning About a Highly Connected World.* Cambridge University Press, 2010.
- [2] Y.-X. Wang and Y.-J. Zhang, “Nonnegative matrix factorization: A comprehensive review,” *Knowledge and Data Engineering, IEEE Transactions on*, vol. 25, no. 6, pp. 1336–1353, 2013.
- [3] M. Newman, *Networks: An Introduction.* New York, NY, USA: Oxford University Press, Inc., 2010.
- [4] D. Bu, Y. Zhao, L. Cai, H. Xue, X. Zhu, H. Lu, J. Zhang, S. Sun, L. Ling, N. Zhang, G. Li, and R. Chen, “Topological structure analysis of the protein-protein interaction network in budding yeast,” *Nucleic Acids Research*, vol. 31, pp. 2443–2450, 2003.
- [5] J. J. McAuley and J. Leskovec, “Learning to discover social circles in ego networks,” in *NIPS*, 2012, pp. 548–556.
- [6] J. Leskovec, J. M. Kleinberg, and C. Faloutsos, “Graph evolution: Densification and shrinking diameters,” *Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, 2007.
- [7] M. Fire, R. Puzis, and Y. Elovici, “Organizational mining using online social networks,” *ArXiv Preprint*, 2012.
- [8] M. Ripeanu, A. Iamnitchi, and I. T. Foster, “Mapping the gnutella network,” *IEEE Internet Computing*, vol. 6, no. 1, pp. 50–57, 2002.
- [9] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, “Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters.” *Internet Mathematics*, vol. 6, no. 1, pp. 29–123, 2009.
- [10] J. Yang and J. Leskovec, “Defining and evaluating network communities based on ground-truth,” in *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, ser. MDS ’12. New York, NY, USA: ACM, 2012, pp. 3:1–3:8.
- [11] “<http://www.facebook.com>.”
- [12] J. L. Schafer and J. W. Graham, “Missing data: Our view of the state of the art.” *Psychological Methods*, vol. 7, no. 2, pp. 147–177, 2002.
- [13] C. T. Butts, “Network inference, error, and informant (in)accuracy: a Bayesian approach,” *Social Networks*, vol. 25, no. 2, pp. 103–140, May 2003.

- [14] C. von Mering, R. Krause, B. Snel, M. Cornell, S. G. Oliver, S. Fields, and P. Bork, “Comparative assessment of large-scale data sets of protein-protein interactions,” *Nature*, vol. 417, no. 6887, pp. 399–403, May 2002.
- [15] G. Kossinets, “Effects of missing data in social networks,” *Social Networks*, vol. 28, no. 3, pp. 247–268, Jul. 2006.
- [16] H. Lee, D. S. Lee, H. Kang, B.-N. Kim, and M. K. Chung, “Sparse brain network recovery under compressed sensing,” *IEEE Transactions on Medical Imaging*, vol. 30, no. 5, pp. 1154–1165, 2011.
- [17] P. Siyari, H. R. Rabiee, M. Salehi, and M. E. Mehdiabadi, “Network reconstruction under compressive sensing,” in *Proceedings of International Conference on Social Informatics*. IEEE Computer Society, 2012, pp. 19–25.
- [18] ——, “Network reconstruction under compressive sensing,” *ASE Human Journal*, vol. 1, no. 3, pp. 130–143, 2012.
- [19] E. J. Candes and T. Tao, “Decoding by Linear Programming,” *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.
- [20] E. J. Candes and M. B. Wakin, “An Introduction To Compressive Sampling,” *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 21–30, Mar. 2008.
- [21] E. J. Candes, J. K. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, Aug. 2006.
- [22] D. L. Donoho, “High-dimensional centrally symmetric polytopes with neighborliness proportional to dimension.” *Discrete and Computational Geometry*, vol. 35, no. 4, pp. 617–652, 2006.
- [23] ——, “Compressed sensing,” *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [24] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society (Series B)*, vol. 58, pp. 267–288, 1996.
- [25] E. J. Candes, M. Rudelson, T. Tao, and R. Vershynin, “Error Correction via Linear Programming,” *Foundations of Computer Science, Annual IEEE Symposium on*, pp. 295–308, 2005.
- [26] J. A. Tropp and A. C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [27] E. J. Candes, “The restricted isometry property and its implications for compressed sensing,” *Comptes Rendus Mathematique*, vol. 346, no. 9-10, pp. 589–592, May 2008.
- [28] A. C. Sankaranarayanan, P. K. Turaga, R. Chellappa, and R. G. Baraniuk, “Compressive acquisition of dynamic scenes,” *CoRR*, vol. abs/1201.4895, 2012.

- [29] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, "Single-Pixel Imaging via Compressive Sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, Mar. 2008.
- [30] M. B. Wakin, J. N. Laska, M. F. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. F. Kelly, and R. G. Baraniuk, "An architecture for compressive imaging," in *in IEEE International Conference on Image Processing (ICIP)*, 2006, pp. 1273–1276.
- [31] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, "Compressive wireless sensing," in *Proceedings of the 5th international conference on Information processing in sensor networks*, ser. IPSN '06. New York, NY, USA: ACM, 2006, pp. 134–142.
- [32] J. J. Meng, H. Li, and Z. Han, "Sparse event detection in wireless sensor networks using compressive sensing," in *Proceedings of 47th Conference on Information Sciences and Systems, CISS '09*. IEEE, 2009, pp. 181–185.
- [33] J. Haupt, W. U. Bajwa, M. Rabbat, and R. Nowak, "Compressed Sensing for Networked Data," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 92–101, Mar. 2008.
- [34] W. Xu, E. Mallada, and A. Tang, "Compressive sensing over graphs," *CoRR*, vol. abs/1008.0919, 2010.
- [35] H. Mahyar, H. R. Rabiee, Z. S. Hashemifar, and P. Siyari, "UCS-WN: An unbiased compressive sensing framework for weighted networks," in *Proceedings of 47th Conference on Information Sciences and Systems, CISS '13*, 2013.
- [36] H. Mahyar, H. R. Rabiee, and Z. S. Hashemifar, "UCS-NT: An unbiased compressive sensing framework for network tomography," in *Proceedings of 38th International Conference on Acoustics, Speech, and Signal Processing, ICASSP '13*, 2013.
- [37] Y. Lu, A. Montanari, B. Prabhakar, S. Dharmapurikar, and A. Kabbani, "Counter braids: a novel counter architecture for per-flow measurement," in *Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, ser. SIGMETRICS '08. New York, NY, USA: ACM, 2008, pp. 121–132.
- [38] Y. Zhang, M. Roughan, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices," in *SIGCOMM*, 2009, pp. 267–278.
- [39] R. Gaeta, M. Grangetto, and M. Sereno, "Local access to sparse and large global information in p2p networks: A case for compressive sensing," in *Peer-to-Peer Computing*. IEEE, 2010, pp. 1–10.
- [40] W.-X. Wang, Y.-C. Lai, C. Grebogi, and J. Ye, "Network Reconstruction Based on Evolutionary-Game Data via Compressive Sensing," *Physical Review X*, vol. 1, no. 2, p. 021021, Oct. 2011.
- [41] M. Gomez Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '10. New York, NY, USA: ACM, 2010, pp. 1019–1028.

- [42] D. Kempe, J. Kleinberg, and E. Tardos, “Maximizing the spread of influence through a social network,” in *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD ’03. New York, NY, USA: ACM, 2003, pp. 137–146.
- [43] J. Leskovec, M. McGlohon, C. Faloutsos, N. S. Glance, and M. Hurst, “Patterns of Cascading Behavior in Large Blog Graphs,” in *Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA*, 2007.
- [44] M. Eslami, H. R. Rabiee, and M. Salehi, “Dne: A method for extracting cascaded diffusion networks from social networks.” in *SocialCom/PASSAT*. IEEE, 2011, pp. 41–48.
- [45] J. Edmonds, “Optimum branchings,” *Journal of Research of the National Bureau of Standards*, vol. 71B, pp. 233–240, 1967.
- [46] P. Erdos and A. Renyi, “On the evolution of random graphs,” *Publ. Math. Inst. Hung. Acad. Sci.*, vol. 5, pp. 17–61, 1960.
- [47] A.-L. Barabási and R. Albert, “Emergence of Scaling in Random Networks,” *Science*, vol. 286, no. 5439, pp. 509–512, Oct. 1999.
- [48] D. J. Watts and S. H. Strogatz, “Collective dynamics of small-world networks,” *Nature*, vol. 393, no. 6684, pp. 440–442, June 1998.
- [49] J. Leskovec and C. Faloutsos, “Scalable modeling of real graphs using kronecker multiplication,” in *Proceedings of the 24th international conference on Machine learning*, ser. ICML ’07. New York, NY, USA: ACM, 2007, pp. 497–504.
- [50] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002.
- [51] V. Colizza, R. Pastor-Satorras, and A. Vespignani, “Reaction–diffusion processes and metapopulation models in heterogeneous networks,” *Nat Phys*, vol. 3, pp. 276–282, Jan. 2007.
- [52] A. K. Menon and C. Elkan, “Link prediction via matrix factorization,” in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, vol. 6912. Springer Berlin Heidelberg, 2011, pp. 437–452.
- [53] E. J. Candes and B. Recht, “Exact matrix completion via convex optimization,” *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, Dec. 2009.
- [54] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *Computer*, vol. 42, no. 8, pp. 30–37, 2009.
- [55] “<http://netflixprize.com/>”
- [56] R. Salakhutdinov and A. Mnih, “Probabilistic matrix factorization,” in *NIPS*, 2008, pp. 1257–1264.

- [57] R. Meka, P. Jain, and I. S. Dhillon, “Matrix completion from power-law distributed samples.” in *NIPS*, 2009, pp. 1258–1266.
- [58] C.-J. Hsieh, K.-Y. Chiang, and I. S. Dhillon, “Low rank modeling of signed networks.” in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, Q. Yang, D. Agarwal, and J. Pei, Eds. ACM, 2012, pp. 507–515.
- [59] A. Jalali, Y. Chen, S. Sanghavi, and H. Xu, “Clustering partially observed graphs via convex optimization,” in *Proceedings of the 28th International Conference on Machine Learning*, ser. ICML ’11. Omnipress, 2011, pp. 1001–1008.
- [60] D. Liben-Nowell and J. Kleinberg, “The link-prediction problem for social networks,” *J. Am. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, May 2007.
- [61] R. Raymond and H. Kashima, “Fast and scalable algorithms for semi-supervised link prediction on static and dynamic graphs,” in *Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part III*, ser. ECML PKDD’10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 131–147.
- [62] T. A. B. Snijders and K. Nowicki, “Estimation and Prediction for Stochastic Blockmodels for Graphs with Latent Block Structure,” *Journal of Classification*, vol. 14, no. 1, pp. 75–100, Jan. 1997.
- [63] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, “Mixed membership stochastic blockmodels,” *Journal of Machine Learning Research*, vol. 9, pp. 1981–2014, Jun. 2008.
- [64] T. Fawcett, “An introduction to roc analysis,” *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, Jun. 2006.
- [65] W.-J. Li, D.-Y. Yeung, and Z. Zhang, “Generalized latent factor models for social network analysis,” in *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Two*, ser. IJCAI’11. AAAI Press, 2011, pp. 1705–1710.
- [66] S. Gao, L. Denoyer, P. Gallinari, and J. Guo, “Latent factor blockmodel for modelling relational data,” in *Proceedings of the 35th European conference on Advances in Information Retrieval*, ser. ECIR’13. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 447–458.
- [67] K. Lange, D. R. Hunter, and I. Yang, “Optimization Transfer Using Surrogate Objective Functions,” *Journal of Computational and Graphical Statistics*, vol. 9, no. 1, 2000.
- [68] P. D. Hoff, “Multiplicative latent factor models for description and prediction of social networks,” *Computational and Mathematical Organization Theory*, vol. 15, no. 4, pp. 261–272, Dec. 2009.
- [69] K. Miller, T. Griffiths, and M. Jordan, “Nonparametric Latent Feature Models for Link Prediction,” in *NIPS*, 2009, pp. 1276–1284.

- [70] K. Palla, D. A. Knowles, and Z. Ghahramani, “An infinite latent attribute model for network data,” in *29th International Conference on Machine Learning*, ser. ICML ’12, Edinburgh, Scotland, June 2012.
- [71] M. Kim and J. Leskovec, “The network completion problem: Inferring missing nodes and edges in networks,” in *SIAM International Conference on Data Mining, SDM ’11*, 2011, pp. 47–58.
- [72] T. H. McCormick, M. J. Salganik, and T. Zheng, “How Many People Do You Know?: Efficiently Estimating Personal Network Size,” *Journal of the American Statistical Association*, vol. 105, no. 489, pp. 59–70, Mar. 2010.
- [73] S. Hanneke and E. P. Xing, “Network completion and survey sampling,” *Journal of Machine Learning Research - Proceedings Track*, vol. 5, pp. 209–215, 2009.
- [74] A. K. Menon and C. Elkan, “A log-linear model with latent features for dyadic prediction,” in *Proceedings of the 2010 IEEE International Conference on Data Mining*, ser. ICDM ’10. IEEE Computer Society, 2010, pp. 364–373.
- [75] C. A. Cameron and P. K. Trivedi, *Regression Analysis of Count Data (Econometric Society Monographs)*. Cambridge, United Kingdom: Cambridge University Press, Sep. 1998.
- [76] I. Psorakis, S. J. Roberts, M. Ebden, and B. Sheldon, “Overlapping community detection using bayesian nonnegative matrix factorization.” *Physical Review E*, vol. 83, no. 6, 2011.
- [77] D. Wind and M. Morup, “Link prediction in weighted networks,” in *Machine Learning for Signal Processing (MLSP), 2012 IEEE International Workshop on*, 2012, pp. 1–6.
- [78] D. D. Lee and H. S. Seung, “Algorithms for non-negative matrix factorization,” in *NIPS*, 2000, pp. 556–562.
- [79] C.-J. Hsieh and I. S. Dhillon, “Fast coordinate descent methods with variable selection for non-negative matrix factorization,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD ’11. ACM, 2011, pp. 1064–1072.
- [80] J. Yang and J. Leskovec, “Overlapping community detection at scale: a nonnegative matrix factorization approach,” in *Proceedings of the sixth ACM international conference on Web search and data mining*, ser. WSDM ’13. ACM, 2013, pp. 587–596.
- [81] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum Likelihood from Incomplete Data via the EM Algorithm,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.
- [82] R. N. Lichtenwalter and N. V. Chawla, “Lpmade: Link prediction made easy,” *Journal of Machine Learning Research*, vol. 999888, pp. 2489–2492, Nov. 2011.
- [83] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

- [84] D. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.
- [85] I. S. Dhillon, P. D. Ravikumar, and A. Tewari, “Nearest neighbor based greedy coordinate descent.” in *NIPS*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. C. N. Pereira, and K. Q. Weinberger, Eds., 2011, pp. 2160–2168.
- [86] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. The MIT Press, 2001.
- [87] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, “Rcv1: A new benchmark collection for text categorization research,” *Journal of Machine Learning Research*, vol. 5, no. Apr, pp. 361–397, 2004.
- [88] C.-J. Lin, “Projected gradient methods for nonnegative matrix factorization,” *Neural Comput.*, vol. 19, no. 10, pp. 2756–2779, Oct. 2007.

واژه‌نامه

Cascade Information	آبشار اطلاعاتی
Transmission Probability	احتمال گذر
Inference	استنتاج
Cross Validation	اعتبارسنجی چندگانه
Information Diffusion	انتشار اطلاعات
Label and Link Propagation	انتشار برچسب و لینک
Vector Norm	اندازه برداری
Latent Semantic Indexing	اندیس‌سازی معنایی پنهان
Network Reconstruction	بازسازی شبکه
Evolutionary Game	بازی تکاملی
Unsupervised	بدون ناظر
Expectation Maximization	برآورده‌بیشینه‌سازی
Penalized Linear Regression	برازش خطی با جرمیه
Logistic Regression	برازش لجیستیک
Linear Programming	برنامه‌ریزی خطی
Semi-definite Programming	برنامه‌ریزی نیمه‌معین
Over Fitting	بیش‌برازش
Temporal Link Prediction	پیش‌بینی لینک زمانی
Structural Link Prediction	پیش‌بینی لینک ساختاری
Link Function	تابع پیوند
Loss Function	تابع خطأ
Objective Function	تابع هدف
Singular Value Decomposition	تجزیه مقادیر تکین
Strict Convexity	تحدیب صریح
Latent Discriminant Analysis	تحلیل مولفه جداساز پنهان
Principal Component Analysis	تحلیل مولفه‌های اصلی
Independent Component Analysis	تحلیل مولفه‌های مستقل
Local and Global Balance	تعادل محلی و عمومی
Network Completion	تکمیل شبکه
Sparse	تنک
Power Law Distribution	توزیع توانی
Multinomial Distribution	توزیع چندجمله‌ای
Dirichlet Distribution	توزیع دریکله
Contingency Table	جدول احتمال وقوع
Basis Pursuit	جستجوی پایه

Matching Pursuit	جستجوی تطابق
Backtracking Line Search	جستجوی خطی بازگشتی
Cyclic	چرخشی
Maximum Spanning Tree	درخت پوشای بیشینه
Precision	دقت
Classification	دسته‌بندی
Rank	رتبه
Hit Time	زمان برخورد
Under-determined	زیرمعین
Payoff	سود
Online Social Networks	شبکه‌های اجتماعی برخط
Diffusion Networks	شبکه‌های انتشار
Quasi-Newton	شبه‌نیوتن
Accuracy	صحت
Condition Number	عدد حالت
Matrix Factorization	فاکتورگیری ماتریس
Recall	فراخوانی
Indian Buffet Process	فرایند بوفه هندی
Collaborative Filtering	فیلترینگ اشتراکی
Vector Quantization	قسمت‌بندی برداری
Dimensionality Reduction	کاهش بعد
Gradient Descent	کاهش گرادیان
Proximal Gradient Descent	کاهش گرادیان مبدایی
Block Coordinate Descent	کاهش مولفه‌ای بلوکی
Measurement Matrix	ماتریس اندازه‌گیری
Document-Term Matrix	ماتریس سند-واژه
Maximum A Posteriori	ماکزیمم احتمال پسین
Maximum Likelihood	ماکزیمم درستنمایی
Gradient Residue	مانده گرادیان
Independent Cascade Model	مدل آبشار مستقل
Mixture Model	مدل اختلاطی
Generative Model	مدل تولیدکننده
Stochastic Block Modeling	مدل‌سازی بلوکی اتفاقی
Latent Feature Models	مدل‌های ویژگی پنهان
Partial Observations	مشاهدات جزئی
F-measure	معیار F
Harmonic Mean	میانگین موزون
Nuclear Norm	نرم هسته‌ای
Supervised	ناظارتی
Peer-to-peer	نظیر به نظیر
Stationary Point	نقطه ایستا
Survey Sampling	نمونه‌برداری بازدیدی
Compressed Sensing	نمونه‌برداری فشرده
Semi-supervised	نیمه‌ناظارتی
Adaptive	وقتی

Heap

Sparse Partial Correlation

Regularization

هرم
همبستگی جزئی تنک
هموارسازی

پیوست آ: مقالات استخراج شده

Network Reconstruction under Compressive Sensing

ارجاع به مقاله:

P. Siyari, H. R. Rabiee, M. Salehi, and M. E. Mehdiabadi, “Network reconstruction under compressive sensing,” in Proceedings of International Conference on Social Informatics. IEEE Computer Society, pp. 19-25, 2012.

Abstract: Many real-world systems such as the Internet, the World Wide Web, and social interactions can be modeled as networks of interacting dynamical nodes. In many cases, one encounters the situation where the pattern of the node-to-node interactions (i.e., edges) or the structure of a network is unknown. We address this issue by studying the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements?

We propose a novel framework called CS-NetRec based on Compressive Sensing (CS), a newly emerged paradigm in sparse signal recovery. The general idea of using CS is that if the presentation of information is sparse, then it can be recovered by using a few number of linear measurements. In particular, we utilize the observed data of information cascades in the context of CS for network reconstruction.

Our comprehensive empirical analysis over both synthetic and real datasets demonstrates that the proposed framework leads to an efficient and effective reconstruction. Specifically, the results demonstrate that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network). Furthermore, our framework is capable of near-perfect reconstruction of the desired network in the presence of 95% sparsity. Also, we compared the performance of our framework with one of the state-of-the-art methods for inferring the networks of diffusion, NetInf. The results suggest that our method outperforms NetInf by an average 10% improvement in the resulting F-measure.

Network Reconstruction under Compressive Sensing

Payam Siyari*, Hamid R. Rabiee†, Mostafa Salehi* and Motahareh Eslami Mehdiabadi *

Department of Computer Engineering, Sharif University of Technology

* Email: {siyari, mostafa_salehi, eslami}@ce.sharif.edu

† Email: rabiee@sharif.edu

Abstract—Many real-world systems and applications such as World Wide Web, and social interactions can be modeled as networks of interacting nodes. However, in many cases, one encounters the situation where the pattern of the node-to-node interactions (i.e., edges) or the structure of a network is unknown. We address this issue by studying the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements? We propose a novel framework called CS-NetRec based on a newly emerged paradigm in sparse signal recovery called Compressive Sensing (CS). The results demonstrate that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network). Furthermore, our framework is capable of near-perfect reconstruction of the desired network in presence of 95% sparsity. In addition, we compared the performance of our framework with NetInf; one of the state-of-the-art methods in inferring the networks of diffusion. The results suggest that the proposed method outperforms NetInf by an average of 10% improvement based on the F-measure.

I. INTRODUCTION

In large scale networked systems, which arise in many scientific and engineering applications, the node-to-node interactions or the network structure is not usually known. In such situations, it is important to propose an efficient method to reconstruct the network structure based on partial observations. This issue is known as the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements? This problem is encountered in many real-world situations. However, it is still a challenging issue to be addressed by introduction of new frameworks. For example, in biological systems, although there has been a great effort in improving the technologies to uncover the Protein interaction data, there are several reports of their inaccuracies [26]. In the analysis of social networks, particularly online social networks, the existence of missing data is almost inevitable due to several reasons, e.g. security, user privacy, data aggregation overhead, etc. Analysis of such data can lead to misleading estimation of network properties [3].

In this paper, for the first time, we introduce a general framework called “CS-NetRec”(Compressive Sensing for Network Reconstruction), based on the concept of Compressive Sensing (CS); a recently emerged paradigm for efficient

sparse-signal recovery. Our motivation for using CS is that it can provide a concrete mathematical framework for the problem of network reconstruction. The basic idea in CS [6] is that in an appropriate lower dimensional representation (e.g. sparse vector, low-rank matrix, etc.), the under-sampled data of a signal have all the information needed about that signal. This means that a signal can be reconstructed from a small set of sampled data. In many real-world situations, the existing sparsity in the network structure, helps to make this technique applicable.

In this paper, we utilize the cascade probability data from the diffusion of an arbitrary type of information throughout the desired network. We consider each information cascade as a single measurement on the network structure. Then we estimate the probability that a cascade can diffuse over the network by considering the probability of the most likely tree related to a cascade. Finally, by formation of a linear system from the diffusion process, we utilize the theory of CS in order to reconstruct the network of interest.

We evaluated the proposed framework with various configurations in terms of the dependency of its accuracy to the number of observations. The results show that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network) with F-measures above 0.5, where the number of possible edges to predict is 5 to 10 times more than the number of running cascades. In addition, we evaluated the effect of the sparsity of the network structure on the performance of our framework. In the random graph model, we observed that when about 5% of the edges are available, meaning in the presence of 95% sparsity in the graph structure, we have a near perfect recovery with F-measures above 0.9. Furthermore, we compared our framework with NetInf; one of the state-of-the-art methods for inferring the networks of diffusion. The comparison showed that for different cascade lengths, our method outperforms NetInf by an average of 9-10% improvement in terms of F-measure.

In summary, the main contributions of this paper can be stated as follows:

- Proposing a novel and general network reconstruction framework based on the rich mathematical framework of CS.
- Ability to reconstruct the underlying network without

any knowledge about the topological features of the underlying network.

II. RELATED WORK

A. Network reconstruction

The problem of reconstructing the network structure from incomplete or missing data have been studied in many different contexts and different techniques. In [18], the main problem being studied is the network completion problem where a network with missing nodes and edges is given. It is assumed that the underlying network follows the Kronecker graph model. A similar problem is also studied as the inference of missing links in the context of survey sampling in [16], and in social and biological networks in [2], [15]. Also, the problem of link prediction relates to our work in which the aim is to predict the future edges of a network. Link prediction has been studied in social [22], and biological networks [13].

The most related problem to our work is where the goal is to infer hidden underlying network on which some type of information diffuses (e.g. virus, rumor, news, etc.) [11], [14]. However, this problem is a special case of the network reconstruction problem, and we develop our general framework based on such configuration for better formulation and evaluation.

B. Compressive sensing

The developments in compressive sensing began with the seminal works in [5], [8]. The authors showed that combining the l_1 -minimization and random matrices can lead to efficient recovery of sparse vectors. Moreover, the authors showed that such concepts have strong potential to be used in many applications.

Consider the linear system:

$$y_{m \times 1} = A_{m \times n} x_{n \times 1} \quad (1)$$

Where $m \ll n$. In sparse recovery (esp. CS), the set of sparse solutions to this system are of interest. Thus, we have to add a constraint to the system so that we can limit the solution space. Specifically, we assume x is k -sparse, meaning there is at most k nonzero elements in the vector x , and $k \ll n$. In CS theory, it is stated that the optimal solution can be obtained by:

$$\min_x \|x\|_0 \quad s.t. \quad Ax = y \quad (2)$$

Since solving Eq. (2) is NP-Hard, one can use l_1 -minimization instead [5], [8]:

$$\min_x \|x\|_1 \quad s.t. \quad Ax = y \quad (3)$$

Combining least squares and (3), we obtain:

$$\min_x \|x\|_1 + \|Ax - y\|_2^2. \quad (4)$$

This change in the objective function (also known as LASSO [4], [27]) makes it possible to solve the linear system, even in presence of noise or truncated values in the matrix A , and vector y .

CS has been mainly studied in the context of signal and image processing [9], [25], and its use in the area of network analysis is still in its first stages of development.

In the context of network analysis, CS has been mostly used in the field of wireless sensor networks to recover the information (e.g. the temperature of an area) based on the samples aggregated from the network sensors [1], [24]. In [30], the question of whether it is possible to quickly infer and monitor the network link characteristics from indirect end-to-end (aggregate) measurements is analyzed. In network traffic monitoring, we can mention the works in [23], [31]. In [19], the authors introduced a new approach based on the penalized linear regression to estimate sparse partial correlation between different regions of interests of the brain network, and by employing the CS theory they were able to have a successful recovery of the brain network. However, the proposed approach in that work can not be generalized in other areas and can only be used in the context of brain networks. In [28], the authors articulate a general method for addressing the problem of how to uncover the network topology using evolutionary-game data based on compressive sensing.

III. PROBLEM FORMULATION

A. Problem statement

Consider the static directed network $G(V, E)$, with $|V| = n$ nodes and set of the edges E . We assume that we are totally aware of which nodes exist in the network and have no information about the edges. To reconstruct the network, we must predict the value of the elements of the adjacency matrix. Hence, we are looking for G^* where $\|G^* - G\| \leq \epsilon$. We assume that there is an external process that can be run on the network and has the following features:

- 1) It can run several times over the network
- 2) Its output can be considered as the linear combination of the edges of the network and a particular measure defined on each edge.

From now on, we consider the information diffusion as the external process in which any type of information or disease (e.g. news headlines, virus, rumor, etc.) diffuses over an underlying network. As an example, Figure 1(a) shows a news blogs network, where the nodes represent the blogs. The diffusion process on such network can be considered as the coverage of a particular news in different blogs with different time stamps. However, the network over which propagations take place is usually unknown and unobserved [14]. The goal now is to reconstruct the unknown network over which news originally propagated so that we know each node's information sources. Obviously, news propagation satisfies the first property that we would like the external process to have. To show that the diffusion process satisfies the second property of an external process, we mention some preliminaries related to the information networks and then in section IV, we show the second property for this process.

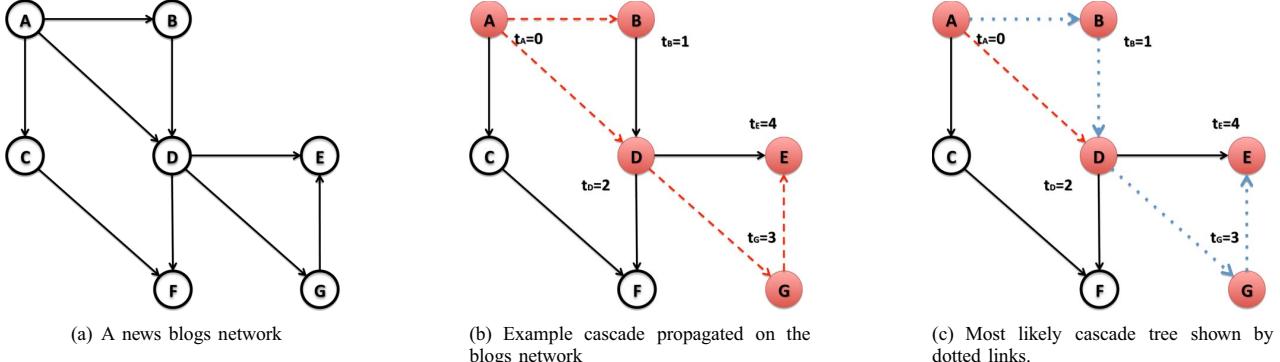


Fig. 1. An example of information diffusion on a news blogs network. (a) A news blogs underlying network (i.e. news coverage sequence) propagated on the blogs network. Each node hit by the cascade is shown in different color and is labeled by a time stamp which shows the time of its news coverage. The cascade caused by the propagation of a particular news can be represented as: *News Cascade* = < $(\phi, A, t_A), (A, B, t_B), (A, D, t_D), (D, G, t_G), (G, E, t_E)$ >. (c) The most likely cascade tree (shown by dotted links) as an approximation for the hidden real cascade tree. It can be seen that the most likely cascade tree is not necessarily identical to the real cascade tree in (b).

B. Information networks and the cascading behavior

Each diffusion of the information throughout the network, creates an information cascade. In other words, an information cascade can be considered as a sequence of nodes, that is being visited by the diffusion process.

A cascade is a set of triples $(u, v, t_v)_c$, which means that cascade c reaches node v at time t_v by spreading from node u (by propagating over the edge (u, v)) [14]. We denote the fact that the cascade initially starts from some active node v at time t_v as $(\phi, v, t_v)_c$. We assume the only data we can obtain from the cascades is the time that a cascade has reached a node. Hit time can be described as the time t_v when node v is being included in the cascade c . Now, given such data about node hit times for a number of different cascades, we aim to recover the unobserved directed network G ; the network over which the cascades originally spread.

We consider the independent cascade model [17] in which it only matters when the first neighbor of v spreads the cascade. Thus, the structure of a cascade c created by the diffusion process, is fully described by a directed tree T , contained in the directed graph G .

C. Maximum probability cascade tree

Now, we aim to give an approximation for the directed tree T that is related to the cascade c . We use the probabilistic model of how cascades spread over the edges of the network.

Define the probability $P_c(u, v)$ as the conditional probability of observing cascade c spreading from u to v [14]. Intuitively, the probability of propagation $P_c(u, v)$ between a pair of nodes u and v in the cascade c decreases with as the difference of their hit times increases. In the news blogs networks, an old news most probably does not provide any information for the blog. For simplicity, we consider $P_c(u, v)$ to follow the well-known exponential cascade transmission distribution [21]:

$$P_c(u, v) = P_c(\Delta_{u,v}) = e^{-\frac{\Delta_{u,v}}{\alpha}} \quad (5)$$

Where $\Delta_{u,v} = t_v - t_u$, and α is the adjustment parameter.

A cascade stops with the probability $(1 - \beta)$, and continues over an edge with the probability of β . The likelihood of a cascade spreading in a given tree pattern T is calculated as [14]:

$$P(c|T) = \prod_{u,v \in E_T} \beta P_c(u, v) \prod_{u \in V_T, (u,x) \in E \setminus E_T} (1 - \beta) \quad (6)$$

Where $T = (V_T, E_T)$ is a tree.

Also the probability that a cascade c can occur in the graph G is defined as [14]:

$$P(c|G) = \sum_{T \in \tau_c(G)} P(c|T) P(T|G) \quad (7)$$

Where $\tau_c(G)$ is the set of all the directed connected spanning trees on a subgraph of G , induced by the nodes that have been visited by cascade c . In case T is inconsistent with the observed data, then $P(c|T) = 0$.

As there exist exponential number of possible trees, we use an approximation for the likelihood of a single cascade by considering only the most likely tree instead of all possible propagation trees:

$$P(c|G) = \max_{T \in \tau_c(G)} P(c|T) P(T|G) \quad (8)$$

We consider all possible trees to be equiprobable to occur. In other words, $P(T|G) = \frac{1}{|\tau_c(G)|}$. The maximum probability tree can be found by choosing, for each node v , an incoming edge (u, v) with maximum probability [10]. Figure 1(c) shows the most probable cascade tree for the news cascade in Figure 1(b).

IV. PROPOSED FRAMEWORK: CS-NETREC

Since we can run several cascades on the desired network, we run several cascades and use the cascades' probability data to reconstruct the network of interest. In particular, we form

a linear system with the edges as the unknown vector and try to find a solution for this system (i.e. a reconstruction for the network). More details can be seen in Algorithm IV.1.

Assuming T_c^* is the most likely tree corresponding to the cascade c , we hope that all the edges in T_c^* represent the real edges in the underlying network G , although this is not necessary.

Thus, after taking log from both sides of the Eq. (8), we can

Algorithm IV.1 CS-NetRec(C, V)

```

for each  $c \in C$  do
     $T_c^* \leftarrow$  The Most Probable Tree for  $c$ .  $\triangleright$  (Section III.C)
    for each  $(i, j) \in V \times V$  do
         $LP_c(i, j) \leftarrow \begin{cases} \log P_c(i, j) & (i, j) \in T_c^* \\ 0 & o.w. \end{cases}$   $\triangleright$  (Eq. (5))
    end for
    Calculate  $LP(c|G) = \log P(c|G)$ .  $\triangleright$  (Eq. (8))
    Form the vector  $vLP(c|T_c^*)$  from  $LP_c(i, j)$ s.  $\triangleright$  (Eq.(10))
    Add  $LP(c|G)$  as a row to  $y$ .
    Add  $vLP(c|T_c^*)$  as a row to  $A$ .
end for

 $x^* \leftarrow \arg \min_x \|x\|_1 + \|Ax - y\|_2^2$   $\triangleright$  (Section II.B)

return  $x^*$ 

```

approximate it as the inner product of two vectors:

$$LP(c|G) = vLP(c|T_c^*)^T \cdot vec(Adj(G)) \quad (9)$$

Where $LP(c|G) = \log P(c|G)$, and $vec(Adj(G))$ is the vectorized binary adjacency matrix of network G in some particular order. $vLP(c|T_c^*)$ is the vector of individual edge transmission log probabilities with nonzero elements corresponding to the edges in T_c^* . In particular, the k -th element of $vLP(c|T_c^*)$ can be shown as:

$$vLP(c|T_c^*)_k = LP_c(i, j) = \log P_c(i, j) \quad (10)$$

By this approximation, we are now able to show a single run of the diffusion process (i.e. a cascade) as the linear combination of the edges of the network and the probability measure defined on each edge. Thus, the diffusion process satisfies the second property of an external process that is mentioned earlier.

Now that the diffusion process satisfies both properties that an external process should have, we use this process, and the CS framework in order to reconstruct the network of interest.

We can model m runs of the diffusion process as the linear system: $y = Ax$, where each equation in this linear system is equal to (9), and the unknowns are the edges of the desired network.

V. EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of the proposed framework for network reconstruction. First we introduce the

synthetic and real datasets we use for the evaluation. Next, we introduce the metrics of evaluation, and finally, we present an analysis on the achieved results.

A. Settings

We consider both synthetic and real networks. We use three well-known classic models for generating directed networks, namely, the Erdos-Renyi model with 500 edges, the Small world graph model with each node being connected to 4 nearest neighbors in ring topology and the rewiring probability of 0.4, and the Barabasi-Albert model with each new node getting connected to 5 existing nodes. Each of these networks have 100 nodes. Also, we use a variant of Kronecker graph model, namely, Core-Periphery Kronecker [20], with 256 nodes which results in around 600 edges in several generations. We consider three directed real-world networks: The network of American football games between Division IA colleges during regular season fall 2000 [12] with 115 nodes and 615 edges, the neural network of the Caenorhabditis elegans worm (C.elegans) [29] with 306 nodes and 2345 edges, and the network of 500 busiest commercial airports in the USA [7] with 500 nodes and 2980 edges.

In each of the test cases, we generated 100 sets of cascades. For the cascade generation, we use the same process as NetInf, one of the state-of-the-art methods for diffusion network inference.

Moreover, to avoid the trade-off between precision and recall and to consider both, we used the F-measure metric. This metric presents the harmonic mean of both precision and recall.

We set a threshold of 0.5 for the predicted values of the vector x . We tested other values of threshold and observed no significant effect in the accuracy of our approach. Thus, we consider all elements above 0.5 as a predicted edge.

B. Parameter analysis

We analyzed the sensitivity of our approach to the parameters of the cascades and if possible fix them, so that we obtain the rest of the results based on the fixed values of the parameters. The information cascades have two parameters to diffuse throughout the network.

First, we show the sensitivity of the results to α which represents the cascades' speed. To this end, we consider the cascades with a constant β equal to 0.5. As shown in Figure 2, it seems that for all the networks, the F-measures reach consistency, when the value of α is around 3. Hence we used this fix value for α .

The length of the cascades, controlled by the parameter β , can significantly alter the results of any network reconstruction approach. We consider the cascades with a constant α equal to 1. Obviously, when one runs a number of cascades on a network with a small value of β , the cascades will be shorter. Thus, a smaller number of edges will be observed in such situations, and as a consequence, the resulted recall would be very small. Considering Figure 3, we fix β at the value of 0.5.

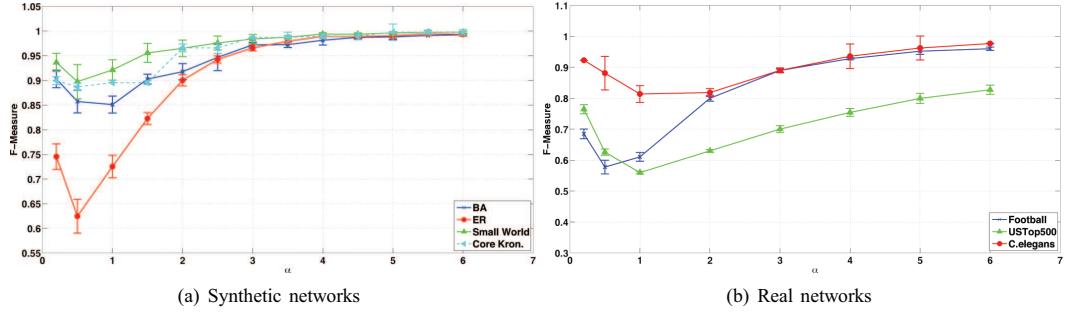


Fig. 2. α parameter analysis using CS-NetRec - It can be seen that the value $\alpha = 3$ is where the F-measures become consistent.

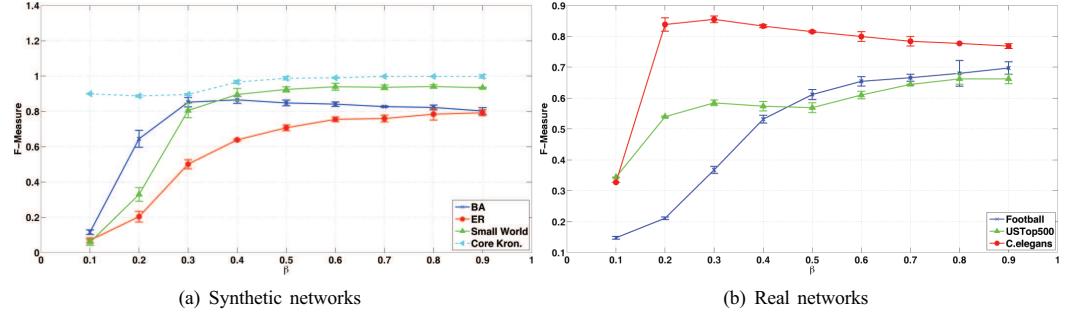


Fig. 3. β parameter analysis using CS-NetRec - The value of $\beta = 0.5$ is where the F-measures become consistent.

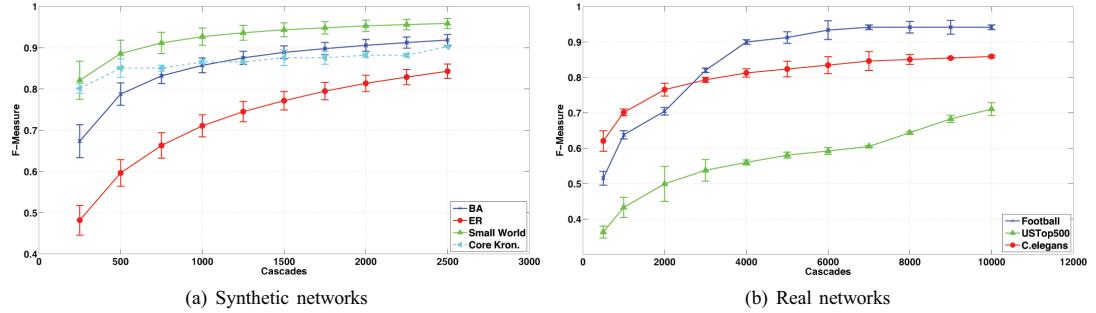


Fig. 4. Different networks cascade dependency using CS-NetRec

C. Cascade dependency

Obviously, it is almost never possible to have number of processes equal to the number of possible edges in a network, as it becomes computationally expensive. Also, increasing the number of transmitting cascades over the network leads to more accurate prediction of the network topology. We tested this scenario by running 250 to 2500 cascades on each of the synthetic networks and the Football network, where there are around 10000 possible edges in the graph. On other real datasets, we ran 250 to 10000 cascades. In C.elegans dataset there are about 93000 possible edges, and in US Top 500 dataset, this number reaches to around 250000. In all of our measurements we used F-measures for comparison purposes.

In Figure 4, particularly for Small world and BA datasets which are closer to real-world graphs than ER model, it can be observed that even on low number of cascades (e.g. half of the

number of existing edges in the graphs), we have high values of F-measure (almost all above 0.5), considering around 10000 possible edges to be predicted. Thus the results demonstrate that our framework can work accurately even on very low information from the network of interest. In the Kronecker dataset and Small world, we have above 0.8 F-measure when the number of cascades are roughly the same as the number of edges in the graph.

In C.elegans and US Top 500, with 2000 cascades we obtain around 0.75 and 0.5 F-measures while there are around 93000 and 250000 possible edges to predict in each of these datasets, respectively. Further, we observe that the F-measure in Football dataset, due to the lower number of nodes, reaches to 0.9 F for 4000 cascades. Thus, we observe that in spite of the same scale of this dataset to the synthetic ones, we need more number of measurements to reach a high F-measure

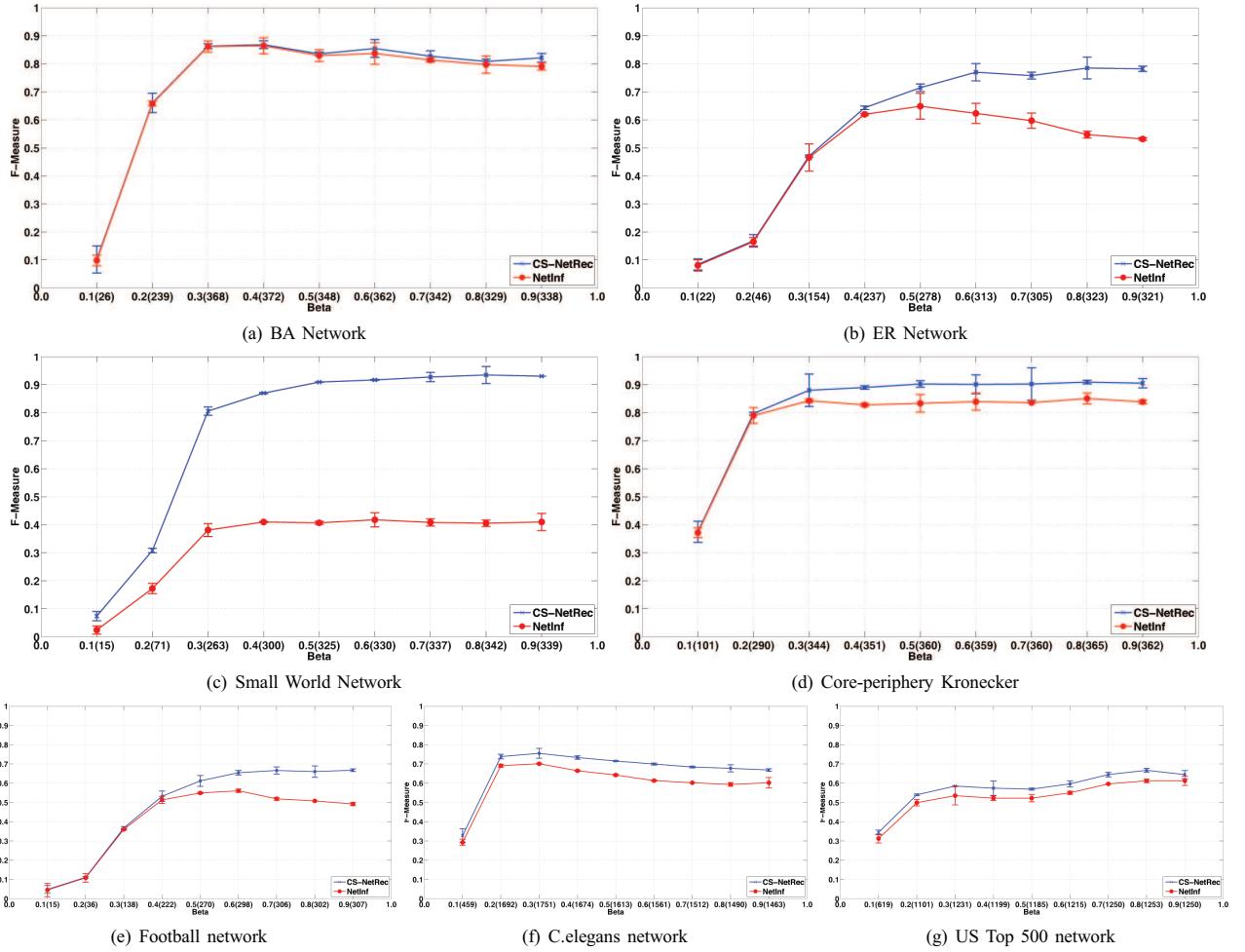


Fig. 6. Performance comparison of CS-NetRec with NetInf, considering different values of β parameter (i.e. cascade lengths) and $\alpha = 1$. (a,b,c,d) Synthetic networks, (e,f,g) Real networks. We considered 2500 cascades for BA, ER, SmallWorld and Football datasets, 8000 cascades for Kronecker graph and 10000 cascades for C.elegans and US Top 500. The numbers in the parenthesis show the number of edges recovered.

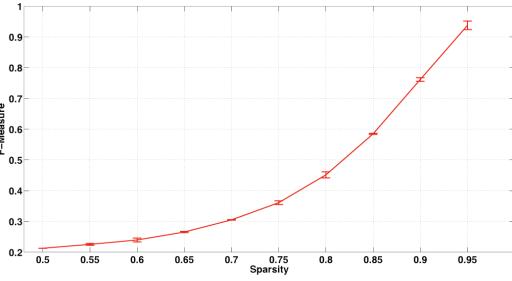


Fig. 5. The effect of sparsity of the network adjacency in ER model using CS-NetRec. The less sparsity in the unknown vector, the less accuracy in the reconstruction of the network.

value. With more cascades, we traverse more edges of the graph. Meanwhile, the transmission probability over an edge gets repeated in more equations of the linear system. Thus,

the effect of that edge on the increase of cascade probabilities can be seen more.

D. The effect of the sparsity of the network adjacency

According to the CS theory, the less sparsity in the unknown vector, the less accuracy in its recovery. To show this phenomenon in our framework, we consider several ER networks with 100 nodes and number of edges ranging from 500 to 5000. In other words, the sparsity ranges from 0.5 to 0.95. Again we consider all the F-measures to evaluate the results.

As expected, in Figure 5, we observe that the more sparsity in the same scale (i.e. number of nodes) yields to better performance in our approach. For 0.95 sparsity, we have a near perfect reconstruction. However, decreasing the sparsity to 0.9, diminishes the F-measure to below 0.8. Thus, in the same dataset, the network reconstruction with more sparsity can result in higher accuracy.

E. Performance comparison with NetInf

Here, we compare the performance of our method against one of the state-of-the-art methods for inferring networks of diffusion, namely NetInf [14].

We used the code provided by the authors and considered different values for the parameter β as we would like to evaluate the performance of methods in presence of local information (i.e. shorter cascades), and global information (i.e. longer cascades). We also set α to 1 in our tests. Given a fixed number of cascades, we ran NetInf with the number of iterations equal to the number of edges that our method can reconstruct. The results can be seen in Figure 6. It can be observed that in all test cases, the resulted F-measure in our method is higher than or equal to NetInf, and as we increase the value of β , the difference in F-measure becomes even more. In Small world, ER, and Football datasets, the difference is more clear. For $\beta = 0.9$, the improvement by our method in F-measure is about 0.5, 0.25 and 0.18 for Small world, ER and Football datasets, respectively. Overall, for different cascade lengths, our method outperforms NetInf by an average of 9-10% improvement in terms of the resulted F-measure.

VI. CONCLUSION

In this paper, we introduced a general framework for network reconstruction by using external processes on the network. By utilizing the probabilistic measures of information cascades, we formulated the problem of network reconstruction as a linear system. Since most of the time, this linear system is under-determined, we used the theory of compressive sensing as a tool to reconstruct the network of interest. By numerical experiments, we demonstrated that this framework will converge to an accurate solution. Also, the results suggest that in the context of information networks, our method outperforms the state-of-the-art method, NetInf.

For the future work, it would be interesting to use other external processes and utilize other features for the network nodes to reconstruct the network of interest using the proposed framework.

REFERENCES

- [1] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak. Compressive wireless sensing. In *Proceedings of the 5th international conference on Information processing in sensor networks, IPSN '06*, pages 134–142, New York, NY, USA, 2006. ACM.
- [2] K. Bleakley, G. Biau, and J.-P. Vert. Supervised reconstruction of biological networks with local models. In *ISMB/ECCB (Supplement of Bioinformatics)*, pages 57–65, 2007.
- [3] C. Butts. Network inference, error, and informant (in)accuracy: a Bayesian approach. *Social Networks*, 25(2):103–140, May 2003.
- [4] E. Candes, M. Rudelson, T. Tao, and R. Vershynin. Error Correction via Linear Programming. *Foundations of Computer Science, Annual IEEE Symposium on*, 0:295–308, 2005.
- [5] E. J. Candes, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure Appl. Math.*, 59(8):1207–1223, Aug. 2006.
- [6] E. J. Candes and T. Tao. Decoding by Linear Programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, Dec. 2005.
- [7] V. Colizza, R. Pastor-Satorras, and A. Vespignani. Reaction-diffusion processes and metapopulation models in heterogeneous networks. *Nat Phys*, 3:276–282, Jan. 2007.
- [8] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.
- [9] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk. Single-Pixel Imaging via Compressive Sampling. *IEEE Signal Processing Magazine*, 25(2):83–91, Mar. 2008.
- [10] J. Edmonds. Optimum branchings. *Res. Natl. Bur. Standards*, 71B:233–240, 1967.
- [11] M. Eslami, H. R. Rabiee, and M. Salehi. Dne: A method for extracting cascaded diffusion networks from social networks. In *Social-Com/PASSAT*, pages 41–48. IEEE, 2011.
- [12] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, June 2002.
- [13] D. S. Goldberg and F. P. Roth. Assessing experimentally derived interactions in a small world. *Proceedings of the National Academy of Sciences*, 100(8):4372–4376, Apr. 2003.
- [14] M. Gomez Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '10*, pages 1019–1028, New York, NY, USA, 2010. ACM.
- [15] R. Guimerà and M. Sales-Pardo. Missing and spurious interactions and the reconstruction of complex networks. *Proceedings of the National Academy of Sciences*, 106(52):22073–22078, Dec. 2009.
- [16] S. Hanneke and E. P. Xing. Network completion and survey sampling. *Journal of Machine Learning Research - Proceedings Track*, 5:209–215, 2009.
- [17] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '03*, pages 137–146, New York, NY, USA, 2003. ACM.
- [18] M. Kim and J. Leskovec. The Network Completion Problem: Inferring Missing Nodes and Edges in Networks. In *SDM*, pages 47–58. SIAM / Omnipress, 2011.
- [19] H. Lee, D. S. Lee, H. Kang, B.-N. Kim, and M. K. Chung. Sparse brain network recovery under compressed sensing. *IEEE Trans. Med. Imaging*, 30(5):1154–1165, 2011.
- [20] J. Leskovec and C. Faloutsos. Scalable modeling of real graphs using kronecker multiplication. In *Proceedings of the 24th international conference on Machine learning, ICML '07*, pages 497–504, New York, NY, USA, 2007. ACM.
- [21] J. Leskovec, M. McGlohon, C. Faloutsos, N. S. Glance, and M. Hurst. Patterns of Cascading Behavior in Large Blog Graphs. In *Proceedings of the Seventh SIAM International Conference on Data Mining, April 26–28, 2007, Minneapolis, Minnesota, USA*, 2007.
- [22] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *J. Am. Soc. Inf. Sci. Technol.*, 58(7):1019–1031, May 2007.
- [23] Y. Lu, A. Montanari, B. Prabhakar, S. Dharmapurikar, and A. Kabbani. Counter braids: a novel counter architecture for per-flow measurement. In *Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, SIGMETRICS '08*, pages 121–132, New York, NY, USA, 2008. ACM.
- [24] J. J. Meng, H. Li, and Z. Han. Sparse event detection in wireless sensor networks using compressive sensing. In *CISS*, pages 181–185. IEEE, 2009.
- [25] A. C. Sankaranarayanan, P. K. Turaga, R. Chellappa, and R. G. Baraniuk. Compressive acquisition of dynamic scenes. *CoRR*, abs/1201.4895, 2012.
- [26] J. L. Schafer and J. W. Graham. Missing data: Our view of the state of the art. *Psychological Methods*, Vol 7(2), Jun 2002, 147–177, 7(2):147–177, 2002.
- [27] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.
- [28] W.-X. Wang, Y.-C. Lai, C. Grebogi, and J. Ye. Network Reconstruction Based on Evolutionary-Game Data via Compressive Sensing. *Physical Review X*, 1(2):021021, Oct. 2011.
- [29] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, June 1998.
- [30] W. Xu, E. Mallada, and A. Tang. Compressive sensing over graphs. *CoRR*, abs/1008.0919, 2010.
- [31] Y. Zhang, M. Roughan, W. Willinger, and L. Qiu. Spatio-temporal compressive sensing and internet traffic matrices. In *SIGCOMM*, pages 267–278, 2009.

Network Reconstruction under Compressive Sensing

ارجاع به مقاله:

P. Siyari, H. R. Rabiee, M. Salehi, and M. E. Mehdiabadi, “Network reconstruction under compressive sensing,” ASE Human Journal, vol. 1, no. 3, pp. 130-143, 2012.

Abstract: Many real-world systems and applications such as World Wide Web, and social interactions can be modeled as networks of interacting dynamical nodes. However, in many cases, one encounters the situation where the pattern of the node-to-node interactions (i.e., edges) or the structure of a network is unknown. We address this issue by studying the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements? We propose a novel framework called CS-NetRec based on a newly emerged paradigm in sparse signal recovery called Compressive Sensing (CS). The general idea of using CS is that if the presentation of information is sparse, then it can be recovered by using a few number of linear measurements. In particular, we utilize the observed data of information cascades in the context of CS for network reconstruction. Our comprehensive empirical analysis over both synthetic and real datasets demonstrates that the proposed framework leads to an efficient and effective reconstruction. More specifically, the results demonstrate that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network). Furthermore, our framework is capable of near-perfect reconstruction of the desired network in presence of 95% sparsity. In addition, we compared the performance of our framework with NetInf; one of the state-of-the-art methods in inferring the networks of diffusion. The results suggest that the proposed method outperforms NetInf by an average of 10% improvement based on the F-measure.

Network Reconstruction under Compressive Sensing

Payam Siyari
Department of
Computer Engineering
Sharif University of
Technology
Email:
siyari@ce.sharif.edu

Hamid R. Rabiee
Department of
Computer Engineering
Sharif University of
Technology
Email: rabiee@sharif.edu

Mostafa Salehi
Department of
Computer Engineering
Sharif University of
Technology
Email:
mostafa.salehi@ce.sharif.edu

Motahareh Eslami
Mehdiabadi
Department of
Computer Engineering
Sharif University of
Technology
Email:
eslami@ce.sharif.edu

ABSTRACT

Many real-world systems and applications such as World Wide Web, and social interactions can be modeled as networks of interacting dynamical nodes. However, in many cases, one encounters the situation where the pattern of the node-to-node interactions (i.e., edges) or the structure of a network is unknown. We address this issue by studying the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements? We propose a novel framework called CS-NetRec based on a newly emerged paradigm in sparse signal recovery called Compressive Sensing (CS). The general idea of using CS is that if the presentation of information is sparse, then it can be recovered by using a few number of linear measurements. In particular, we utilize the observed data of information cascades in the context of CS for network reconstruction. Our comprehensive empirical analysis over both synthetic and real datasets demonstrates that the proposed framework leads to an efficient and effective reconstruction. More specifically, the results demonstrate that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network). Furthermore, our framework is capable of near-perfect reconstruction of the desired network in presence of 95% sparsity. In addition, we compared the performance of our framework with NetInf; one of the state-of-the-art methods in inferring the networks of diffusion. The results suggest that the proposed method outperforms NetInf by an average of 10% improvement based on the F-measure.

I INTRODUCTION

In many scientific and engineering applications, the systems under study can be modeled as a set of networked elements, called nodes. Usually, depending on

the domain, the interactions between these elements are shown as the edges between the nodes. In large scale networks, the node-to-node interactions or the network structure is not usually known. In such situations, it is important to propose an efficient method to reconstruct the network structure based on partial observations. This issue is known as the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements?

Network reconstruction problem is encountered in many real-world situations. However, it is still a challenging issue to be addressed by introduction of new frameworks. For example, in biological systems, although there has been a great effort in improving the technologies to uncover the Protein interaction data, there are several reports of their inaccuracies [1]. In the analysis of social networks, particularly online social networks, the existence of missing data is almost inevitable due to several reasons, e.g. security, user privacy, data aggregation overhead, etc. Analysis of such data can lead to deceptive estimation of network properties [2, 3].

An illustrative example of network reconstruction problem is provided in Figure 1. An example network is shown in Figure 1(a), and the network reconstruction problem where only the network's nodes are available is shown in Figure 1(b). Here, the goal is to reconstruct the network in Figure 1 based on the partial observations in Figures 1(c) and 1(d). In the context of information diffusion, we assume that the observations in Figures 1(c) and 1(d) are the outputs of some processes which are run over the edges of the network. Each process measures a value for each node and its output is a function of these values. Based on the node values and process outputs (and not the edges that process has been run over), we aim to propose a reconstruction scheme for the network. Depending on the domain of study, the observations can be more implicit or complex. In this

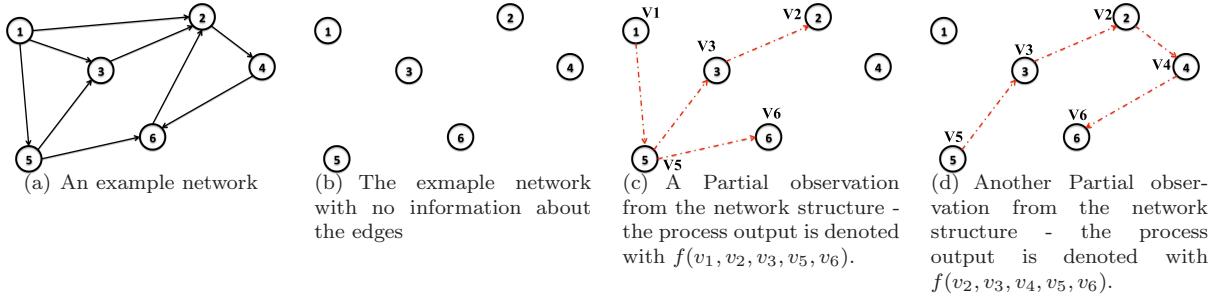


Figure 1: An example of the network reconstruction problem. (a) An example network. (b) The example network with known nodes and no information about the edges. This is the only information we have about the topology of the network in network reconstruction problem. (c,d) Example partial observations (i.e. node values and process outputs) over the graph used to reconstruct the network of interest.

paper, for the partial observations over the network, we consider information cascades (e.g. virus propagation) as the process, and node hit times (e.g. node infection times) as the node values. We will cover the concepts related to the information networks and information diffusion in Section III.

Although the problem of network reconstruction has been analyzed in various contexts in different approaches [4–9], in this paper for the first time, we introduce a general framework called “CS-NetRec” (Compressive Sensing for Network Reconstruction), based on the concept of Compressive Sensing (CS); a recently emerged paradigm for efficient sparse-signal recovery. Our motivation for using CS is that it can provide a concrete mathematical framework for the problem of network reconstruction.

The basic idea in CS [10–12] is that in an appropriate lower dimensional representation (e.g. sparse vector, low-rank matrix, etc.), the under-sampled data of a signal have all the information needed about that signal. In other words, if the presentation of information is sparse, then it can be recovered by using a few number of linear measurements. This means that a signal can be reconstructed from a small set of sampled data. In many real-world situations, the existing sparsity in the network structure, helps to make this technique applicable.

In this paper, we utilize the cascade probability data from the diffusion of an arbitrary type of information throughout the desired networked data. We consider each information cascade as a single measurement on the network structure. Then we estimate the probability that a cascade can diffuse over the network

by considering the probability of the most likely tree related to a cascade. Finally, by formation of a linear system from the diffusion process, we utilize the theory of CS in order to reconstruct the network of interest.

We evaluated the proposed framework over both synthetic and real datasets with various configurations in terms of the dependency of its accuracy to the number of observations from the network. The results demonstrate that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network) with the F-measure above 0.5 where the number of possible edges to predict is 5 to 10 times more than the number of running cascades. As another part of our experimental results, we demonstrated the effect of different cascade parameters to the accuracy of the reconstruction. In addition, we evaluated the effect of the sparsity of the network structure on the performance of our framework. In the random graph model, we observed that when about 5% of the edges are available, meaning in the presence of 95% sparsity in the graph structure, we have a near perfect recovery with the F-measure above 0.9. Furthermore, we compared our framework with NetInf; one of the state-of-the-art methods for inferring the networks of diffusion. The comparison showed that for different cascade lengths, our method outperforms NetInf by an average of 9–10% improvement in terms of F-measure.

In summary, the main contributions of this paper can be stated as follows:

- Proposing a novel and general framework based on the rich mathematical framework of CS.

- Ability to reconstruct the underlying network without any knowledge about the topological features of the underlying network.

The rest of the paper is organized as follows. In Section II, we provide some of the related works and similar problems to network reconstruction problem and a short survey on the network analysis methods that utilize CS as a part of their algorithms. In Section III, we will state the problem details and the related concepts to the information networks. In Section IV, we introduce our framework in details. The experimental evaluation of the proposed framework is presented in Section V. Finally, we conclude the paper in Section VI.

II RELATED WORK

1 NETWORK RECONSTRUCTION

The problem of reconstructing the network structure from incomplete or missing data have been studied in many different contexts and different techniques. In [8], the main problem being studied is the network completion problem where a network with missing nodes and edges is given and the goal is to estimate the unobserved part of the network. However, in this work it is assumed that the underlying network follows the Kronecker graphs model. A similar problem is also studied as the inference of missing links in the context of survey sampling in [9], and in social and biological networks in [4–7].

A related problem to network reconstruction is the matrix completion problem [13] where a data matrix with missing entries is given, and the goal is to predict its missing elements. However, there still exist many challenges to utilize the theory of matrix completion in the reconstruction of the networks. As an example, several properties of real networks (e.g. heavy-tailed degree distribution, binary valued matrix entries) are not considered in the formulation of the matrix completion algorithms.

Also, the problem of link prediction relates to our work in which the aim is to predict the future edges of a network. Link prediction has been studied in social [14], and biological networks [15].

The most related problem to our work is where the goal is to infer hidden underlying network on which some type of information diffuses (e.g. virus, rumor, news, etc.) [16, 17]. Although this problem is a special case of the network reconstruction problem, we

develop our general framework based on such configuration for better formulation and evaluation. In this work, we consider the diffusion of virus, rumor or any similar propagating information throughout the network. Some of the works in information networks try to find propagation links by using the structure and topological features of underlying network [18, 19]. However, we do not consider any topological assumptions about the underlying network, and propose a framework which is based on the rich mathematical framework of CS.

2 COMPRESSIVE SENSING

The developments in compressive sensing began with the seminal works in [20, 21]. The authors showed that combining the l_1 -minimization and random matrices can lead to efficient recovery of sparse vectors. Moreover, the authors showed that such concepts have strong potential to be used in many applications.

Consider the linear system:

$$y_{m \times 1} = A_{m \times n} x_{n \times 1} \quad (1)$$

Where $m \ll n$, and we are interested in finding a feasible value for x . Typically, the solution of a linear system can be obtained by the least squares minimization:

$$x^* = \arg \min_x \|Ax - y\|_2^2 \quad (2)$$

In sparse recovery (esp. CS), the set of sparse solutions to this system are of interest. Thus, we have to add a constraint to the system so that we can limit the solution space. Here, we add the sparsity of x as a constraint to obtain a solution. Specifically, we assume x is k -sparse, meaning there is at most k nonzero elements in the vector x , and $k \ll n$. In CS theory, it is stated that the sparsest solution can be obtained by:

$$\min_x \|x\|_0 \text{ s.t. } Ax = y \quad (3)$$

Since solving Eq. (3) is NP-Hard, one can use l_1 -minimization instead [20, 21]:

$$\min_x \|x\|_1 \text{ s.t. } Ax = y \quad (4)$$

Combining (2) and (4), we obtain:

$$\min_x \|x\|_1 + \|Ax - y\|_2^2. \quad (5)$$

This change in the objective function (also known as LASSO [22, 23]) makes it possible to solve the linear

system, even in presence of noise or truncated values in the matrix A , and vector y .

The l_1 -minimization problem of (4) can be converted to a linear programming problem [10]. This leads to a set of algorithms in CS which are referred to as “Basis Pursuit” [21]. There also exist other sets of algorithms that use a greedy iterative approach, known as “Matching Pursuit” [24]. Moreover, there are several strong guarantees for the reconstruction through l_1 -minimization [25].

As we make use of the CS technique in our approach, we would like to have a quick survey of some of the works that utilized this technique in the context of network analysis. CS has been mainly studied in the context of signal and image processing [26–28] and its use in the area of network analysis is still in its first stages of development.

In the context of network analysis, CS has been mostly used in the field of wireless sensor networks [29–31]. The main idea concerns the recovery of information (e.g. the temperature of an area) based on the samples aggregated from the network sensors.

In [32], the question of whether it is possible to quickly infer and monitor the network link characteristics from indirect end-to-end (aggregate) measurements is analyzed. This question lies in the area of network tomography, and in this work different aspects of it are analyzed by using the CS theory. Also, in network traffic monitoring, we can mention the works in [33, 34]. In [33], CS theory is used in order to reduce the memory cost in routers and switches. In [34], CS is exploited in order to recover the missing values of a network traffic matrix. In [35], CS is used in the context of P2P networks. By exploiting CS theory, the authors devise an approach based on random walks to spread CS random combinations to participants in a random peer-to-peer (P2P) overlay network.

The other two works which are more related to ours are [36] and [37]. In both of these works, the authors aim to use CS as a tool to predict the topology of the network, although their settings and assumption are completely different from each other. In [36], the authors introduce a new approach based on the penalized linear regression to estimate sparse partial correlation between different regions of interests of the brain network, and by employing the CS theory they are able to have a successful recovery of the brain network. However, the proposed approach in that work can not be generalized in other areas and can only

be used in the context of brain networks. In particular, we can not always define a partial correlation between different regions of the network. For many types of networks, such as online social networks, we don’t have access to several sample subjects of the whole network. Hence, we can not define any correlation measure and we are unable to use their approach in networks other than networks with similar properties as brain networks. In [37], the authors articulate a general method for addressing the problem of how to uncover the network topology using evolutionary-game data based on compressive sensing. The key to solving the network reconstruction using evolutionary-game data problem lies in the relationship between the agents payoffs and strategies. The interactions among agents in the network can be characterized by an $N \times N$ adjacency matrix and the sparsity of a single node’s neighborhood (adjacent nodes) makes the compressive sensing framework applicable.

III PROBLEM FORMULATION

1 PROBLEM STATEMENT

Consider the static directed network $G(V, E)$ with $|V| = n$ nodes and set of the edges E . We assume that we are totally aware of which nodes exist in the network and have no information about the edges. To reconstruct the network, we must predict the value of the elements of the adjacency matrix. Hence, we are looking for G^* where $\|G^* - G\| \leq \epsilon$. Ideally we would like to have $\epsilon = 0$, but we examine a more general setting where it is possible to have noisy or truncated data. Obviously this few information about the network is not enough for a tractable reconstruction. Thus, we need some external data about the network in order to be able to continue any further. We assume that there is an external process which has the two below features:

1. It can be run several times on the network
2. Its output can be considered as the linear combination of the edges of the network and a particular measure defined on each edge.

From now on, we consider the information diffusion as the external process in which any type of information or disease (e.g. news headlines, virus, rumor, etc.) diffuses over an underlying network. As an example, Figure 2(a) shows a news blogs network, where the nodes represent the blogs. There can be

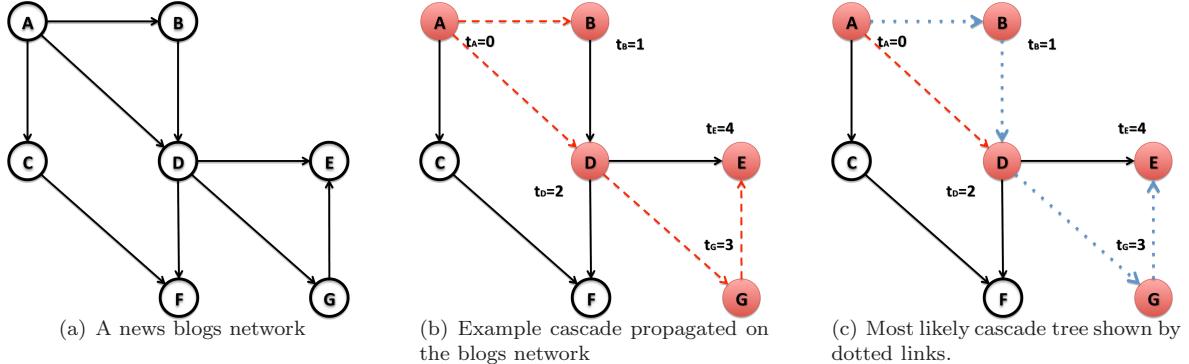


Figure 2: An example of information diffusion on a news blogs network. (a) A news blogs underlying network. (b) Example of a hidden real cascade tree (i.e. news coverage sequence) propagated on the blogs network. Each node hit by the cascade is shown in a different color and is labeled by a time stamp which shows the time of its news coverage. The cascade caused by the propagation of a particular news can be represented as: $News\ Cascade = <(\phi, A, t_A), (A, B, t_B), (A, D, t_D), (D, G, t_G), (G, E, t_E) >$. (c) The most likely cascade tree (shown by dotted links) as an approximation for the hidden real cascade tree. It can be seen that the most likely cascade tree is not necessarily identical to the real cascade tree in (b).

many interpretations for the edges of such network. In this example, we can simply consider for any directed edge (u, v) that node u is one of the sources of the news for node v . It is usually assumed that the diffusion as an external process occurs on a network, meaning the information spreads over the edges of an underlying network. A simple example for the diffusion process is shown in Figure 2(b). The diffusion process on such network can be considered as the coverage of a particular news in the different blogs with different time stamps. However, the network over which propagations take place is usually unknown and unobserved [16]. The goal now is to reconstruct the unknown network over which news originally propagated so that we know each node's information sources.

Consider the network of news blogs in Figure 2(b). Obviously, news propagation acts as a diffusion process and happens frequently in such network, which satisfies the first property that we would like the external process to have. To show that the diffusion process satisfies the second property of an external process, we mention some preliminaries related to the information networks and then in the section IV, we show the second property for this process.

2 INFORMATION NETWORKS AND THE CASCADING BEHAVIOUR

Each diffusion of the information throughout the network, creates an information cascade. In other words, an information cascade can be considered as a sequence of nodes, that are being visited by the diffusion process.

Definition 1. [16] A cascade is a set of triples $(u, v, t_v)_c$, which means that cascade c reached node v at time t_v by spreading from node u (by propagating over the edge (u, v)). We denote the fact that the cascade initially starts from some active node v at time t_v as $(\phi, v, t_v)_c$. In Figure 2(b), the cascade caused by the propagation of a particular news can be represented as:

$$News\ Cascade = <(\phi, A, t_A), (A, B, t_B), (A, D, t_D), (D, G, t_G), (G, E, t_E) >.$$

We assume the only data we can obtain from the cascades is the time that the cascade has reached a node. In the news blogs example in Figure 2(b), the cascade data can be considered as the coverage of a particular news in the different blogs ordered by the time stamp of the coverage. In information networks like our news blog example, we are usually not aware of the nodes sources of information. Although there are latent sources for any node, we ignore them and assume each node's information source is its neighbors

in the underlying network. This fact happens commonly in blogs networks as we do not know how a blog got information about a particular news. Thus, we only get to observe the pairs $(v, t_v)_c$.

Definition 2. Hit time can be described as the time t_v when node v got included by the cascade c . In news blogs example, the cascade hit times of each blog by the news cascade can be shown as:

$$\text{Cascade Hit Times} = < (A, t_A), (B, t_B), (D, t_D), (F, t_F), (G, t_G), (E, t_E) >$$

Now, given such data about node hit times for a number of different cascades, we aim to recover the unobserved directed network G , the network over which the cascades originally spread.

We consider the independent cascade model [38] which states that a node spreads the information to each of its neighbors independently with some chosen probability. This model implicitly assumes that every node v in cascade c is hit by at most one node u . That is, it only matters when the first neighbor of v spreads the cascade. In other words, only one neighbor of v actually activates v . Thus, the structure of a cascade c created by the diffusion process is fully described by a directed tree T , contained in the directed graph G .

3 MAXIMUM PROBABILITY CASCADE TREE

Now, we aim to give an approximation for the directed tree T related to a cascade c . We use the probabilistic model of how cascades spread over the edges of the network.

Definition 3. [16] Define the probability $P_c(u, v)$ as the conditional probability of observing cascade c spreading from u to v . Since the cascade can only propagate forward in time, if node u got reached after node v ($t_u > t_v$), then $P_c(u, v) = 0$. As an intuition, the probability of propagation $P_c(u, v)$ between a pair of nodes u and v in the cascade c decreases with the more difference in their hit times. In news blogs networks, an old news most probably does not provide any information for any blog. For simplicity, we consider $P_c(u, v)$ to follow the well-known exponential cascade transmission distribution [39]:

$$P_c(u, v) = P_c(\Delta_{u,v}) = e^{-\frac{\Delta_{u,v}}{\alpha}} \quad (6)$$

Where $\Delta_{u,v} = t_v - t_u$ and α is the adjustment parameter.

A cascade stops with the probability $(1 - \beta)$ and continues over an edge with the probability β . The likelihood of a cascade spreading in a given tree pattern T is calculated as [16]:

$$P(c|T) = \prod_{u,v \in E_T} \beta P_c(u, v) \prod_{u \in V_T, (u,x) \in E \setminus E_T} (1 - \beta) \quad (7)$$

Where $T = (V_T, E_T)$ is a tree.

Also the probability that a cascade c can occur in the graph G is defined as [16]:

$$P(c|G) = \sum_{T \in \tau_c(G)} P(c|T) P(T|G) \quad (8)$$

Where $\tau_c(G)$ is the set of all the directed connected spanning trees on a subgraph of G induced by the nodes that got hit by cascade c . In case T is inconsistent with the observed data, then $P(c|T) = 0$.

As there exist exponential number of possible trees, calculating the probability that a particular cascade can happen on an underlying network seems intractable. However, we use an approximation for the likelihood of a single cascade by considering only the most likely tree instead of all possible propagation trees:

$$P(c|G) = \max_{T \in \tau_c(G)} P(c|T) P(T|G) \quad (9)$$

We consider all possible trees to be equiprobable to occur. In other words $P(T|G) = \frac{1}{|\tau_c(G)|}$. The maximum probability tree can be found by choosing, for each node v , an incoming edge (u, v) with maximum probability [40]. Figure 2(c) shows the most probable cascade tree for the news cascade in Figure 2(b). As it can be seen, it is not necessary that the most likely cascade tree be the true cascade tree. However, intuitively it can be considered as a reasonable approximation. We will show that this approximation will be suitable for the proposed framework.

IV PROPOSED FRAMEWORK: CS-NETREC

Since we can run several cascades on the desired network, we run several cascades and use the cascades' probability data to reconstruct the network of interest. In particular, we form a linear system with the

$$\begin{bmatrix} LP(c_1|G) \\ LP(c_2|G) \\ LP(c_3|G) \\ \vdots \\ LP(c_m|G) \end{bmatrix} = \begin{bmatrix} LP_{c_1}(v_1, v_2) & \dots & LP_{c_1}(v_i, v_j) & \dots & LP_{c_1}(v_n, v_{n-1}) \\ LP_{c_2}(v_1, v_2) & \dots & LP_{c_2}(v_i, v_j) & \dots & LP_{c_2}(v_n, v_{n-1}) \\ LP_{c_3}(v_1, v_2) & \dots & LP_{c_3}(v_i, v_j) & \dots & LP_{c_3}(v_n, v_{n-1}) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ LP_{c_m}(v_1, v_2) & \dots & LP_{c_m}(v_i, v_j) & \dots & LP_{c_m}(v_n, v_{n-1}) \end{bmatrix} \begin{bmatrix} v_{1,2} \\ \vdots \\ v_{i,j} \\ \vdots \\ v_{n,n-1} \end{bmatrix}$$

Figure 3: The linear system mapped from the diffusion of m cascades on the network where $LP(c_k|G) = \log P(c_k|G)$ and $LP_{c_k}(i, j) = \log P_{c_k}(i, j)$. The vectorized network adjacency is the unknown vector which we know that is a sparse one. Note that the ordering of $LP_{c_k}(i, j)$ s and $v_{i,j}$ s must be the same.

edges as the unknown vector and try to find a solution for this system (i.e. a reconstruction for the network). More details can be seen in Algorithm IV.1.

Assuming T_c^* is the most likely tree corresponding to the cascade c , we hope that all the edges in T_c^* represent the real edges in the underlying network G , although this is not necessary.

Thus, after taking log from both sides of the Eq. (9), we can approximate it as the inner product of two vectors:

$$LP(c|G) = vLP(c|T_c^*)^T \cdot vec(Adj(G)) \quad (10)$$

Where $LP(c|G) = \log P(c|G)$ and $vec(Adj(G))$ is the vectorized binary adjacency matrix of network G in some particular order, and $vLP(c|T_c^*)$ is the vector of individual edge transmission log probabilities with nonzero elements corresponding to the edges in T_c^* . In particular, the k -th element of $vLP(c|T_c^*)$ can be shown as:

$$vLP(c|T_c^*)_k = LP_c(i, j) = \log P_c(i, j) \quad (11)$$

It is obvious that for the consistency of the formulation, the element order of both vectors (which correspond to all possible edges in the network G) must be the same. By this approximation, we are now able to show a single run of the diffusion process (i.e. a cascade) as the linear combination of the edges of the network and the probability measure defined on each edge. Thus, the diffusion process satisfies the second property of an external process that was mentioned earlier.

Now that the diffusion process satisfies both properties that an external process should have, we use this process and the CS framework, in order to reconstruct the network of interest.

We can model m runs of the diffusion process as the linear system: $y = Ax$, where each equation in this linear system is equal to (10) and the unknowns are the edges of the desired network. More details about the formation of this linear system is shown in Figure 3.

Our algorithm consists of two main steps. First, the formation of the linear system and second, finding the solution for it. There are four sub-steps in the linear system formation:

For each cascade c (Which forms each equation in the linear system):

1. We find the most probable tree and set it to T_c^* .
2. For each possible edge in G , we calculate the edge transmission log probabilities. In particular, for each edge in T_c^* we use logarithm of Eq. (6) and for the rest of the edges in G we consider zero probability.
3. We calculate $LP(c|G) = \log P(c|G)$ from Eq. (9) and add it as a row to the vector y .
4. We form the vector $vLP(c|T_c^*)$ from the previous step and add it as a row to the matrix A .

After the formation of the above linear system, we are now ready to apply CS to find a sparse solution for the system. For this purpose we use Eq. (5).

V EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of our approach for network reconstruction. First we introduce the synthetic and real datasets that we used for

the evaluation. Next, we introduce the metrics of evaluation, and finally, we present the required analysis on the achieved results.

1 DATASETS

We consider both synthetic and real networks.

1. Synthetic networks:

We use three well-known classic models for generating directed networks, namely, the Erdos-Renyi model [41] with 500 edges, the Small world graph model [42] with each node being connected to 4 nearest neighbors in ring topology and the rewiring probability of 0.4, and the Barabasi-Albert model [43] with each new node getting connected to 5 existing nodes. Each of these networks have 100 nodes. Also, we use a variant of Kronecker graph model [44], namely, Core-Periphery Kronecker [45] with 256 nodes which results in around 600 edges in several generations.

2. Real networks:

We consider three directed real-world networks. First, we consider the network of American football games between Division IA colleges during regular season fall 2000 [46] which includes 115 nodes and 615 edges. Second, we consider the neural network of the Caenorhabditis elegans worm (C.elegans) [42] with 306 nodes and 2345 edges and Third, we use the network of 500 busiest commercial airports in the United States [47] with 500 nodes and 2980 edges.

2 SETTINGS

In each of the test cases for the synthetic networks, we generated 100 networks with corresponding sets of cascades. For the real-world datasets, we only generated 100 set of cascades. For the cascade generation, we use the same process as NetInf, one of the state-of-the-art methods for diffusion network inference.

To evaluate the accuracy of our approach, we can measure the precision and recall of our method. Precision refers to the number of correctly inferred diffusion links divided by the total number of inferred diffusion links, and recall refers to the number of correctly diffusion links divided by the total number of links in the network. To avoid the trade-off between

precision and recall and to consider both, we used the F-measure metric. This metric presents the harmonic mean of both precision and recall:

$$F = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

We set a threshold of 0.5 for the predicted values of the vector x . We tested other values of threshold and observed no significant effect in the accuracy of our approach. Thus, we consider all elements above 0.5 as a predicted edge.

3 PARAMETER ANALYSIS

Here, we analyze the sensitivity of our approach to the parameters of the cascades, and if possible, fix them to obtain the rest of results based on the fixed values of those parameters. The information cascades have two parameters to diffuse throughout the network.

First, we show the sensitivity of the results to α which represents the cascades' speed. To this end, we consider the cascades with a constant β equal to 0.5. As shown in the Figure 4, it seems for all networks, the F-measures reach a consistency when the value of α is around 3. Hence, we use this fix value for α . For the small values of α , i.e. near zero values, the probability of each edge transmission (Eq. (6)) will get closer to zero. This causes more truncated values in the matrix A and also in calculation of $LP(c|G)$. Thus, we expect a lower accuracy for small values of α as shown in Figure 4. For very large values of α , the probability of each edge transmission becomes closer to 1. Thus, its logarithm equals to zero which results in trivial equations in the linear system ($0 = 0$), and we are not able to obtain any information from the cascade data.

The length of the cascades, controlled by the parameter β , can significantly alter the results of any network reconstruction approach. We consider the cascades with a constant α equal to 1. Obviously, when one runs a number of cascades on a network with a small value of β , the cascades will be shorter. Thus, a smaller number of edges will be observed in such situation and as a consequence, the resulted recall will be very low. The same phenomenon can be seen in our framework in Figure 5, although the F-measure in BA and Small world are being degraded less from the small recall than the others. On the other hand, increasing the value of β will make the cascades longer. This causes the matrix A to have less zero variables

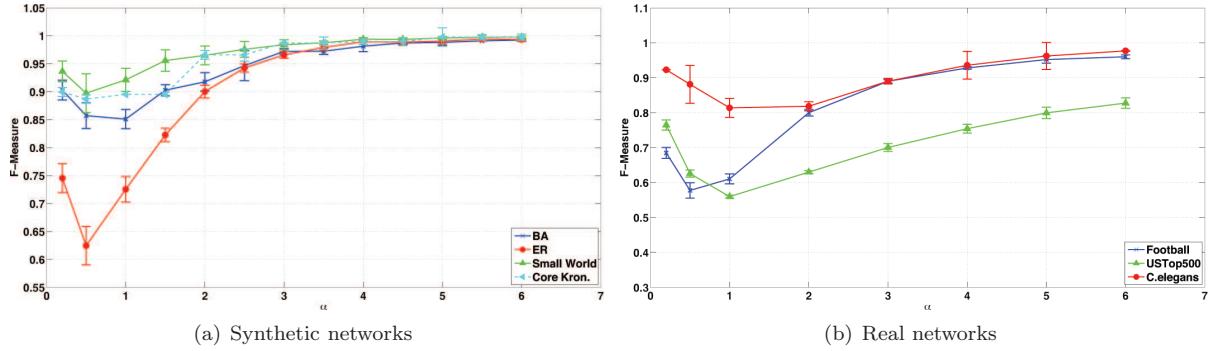


Figure 4: α parameter analysis using CS-NetRec - It can be seen that the value $\alpha = 3$ is where the F-measures become consistent.

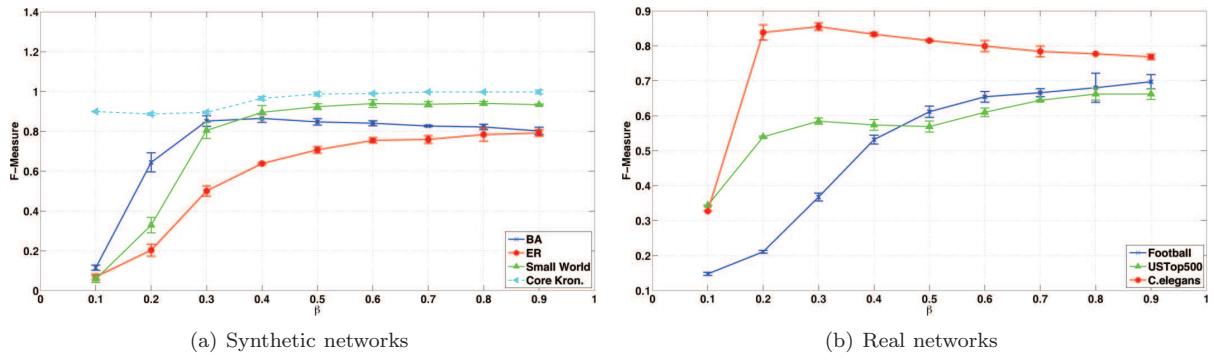


Figure 5: β parameter analysis using CS-NetRec - The value of $\beta = 0.5$ is where the F-measures become consistent.

which results in a longer and in some cases like BA model, less accurate optimization procedure. As seen in Figure 5, in most of the networks, all the measures become consistent when reached to the value of 0.5 for β . Thus we fix β at this value.

4 CASCADE DEPENDENCY

Obviously, it is almost never possible to have equal number of processes to the number of possible edges in a network, because it is computationally expensive. Also, increasing the number of transmitting cascades over the network leads to more accurate prediction of the network topology. We tested this scenario by running 250 to 2500 cascades on each of the synthetic networks and the Football network, where there are around 10000 possible edges in the graph. On other real datasets, we ran 250 to 10000 cascades. In the C.elegans dataset, there are about 93000 possible edges, and in US Top 500 dataset, this number reaches to around 250000. In all of our measurements we consider the resulting F-measures.

In Figure 6, particularly for Small world and BA datasets which are closer to real-world graphs than ER model, it can be observed that even on low number of cascades (e.g. half of the number of existing edges in the graphs), we have high values of F-measure, almost all above 0.5, considering around 10000 possible edges to be predicted. Thus the results demonstrate that our framework can work accurately even on very low information from the network of interest. In the Kronecker dataset and Small world, we have above 0.8 F-measure when the number of cascades are roughly the same as the number of edges in the graph.

In real datasets, we have the same performance in low number of cascades as in the synthetic datasets. In C.elegans and US Top 500, with 2000 cascades we obtain around 0.75 and 0.5 F-measures while there are around 93000 and 250000 possible edges to predict in each of these datasets, respectively. Furthermore, we observe that the F-measure in Football dataset, due to the lower number of nodes, reaches the 0.9 F-measure by 4000 cascades. Thus, we observe that in

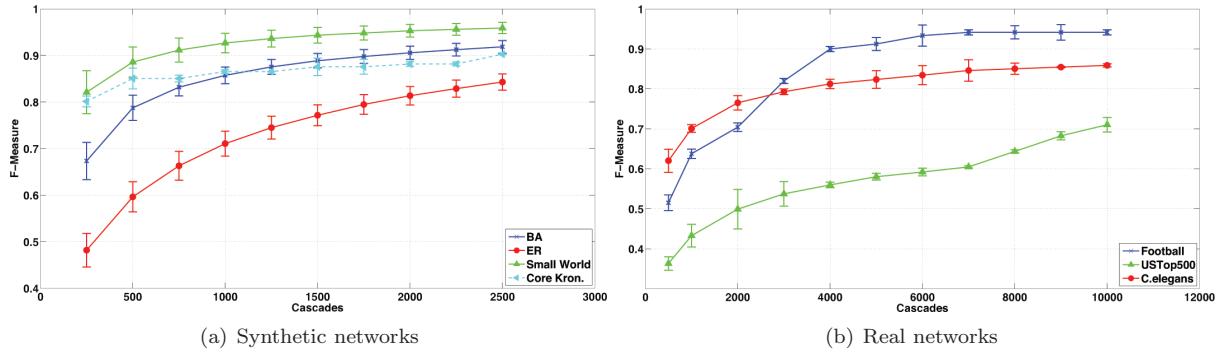


Figure 6: Different networks cascade dependency using CS-NetRec

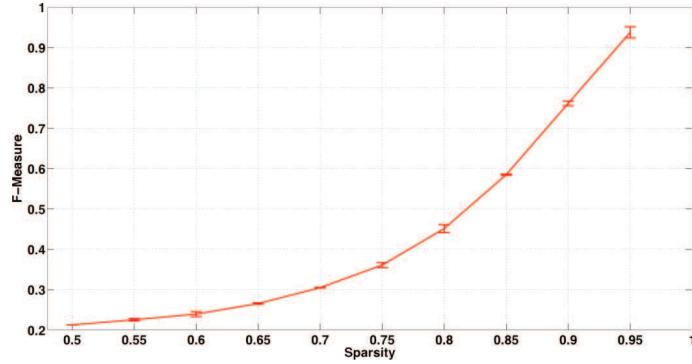


Figure 7: The effect of the sparsity of the network adjacency in ER model using CS-NetRec. The less sparsity in the unknown vector, the less accuracy in the reconstruction of the network.

spite of the same scale of this dataset to the synthetic ones, we need more number of measurements to reach a high F-measure value.

The reason behind better results for F-measure in higher number of cascades can be justified through several aspects. First, with more cascades we traverse more edges of the graph. Meanwhile, the transmission probability over an edge gets repeated in more equations of the linear system. Thus, the effect of that edge on the increase of cascade probabilities can be seen more and the predicted value for the corresponding element in the vectorized adjacency matrix will get a higher value (i.e. passes the considered prediction threshold). Furthermore, in CS theory, higher number of measurements (i.e. rows of the matrix A) most probably results in better recovery which can be observed in this result.

5 THE EFFECT OF SPARSITY OF THE NETWORK ADJACENCY

According to the CS theory, less sparsity in the unknown vector, results in less accuracy in its recovery. To show this phenomenon in our framework, we consider several ER networks with 100 nodes and number of edges ranging from 500 to 5000. In other words, the sparsity ranges from 0.5 to 0.95. Again we consider all the F-measures to evaluate the results.

As expected, in Figure 7, we observe that the more sparsity in the same scale (i.e. number of nodes) yields to better performance in our approach. For 0.95 sparsity, we have a near perfect reconstruction. However, decreasing the sparsity to 0.9, diminishes the F-measure to below 0.8. Thus, in the same dataset, the network reconstruction with more sparsity can result in higher accuracy.

The sparsity can not be compared in different datasets. To see this, we may refer to the results for C.elegans and US Top 500 in Figure 6(a). Although both datasets have above 0.97 sparsity in their vec-

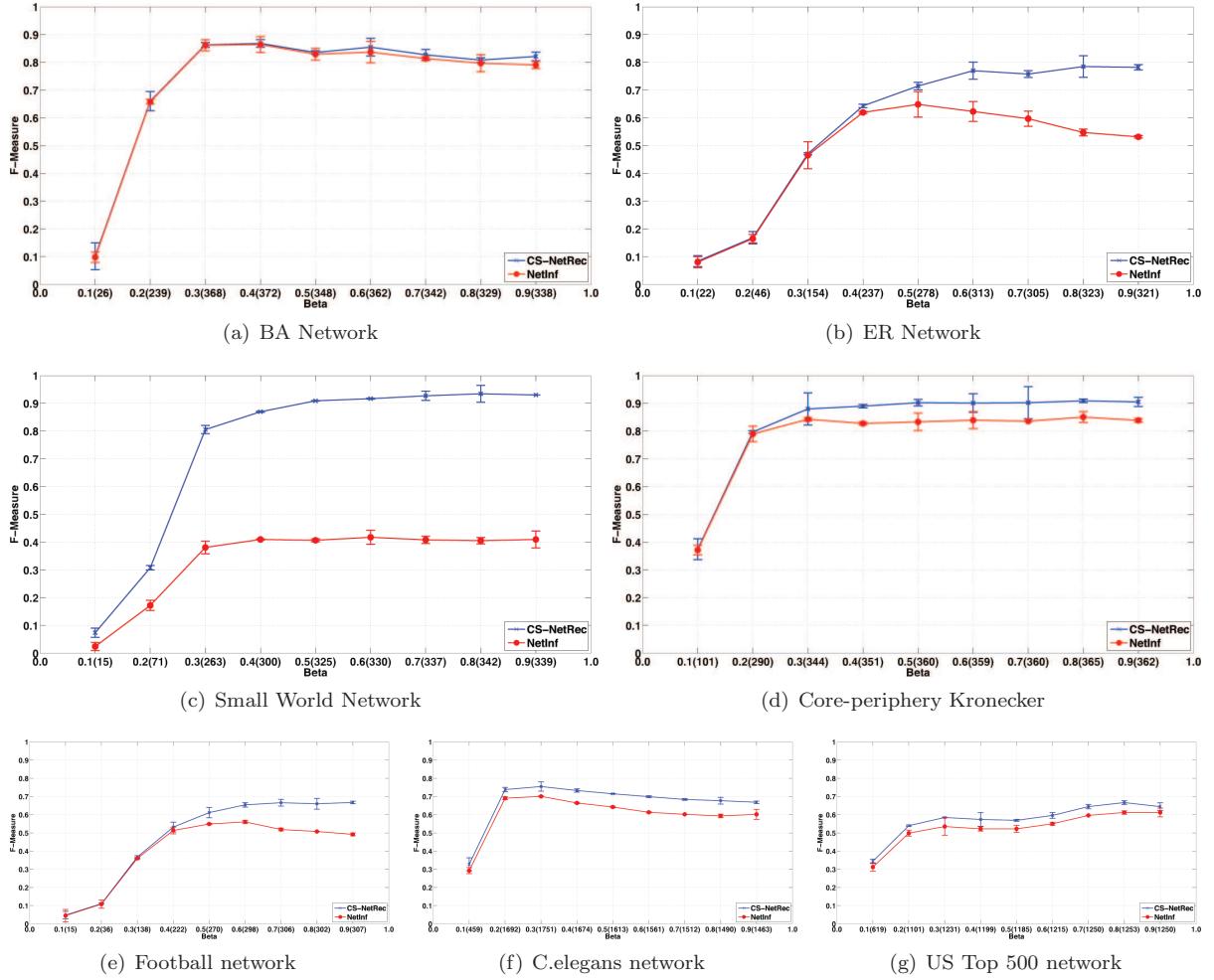


Figure 8: Performance comparison of CS-NetRec with NetInf, considering different values of β parameter (i.e. cascade lengths) and $\alpha = 1$. (a,b,c,d) Synthetic networks (e,f,g) Real networks. We considered 2500 cascades for BA, ER, SmallWorld and Football datasets, 8000 cascades for Kronecker graph and 10000 cascades for C.elegans and US Top 500. The numbers in the parantheses show the number of edges recovered.

torized adjacency matrix, but due to the larger scale of US Top 500 and the more cascades needed for the better reconstruction, the F-measure related to C.elegans is relatively higher than US Top 500 with the same number of cascades.

6 PERFORMANCE COMPARISON WITH NETINF

As we present our proposed framework in the context of information networks, we would like to compare the performance of our method with one of the state-of-the-art methods for inferring networks of diffusion, namely NetInf [16].

For the comparison, we used the code provided by the authors. We consider different values for the parameter β as we would like to evaluate the performance of methods in presence of more local information (i.e. shorter cascades) and more global information (i.e. longer cascades). We also set α to 1 in our tests. Given a fixed number of cascades, we ran NetInf with the number of iterations equal to the number of edges that our method can reconstruct. The results can be seen in Figure 8. It can be observed that in all test cases, the resulted F-measure in our method is higher than or equal to NetInf, and as we increase the value of β , the difference in F-measure becomes more. In Small world, ER, and Football datasets, the difference is more clear. For $\beta = 0.9$ the improvement

by our method in F-measure is about 0.5, 0.25 and 0.18 for Small world, ER and Football datasets, respectively. Overall, for different cascade lengths, our method outperforms NetInf by an average of 9-10% improvement in terms of the resulted F-measure.

VI CONCLUSION

In this paper, we studied the problem of network reconstruction and introduced a general framework based on which any type of observations on an external process on the network can be utilized to reconstruct the network of interest. As a special case, we considered the diffusion of the information cascades as an external process on the underlying network. By utilizing the probabilistic measures of information cascades, we formulated the problem of network reconstruction as a linear system. Since most of the time, this linear system is under-determined, we used the theory of compressed sensing as a tool to reconstruct the network of interest. By numerical experiments, we demonstrated that this framework will converge to an accurate solution. Also, the results suggest that in the context of information networks, our method can perform even better than the state-of-the-art method, NetInf.

Several directions can be pursued for the future work. Here we used the adjacency matrix as the unknown vector and thus it is interesting to look for ways to reduce the dimensionality of the linear system. Also it would be interesting to use other external processes and utilize other features for the network nodes to reconstruct the network of interest using the proposed framework.

References

- [1] J. L. Schafer and J. W. Graham, “Missing data: Our view of the state of the art.”, *Psychological Methods*. Vol 7(2), Jun 2002, 147-177., vol. 7, no. 2, pp. 147–177, 2002.
- [2] C. T. Butts, “Network inference, error, and informant (in)accuracy: a Bayesian approach”, *Social Networks*, vol. 25, no. 2, pp. 103–140, May 2003.
- [3] G. Kossinets, “Effects of missing data in social networks”, *Social Networks*, vol. 28, no. 3, pp. 247–268, July 2006, PT: J; PG: 22.
- [4] R. Guimerà and M. Sales-Pardo, “Missing and spurious interactions and the reconstruction of complex networks”, *Proceedings of the National Academy of Sciences*, vol. 106, no. 52, pp. 22073–22078, Dec. 2009.
- [5] A. Clauset, C. Moore, and M. E. J. Newman, “Hierarchical structure and the prediction of missing links in networks”, *Nature*, vol. 453, no. 7191, pp. 98–101, May 2008.
- [6] K. Bleakley, G. Biau, and J.-P. Vert, “Supervised reconstruction of biological networks with local models”, in *ISMB/ECCB (Supplement of Bioinformatics)*, 2007, pp. 57–65.
- [7] Y. Yamanishi, J.-P. Vert, and M. Kanehisa, “Protein network inference from multiple genomic data: a supervised approach”, *Bioinformatics*, vol. 20, no. 1, pp. 363–370, Jan. 2004.
- [8] M. Kim and J. Leskovec, “The Network Completion Problem: Inferring Missing Nodes and Edges in Networks.”, in *SDM. 2011*, pp. 47–58, SIAM / Omnipress.
- [9] S. Hanneke and E. P. Xing, “Network completion and survey sampling”, *Journal of Machine Learning Research - Proceedings Track*, vol. 5, pp. 209–215, 2009.
- [10] E. J. Candes and T. Tao, “Decoding by Linear Programming”, *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.
- [11] E. J. Candes and M. B. Wakin, “An Introduction To Compressive Sampling”, *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 21–30, Mar. 2008.
- [12] D. L. Donoho, “High-dimensional centrally symmetric polytopes with neighborliness proportional to dimension.”, *Discrete and Computational Geometry*, vol. 35, no. 4, pp. 617–652, 2006.
- [13] E. J. Candes and B. Recht, “Exact matrix completion via convex optimization”, *Found. Comput. Math.*, vol. 9, no. 6, pp. 717–772, Dec. 2009.
- [14] D. Liben-Nowell and J. Kleinberg, “The link-prediction problem for social networks”, *J. Am. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, May 2007.
- [15] D. S. Goldberg and F. P. Roth, “Assessing experimentally derived interactions in a small world”, *Proceedings of the National Academy of Sciences*, vol. 100, no. 8, pp. 4372–4376, Apr. 2003.

- [16] M. Gomez Rodriguez, J. Leskovec, and A. Krause, “Inferring networks of diffusion and influence”, in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, 2010, KDD ’10, pp. 1019–1028, ACM.
- [17] M. Eslami, H. R. Rabiee, and M. Salehi, “Dne: A method for extracting cascaded diffusion networks from social networks.”, in *Social-Com/PASSAT*. 2011, pp. 41–48, IEEE.
- [18] D. Gruhl, R. V. Guha, D. Liben-Nowell, and A. Tomkins, “Information diffusion through blogspace”, in *Proceedings of the 13th international conference on World Wide Web*, New York, NY, USA, 2004, WWW ’04, pp. 491–501.
- [19] G. Kossinets, J. Kleinberg, and D. Watts, “The structure of information pathways in a social communication network”, in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, 2008, KDD ’08, pp. 435–443.
- [20] E. J. Candes, J. K. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements”, *Comm. Pure Appl. Math.*, vol. 59, no. 8, pp. 1207–1223, Aug. 2006.
- [21] D. L. Donoho, “Compressed sensing”, *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [22] R. Tibshirani, “Regression shrinkage and selection via the lasso”, *Journal of the Royal Statistical Society (Series B)*, vol. 58, pp. 267–288, 1996.
- [23] E. J. Candes, M. Rudelson, T. Tao, and R. Vershynin, “Error Correction via Linear Programming”, *Foundations of Computer Science, Annual IEEE Symposium on*, vol. 0, pp. 295–308, 2005.
- [24] J. A. Tropp and A. C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit”, *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [25] E. J. Candes, “The restricted isometry property and its implications for compressed sensing”, *Comptes Rendus Mathematique*, vol. 346, no. 9-10, pp. 589–592, May 2008.
- [26] A. C. Sankaranarayanan, P. K. Turaga, R. Chellappa, and R. G. Baraniuk, “Compressive acquisition of dynamic scenes”, *CoRR*, vol. abs/1201.4895, 2012.
- [27] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, “Single-Pixel Imaging via Compressive Sampling”, *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, Mar. 2008.
- [28] M. B. Wakin, J. N. Laska, M. F. Duarte, D. Baron, S. Sarvotham, D. Takhar, K. F. Kelly, and R. G. Baraniuk, “An architecture for compressive imaging”, in *in IEEE International Conference on Image Processing (ICIP*, 2006, pp. 1273–1276.
- [29] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, “Compressive wireless sensing”, in *Proceedings of the 5th international conference on Information processing in sensor networks*, New York, NY, USA, 2006, IPSN ’06, pp. 134–142, ACM.
- [30] J. J. Meng, H. Li, and Z. Han, “Sparse event detection in wireless sensor networks using compressive sensing.”, in *CISS. 2009*, pp. 181–185, IEEE.
- [31] J. Haupt, W. U. Bajwa, M. Rabbat, and R. Nowak, “Compressed Sensing for Networked Data”, *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 92–101, Mar. 2008.
- [32] W. Xu, E. Mallada, and A. Tang, “Compressive sensing over graphs”, *CoRR*, vol. abs/1008.0919, 2010.
- [33] Y. Lu, A. Montanari, B. Prabhakar, S. Dharmapurikar, and A. Kabbani, “Counter braids: a novel counter architecture for per-flow measurement”, in *Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, New York, NY, USA, 2008, SIGMETRICS ’08, pp. 121–132, ACM.
- [34] Y. Zhang, M. Roughan, W. Willinger, and L. Qiu, “Spatio-temporal compressive sensing and internet traffic matrices”, in *SIGCOMM*, 2009, pp. 267–278.
- [35] R. Gaeta, M. Grangetto, and M. Sereno, “Local access to sparse and large global information in p2p networks: A case for compressive sensing.”, in *Peer-to-Peer Computing*. 2010, pp. 1–10, IEEE.

- [36] H. Lee, D. S. Lee, H. Kang, B.-N. Kim, and M. K. Chung, “Sparse brain network recovery under compressed sensing”, *IEEE Trans. Med. Imaging*, vol. 30, no. 5, pp. 1154–1165, 2011.
- [37] W.-X. Wang, Y.-C. Lai, C. Grebogi, and J. Ye, “Network Reconstruction Based on Evolutionary-Game Data via Compressive Sensing”, *Physical Review X*, vol. 1, no. 2, pp. 021021, Oct. 2011.
- [38] D. Kempe, J. Kleinberg, and E. Tardos, “Maximizing the spread of influence through a social network”, in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, 2003, KDD ’03, pp. 137–146, ACM.
- [39] J. Leskovec, M. McGlohon, C. Faloutsos, N. S. Glance, and M. Hurst, “Patterns of Cascading Behavior in Large Blog Graphs”, in *Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA*, 2007.
- [40] J. Edmonds, “Optimum branchings”, *Res. Nat. Bur. Standards*, vol. 71B, pp. 233–240, 1967.
- [41] P. Erdos and A. Renyi, “On the evolution of random graphs”, *Publ. Math. Inst. Hung. Acad. Sci*, vol. 5, pp. 17–61, 1960.
- [42] D. J. Watts and S. H. Strogatz, “Collective dynamics of small-world networks”, *Nature*, vol. 393, no. 6684, pp. 440–442, June 1998.
- [43] A.-L. Barabási and R. Albert, “Emergence of Scaling in Random Networks”, *Science*, vol. 286, no. 5439, pp. 509–512, Oct. 1999.
- [44] J. Leskovec and C. Faloutsos, “Scalable modeling of real graphs using kronecker multiplication”, in *Proceedings of the 24th international conference on Machine learning*, New York, NY, USA, 2007, ICML ’07, pp. 497–504, ACM.
- [45] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, “Statistical properties of community structure in large social and information networks”, in *Proceedings of the 17th international conference on World Wide Web*, New York, NY, USA, 2008, WWW ’08, pp. 695–704, ACM.
- [46] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks”, *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, June 2002.
- [47] V. Colizza, R. Pastor-Satorras, and A. Vespignani, “Reaction–diffusion processes and metapopulation models in heterogeneous networks”, *Nat Phys*, vol. 3, pp. 276–282, Jan. 2007.

Scalable Link Prediction using Matrix Factorization

ارسال شده به کنفرانس (ICDE'14)

Abstract: Link prediction is a fundamental problem that appears in many applications such as recommender systems and biological networks. In this paper, we exploit the sparsity (*i.e.*, small number of links) of real-world networks in order to introduce a new link prediction method which is improved in scalability and accuracy than the existing state-of-the-art algorithms. We take a probabilistic approach by assigning a Poisson generative model to the underlying network, and present link prediction problem as a variant of matrix factorization problem. To the best of our knowledge, no previous method has used Poisson process for link prediction to improve scalability and accuracy. It is shown that the proposed method leads to fast inference of the parameters, and hence easily scales to large real-world networks. Moreover, it is possible to use the proposed method in several applications other than link prediction such as collaborative filtering, latent semantic indexing and dimensionality reduction. In addition to robustness and fast convergence, experiments show that the proposed method outperforms the popular unsupervised and supervised methods while being significantly more scalable.

Scalable Link Prediction using Matrix Factorization

Payam Siyari ^{#1}, Hamid R. Rabiee ^{#2}, Hamidreza Mahyar ^{#3}

[#] Sharif University of Technology

¹ siyari@ce.sharif.edu

² rabiee@sharif.edu

³ hmahyar@ce.sharif.edu

Abstract—Link prediction is a fundamental problem that appears in many applications such as recommender systems and biological networks. In this paper, we exploit the sparsity (*i.e.*, small number of links) of real-world networks in order to introduce a new link prediction method which is improved in scalability and accuracy than the existing state-of-the-art algorithms. We take a probabilistic approach by assigning a Poisson generative model to the underlying network, and present link prediction problem as a variant of matrix factorization problem. To the best of our knowledge, no previous method has used Poisson process for link prediction to improve scalability and accuracy. It is shown that the proposed method leads to fast inference of the parameters, and hence easily scales to large real-world networks. Moreover, it is possible to use the proposed method in several applications other than link prediction such as collaborative filtering, latent semantic indexing and dimensionality reduction. In addition to robustness and fast convergence, experiments show that the proposed method outperforms the popular unsupervised and supervised methods while being significantly more scalable.

I. INTRODUCTION

Study of networks has drawn considerable attention during the last decade by many researchers and organizations. Importance of this study lies in the appearance of networks in a vast variety of applications, from social interactions to transportation systems. Consequently, depending on that particular application, different challenges may arise. Link prediction is a fundamental problem in the study of all kinds of networks and presented methods for it can benefit many areas. For instance, the study of recommender systems deals heavily with various settings of the link prediction problem, whether the main system can be mapped as a network (*e.g.* the user-item bipartite networks where the underlying system is not necessarily a network) or is structured as a network of communicating entities (*e.g.* Online Social Networks). The existence of incomplete data in many real-world systems is another motivation for proposing an efficient method for link prediction. In biological networks, especially protein-protein interactions, the presence of highly noisy and inaccurate data naturally entails the use of link prediction methods in order to have a better inference about the structure of the interactions. Not only in biological applications, but also in almost all cases where aggregation of the complete data is not possible, having a good estimate of the network structure can help us avoid misleading inferences.

There are two main views to link prediction [1]. In one view

known as structural link prediction, the aim is to predict the presence or absence of the links in unobserved parts of the network. In the other view known as temporal link prediction, the input of the problem is a series of snapshots from the whole network and addition or removal of the links in later timestamps are investigated. In this paper, we focus on the first view, however, ideas concerning either of the views can be used in the other.

One of the most important challenges in link prediction is the ability of the proposed method to be performed on large-scale data, which is the main characteristic of real-world applications. At the same time, networked data happens to be highly sparse, meaning the number of links is very small. However, in many areas such as machine learning and signal processing, it is shown that sparsity can be turned to our advantage by incorporating it into our solutions [2], [3]. In this paper, we are aiming to use this property in real-world networks in order to achieve a significant speedup.

The approach taken in this work, has its roots in collaborative filtering [4]. In a nutshell, collaborative filtering is the study of user-item interactions and developing methods to predict the preference of the users for particular sets of items, based on their past behavior. The dominant approaches in collaborative filtering algorithms are based on latent factor models. In such approaches, the goal is to infer a set of latent factors for both users and items from the patterns in user-item interactions and simply judge the preference of a user for an item by the similarity of the corresponding factors of the user and of the item. The most successful incorporation of latent factor models, both in terms of accuracy and scalability, is done in matrix factorization framework, a recently emerged dimensionality reduction paradigm [5]. In this framework, we seek to find an approximation for an input matrix which minimizes a particular loss function. In basic matrix factorization, this approximation is considered to be the product of two rank- k matrices, one representing the latent factors of users and the other representing the items' in collaborative filtering sense. Therefore, the preference of a user for an item is set to be estimated by the dot product of the inferred user's and item's latent factors.

In this paper, we adopt a probabilistic matrix factorization [6] approach to address the problem of link prediction. Specifically, first we present a Poisson generative model as the likelihood for the network graph and aim to infer the

parameters of it using Maximum Likelihood (ML). We apply coordinate descent for inference. It is shown that the use of Poisson generative model, instead of other well-known models like Bernoulli-Logistic, allows us to use the sparsity in the network structure in order to boost the inference of model parameters and consequently, predicting the missing links. To the best of our knowledge, previous methods for link prediction, whether utilizing matrix factorization or not, do not exploit Poisson process to improve accuracy and speed. It is shown that our algorithm can perform in almost linear time to the number of the nodes of the network. It is important to note that with straightforward changes, our algorithm can be used in other areas involving non-negative factorization from a sparse matrix.

We evaluated the proposed algorithm on different real-world networks, from social to biological and co-authorship networks. The results demonstrate that our method could always perform much better than a well-known set of popular unsupervised scores. Also comparing our method to the state-of-the-art methods in link prediction, we find that our method performs better and is significantly more scalable. We also evaluated the robustness of our method to the fraction of unknown data and observed that even in absence of half of the data, our method performs reasonably well.

In summary, the main contributions of this work are:

- Presenting a fast, robust and scalable algorithm for link prediction by exploiting Poisson generative modeling and the sparse structure of the networks.
- Formulating the proposed method in the framework of matrix factorization which makes it possible to utilize our method in other areas, such as collaborative filtering, latent semantic indexing, dimensionality reduction, etc.

The rest of the paper is organized as follows. Section 2 covers related work and similar problems, in link prediction and matrix factorization. We will also discuss the shortcomings of previous methods. Section 3 presents the proposed method and its complexity analysis. Section 4 is devoted to the settings and results of experimental evaluations. Finally, Section 5 concludes the paper and mentions future directions.

II. RELATED WORK

In this section, we mention a brief introduction to matrix factorization and then, review the related work in the field of link prediction with emphasis on those exploiting the matrix factorization framework. We also present a quick survey of related problems to link prediction in network analysis.

A. Matrix Factorization

Matrix factorization is one of the most developed machine learning approaches in collaborative filtering and in the past decade, there have been numerous studies in this field. [4] presents a great survey of these techniques. Put simply, the goal in collaborative filtering is to predict the unknown entries of a user-item matrix. The entries of this matrix represent the ratings of users to the items. Matrix factorization, operating as a realization for latent feature models, tries to map both

users and items to a joint latent feature space and estimates the entries of the original matrix by:

$$r_{ij} \approx U_i^T V_j \quad (1)$$

where r_{ij} is the original rating of user i to item j and U_i and V_j are the latent feature vectors for i -th user and j -th item. In another sense, we are looking for latent user and item matrices U^* and V^* where:

$$(U^*, V^*) = \arg \min_{(U, V)} l(R, L(U^T V)) \quad (2)$$

where R , $l(\cdot, \cdot)$ and $L(\cdot)$ are rating matrix, proper loss and link functions, respectively. There is also a probabilistic view of matrix factorization in collaborative filtering [6], in which a conditional distribution over the rating matrix is defined as follows:

$$p(R|U, V, \sigma^2) = \prod_{(i,j) \in O} N(R_{ij}|(U_i^T V_j, \sigma^2)) \quad (3)$$

where $N(x|\mu, \sigma^2)$ is the normal density function and O is the set of observed ratings. There may be also priors on the parameters. The goal is now to infer the parameters using statistical techniques like Maximum Likelihood, Maximum A Posteriori, Markov Chain Monte Carlo (MCMC), etc.

Different constraints can be imposed to the optimization problem in (2). One of the most common constraints is the non-negativity of the factor matrices U and V . The studies about this problem lie in the area of Non-negative Matrix Factorization (NMF). [5] provides a comprehensive survey of different formulations and algorithms for NMF problems. We also use such constraint in our formulation for simplicity of optimization, while other reasons like better interpretability of latent factors and sparse decomposition are mentioned in [5].

B. Link Prediction

In general, network graph reconstruction from incomplete data has been widely studied in several fields under many different titles, amongst which link prediction is the most common. As mentioned, this problem has been mainly studied under two scenarios [1]. Structural link prediction is the case where we have the adjacency matrix of an observed network and aim to predict its unknown links. The other is temporal link prediction where we face a set of full snapshots from the graph and we are looking for the future ones. Despite being different, the methods presented in each of these two categories can be adapted to be used in the other. A simple example is the case where only a single snapshot of the graph is available.

Link prediction can be viewed as a binary classification problem in which presence or absence of links are treated as labels to pairs of nodes. Hence, the methods for link prediction are either unsupervised or supervised. In unsupervised methods, a precomputed similarity score (e.g., number of common neighbors) between unknown pairs is used to predict the existence of links between those pairs. Therefore, these methods do not involve any learning [1]. One of the earliest

studies in link prediction is [7] where an array of unsupervised scores are suggested. Many of the unsupervised scores have the advantage of quick calculation for the large graphs. But the varying performance of these scores makes them unable to be used as a reliable method in different datasets.

All other methods for link prediction are supervised methods. See [1] for a comprehensive categorization of these methods. Our proposed method lies in latent feature models category where the goal is to map the nodes to a latent feature space of some dimensionality and to estimate the presence or absence of the links using the inferred features from training data. Methods presented in [1], [8], [9], [10] are based on latent feature models.

In [1], the focus is on optimizing AUC measure (Area under ROC curve) because of high class imbalance between links and non-links. The method is presented in matrix factorization framework and is evaluated under different scenarios, e.g. with and without side information. However, the method does not seem to be scalable to very large graphs as in this method, it is needed to scan over all pairs of links and non-links. This can lead to a space and time complexity of $O(N^3)$ (where N is the number of nodes) which makes the prediction infeasible for a large dataset. Although data reduction and stochastic gradient descent is employed in order to alleviate this problem, there can be inaccurate and very low-speed convergence in large graphs using these techniques.

[9], [8] propose similar probabilistic approaches for link prediction by considering a Bernoulli-Logistic generative model for the underlying graph. They use Minorization-Maximization (MM) algorithm [11] to infer the parameters. In Appendix 1., it is shown that Bernoulli trial modeling may not scale well to large datasets. Also, we will mention some of the issues regarding the optimization procedure of [9] in experimental evaluations section. In this paper, we show that changing the generative model to a Poisson process leads to scalable inference of parameter models. [12] tries to handle the case of Bernoulli trial and logistic regression over the binary adjacency matrix by matrix factorization. However, the training part uses MCMC techniques which is not efficiently applicable in large-scale problems.

One of the important parameters in latent feature models is the latent space dimension. In the meantime, setting an optimal value for the latent space dimension appears to be a time consuming task. Thus, [13], [14] try to present non-parametric approaches where the latent space dimension is inferred automatically from the data. They use fully-Bayesian treatment and Indian Buffet Process, however, the related results for these methods are limited to small datasets.

In summary, the above methods and many others that have been presented based on the latent feature models, cannot be scaled to very large datasets. We aim to present a scalable method considering the fact that network adjacency matrices and other real-world data usually have an extremely sparse structure.

C. Related Problems

There have also been a number of related studies to link prediction in the area of Network Completion. Basically, network completion has a more general view than link prediction where it is also assumed that there are a number of missing nodes. However, the problem of missing nodes is alleviated by using standard methods for estimating the size of missing nodes in the networks [15]. [16] uses a model-based approach where Kronecker graphs [17] is assumed to be the underlying generative model for the graphs. To infer the parameters of this model, the author presents a novel EM [18] approach based on Gibbs sampling. As mentioned before, Gibbs sampling and other MCMC methods can hardly achieve scalability and reasonable run-times. [19] explores the problem of network completion when random subsamples from the network are generated by survey sampling: choose a node in the network uniformly at random, and observe the links that node is incident with.

Link prediction may also be viewed as a special case for the Matrix Completion problem [3]. In matrix completion, the aim is to fill the missing entries of an incomplete matrix, however, it is assumed the original matrix is low-rank and there is also no consideration of adjacency matrices, e.g. binary entries, clustered and sparse structure, etc. Recently, there have been attempts in order to cast this framework to the analysis of networks. [20] uses the sum of a low-rank and a sparse matrix, representing communities and the links between them, respectively, in order to perform clustering on a partially observed graph. However, the method only considers disjoint clusters (as opposed to the findings of [21]) and it is not clear if there are scalable algorithms for it.

The problem of Network Inference, mostly in social networks, is also related to link prediction. In network inference, we are interested in inferring the hidden underlying network from the data of the diffusion of some information through the graph [22]. [23] tries to do the same task via Compressed Sensing [2], a sparse recovery framework which is recently exploited as an efficient tool for sparse signal recovery over networks [24], [25].

III. PROPOSED METHOD

In this section, first we state our problem formally. Second, our generative process as the likelihood of the network data is presented. Then, we propose our inference scheme. Considering the sparse structure of real networks, it is shown that the proposed scheme can be efficiently computed which makes it appealing for large-scale datasets.

A. Problem Formulation

Consider the observed undirected graph $G = (V, E)$, where V represents the set of nodes and E represents the set of undirected links. Suppose M is the corresponding symmetric adjacency matrix of G . We show the latent feature matrix related to the graph nodes by F , where F_i , i -th column of the latent feature matrix, is the latent feature vector of node i . Let k be the dimension of the latent feature vectors. Given

such graph, our goal is to estimate addition or removal of links by inferring the latent feature matrix F^* .

Note that for simplicity of formulations, we present our method and our evaluations for undirected networks. However, there are factorizations suggested for directed networks which may also be applicable in our method [26].

We model the network adjacency matrix M as a Poisson process and set the distribution parameter of M_{ij} to be $F_i^T F_j$:

$$M_{ij} \sim \text{Poisson}(F_i^T F_j) \quad (4)$$

Further, we assume the entries of M are conditionally independent given the latent feature vectors. Hence, the likelihood of the adjacency matrix M is:

$$P(M|F) = \prod_{i,j} \frac{(F_i^T F_j)^{M_{ij}} e^{-F_i^T F_j}}{M_{ij}!} \quad (5)$$

This factorization is applicable on undirected networks. For directed networks, as suggested in many papers like [26], we can set M_{ij} to be:

$$M_{ij} \sim \text{Poisson}(F_i^T \Lambda F_j) \quad (6)$$

where Λ is an asymmetric matrix.

Poisson distribution is mainly used in Poisson regression (in analogy with Bernoulli trials in logistic regression) in order to model count data and contingency tables [27]. Although adjacency matrices mostly comprise binary entries, they can also be considered as a special kind of counting data. Also, we observe that links in a network can be created by the occurrence of special events, e.g., friendship requests, messages, data transfer, etc., where Poisson process is suitable for modeling such events [28]. Extensive experimental investigations in [29] show that it seems Poisson model performs the same as Bernoulli model and approximating the modeling of binary graphs by Poisson likelihood does not appear to influence the ability to detect structure. See [29] for a comprehensive investigation about these models.

The idea of Bernoulli trials for the elements of adjacency matrices is explored in [9] and it is shown that the time complexity of inferring the model parameters is linear to the number of observations in each iteration. However, in the next sections, we will show that it will have scalability issues in the optimization procedure when facing with a very large adjacency matrix with millions of entries. While by using a Poisson process as the underlying generative model, we show that inferring the model parameters in each iteration will be possible in a linear time to the number of nodes which is significantly more scalable than Bernoulli generative model. To the best of our knowledge, no other link prediction method has utilized Poisson generative modeling. In the following sections, we will show that using this generative model, leads to significantly faster prediction and inference of model parameters.

For parameter estimation, we adopt a Maximum Likelihood (ML) approach in which we are seeking to infer the parameters

that maximize the likelihood of the observed data. We can write ML as:

$$\min_F lP(M|F) = \sum_{i,j} -M_{ij} \log(F_i^T F_j) + F_i^T F_j + C \quad (7)$$

where lP is the negative log-likelihood and C is a constant independent of F . The problem in (7) is a variant of matrix factorization problem. Meaning that we want to find F^* where:

$$F^* = \arg \min_F l(M, L(F)) \quad (8)$$

Comparing (7) with (8), negative log-likelihood and $F^T F$ are the corresponding loss ($l(\cdot)$) and link ($L(\cdot)$) functions, respectively. It is important to note that minimizing the negative log-likelihood of Poisson process is equivalent to minimizing generalized KL-Divergence loss [5]. Hence, $l(\cdot)$ can also be considered as the generalized KL-Divergence loss.

B. Parameter Inference

After defining the likelihood of the observed graph, now we aim to find the most likely latent factors to the corresponding observed graph.

First note that the domain of the problem in (7) is the set of matrices F for which:

$$\forall i, j \ F_i^T F_j > 0 \quad (9)$$

For simplicity of the feasible region, consider only a subset of the solutions, in which:

$$\forall i \ F_i > 0 \quad (10)$$

In addition, as [5] suggests, this helps increase interpretability and sparsity of latent factors.

Problem (7) is not jointly convex regarding all latent feature vectors. But if we fix all latent feature vectors except one, the problem becomes a convex optimization problem. Hence, we apply a block coordinate descent algorithm in order to minimize $lP(M|F)$. By fixing all latent vectors except F_i , let:

$$f(F_i) = lP(M|F) \quad (11)$$

As f is convex and there are no closed form solution $\nabla f(F_i) = 0$, the iterative gradient descent update rule for F_i will be:

$$F_i^{n+1} = F_i^{(n)} - \eta \nabla f(F_i^{(n)}) \quad (12)$$

where η is the learning rate. $\nabla f(F_i)$ is written as follows:

$$\nabla f(F_i) = 2 \sum_j F_j \left(1 - \frac{M_{ij}}{F_i^T F_j} \right) \quad (13)$$

Simplifying to:

$$\nabla f(F_i) = 2 \sum_j F_j + 2 \sum_j -\frac{M_{ij}}{F_i^T F_j} F_j \quad (14)$$

We can proceed further and remove the sentences corresponding to the zero entries of M . Thus, (14) is simplified one more step to:

$$\nabla f(F_i) = 2 \sum_j F_j + \sum_{(i,j) \in E} -\frac{2}{F_i^T F_j} F_j \quad (15)$$

A regular calculation of the gradient vector can be done in a linear time to the number of latent features $|V|$. We now seek for a method which updates F_i more quickly. To this end, we precompute the summation over all factors. Many of the matrix factorization techniques (e.g. [30], [21]) use similar ideas in order to speedup the basic algorithms.

If we look at (15), we see that if the first term was calculated, the whole gradient calculation would be done in a time linear to the number of known links incident with node i . This would be significantly faster than the regular calculation of the gradient. To this end, at the very first iteration (when nothing is inferred and the latent factors are simply initialized), we calculate the sum of the latent factors (the first term in (15)) and after each coordinate descent, we will update this summation. In other words, if F_i^{old} and F_i^{new} are the vectors before and after coordinate descent on F_i , we update the sum as:

$$\sum_j F_j \leftarrow \sum_j F_j - F_i^{old} + F_i^{new} \quad (16)$$

This way, for each coordinate, the gradient will be calculated in a time linear to the number of neighbors with the node corresponding to the current coordinate. So far, we have shown that Poisson generative model leads to efficient gradient calculation and thus, fast gradient descent. In Appendix 1., we have also shown that such gradient descent approach for a Bernoulli-Logistic model will not have this property. Hence, Poisson generative model seems to be more suitable for large-scale problems.

We have tested Newton and quasi-Newton methods in order to apply a better coordinate descent, although these methods have the extra complexity of computing the inverse of Hessian matrix. It was observed that in large datasets, the condition number of Hessian matrix turns out to be very large and makes the problem ill-posed. Therefore for line search, we use a combination of projection to ensure the feasibility of our solution and checking Wolfe condition [31] to adaptively change the step size. After each update of F_i , we project it into our feasible region (see (10)) by simply changing the negative entries to zero. This also leads us to sparser latent feature vectors. For ensuring acceptable descent in each iteration, Wolfe condition is checked. This condition can be evaluated quickly in the same manner as the gradient. A step size η can satisfy Wolfe condition if for $0 < \alpha < 1$:

$$f(F_i - \eta \nabla f(F_i)) < f(F_i) - \alpha \eta \|\nabla f(F_i)\|^2 \quad (17)$$

Expanding this condition and performing the simplifications will result in checking the following inequality:

$$\begin{aligned} \sum_{(i,j) \in E} -(\log((F_i - \eta \nabla f(F_i))^T F_j) - \log(F_i^T F_j)) \\ - \eta \nabla f(F_i)^T \sum_j F_j + \alpha \eta \|\nabla f(F_i)\|^2 < 0 \end{aligned} \quad (18)$$

Hence, (18) can also be evaluated in a time linear to the number of neighbors of node i . As the gradient is already calculated and we are ensuring the feasibility of the updated

F_i with projection, (18) can be evaluated both efficiently and correctly.

A summary of our proposed method is shown in Algorithm 1.

Algorithm 1 Scalable Link Prediction using Matrix Factorization (*SLPMF*)

Input: M (Input Matrix),
 k (Latent Space Dimension)

```

1: Initialize  $F$  //Initialization
2: Calculate  $\sum_j F_j$ 
3: repeat
4:   for  $i=1$  to  $|V|$  do //Coordinate Descent
5:     repeat
6:       Calculate the gradient  $\nabla f(F_i)$  from (15)
7:       Choose the step size  $\eta$  by checking (18)
8:        $F_i^* \leftarrow F_i - \eta \nabla f(F_i)$ 
9:        $\sum_j F_j \leftarrow \sum_j F_j - F_i + F_i^*$ 
10:      until Gradient Descent Convergence
11:   end for
12: until Coordinate Descent Convergence

```

Output: F

C. Complexity Analysis

1) *Time Complexity*: Up to line (2) of the Algorithm 1, it takes $O(k|V|)$ time to initialize and compute sum of the latent feature vectors. We assume that c , g and l are the average number of coordinate descent, gradient descent and line search iterations, respectively. For each coordinate, Line (6) takes $O(kN)$ time where N is the average number of neighbors of the network nodes. Line (7) takes $O(lkN)$ and line (8) and line (9) take $O(k)$ time, however, k is considered to be a small constant. Hence, time complexity of the algorithm is:

$$O(k|V| + c|V|g(l+1)kN) \quad (19)$$

Due to the sparsity of the networks, we can consider N to be a constant. Also for c , g and l , we observed that very small values were required to achieve acceptable convergence. Thus, the above algorithm runs in $O(|V|)$ time. In Appendix 1., it is shown that in case of using Bernoulli-Logistic model, instead of N , we would have $O(|V|)$ instead.

Another approach for the analysis of time complexity is to sum up the cost of each iteration of the loop in line (4). In each iteration of the loop in line (4), we iterate over the neighbors of each node. In other words, the i -th iteration of this loop has the cost $O(g(l+1)kN_i)$. By summing up these costs for all nodes, we will have: $O(g(l+1)k \sum_i N_i) = O(g(l+1)k|E|)$. Hence, a more accurate time complexity of the above algorithm will be:

$$O(k|V| + c|E|g(l+1)k) \quad (20)$$

However, due to the sparsity of real networks, we can consider $|E| = O(|V|)$ and therefore this time complexity reveals the same result as in (19).

2) *Space Complexity*: One of the advantages of our algorithm is that its mechanism is based on iterations over the neighbors of nodes. Hence, we can store the input M as adjacency linked list which has the space complexity of $O(|E|)$. For the variable F , we will need $O(k|V|)$ space with k being a very small constant. Overall, we can write the space complexity of the algorithm as:

$$O(k|V| + |E|) \quad (21)$$

Again, with the sparsity assumption on M ($|E| = O(|V|)$), our algorithm runs in $O(|V|)$ space complexity.

IV. EXPERIMENTAL EVALUATIONS

A. Settings

1) **Datasets**: We use a variety of real-world datasets from different areas that link prediction methods can be applied. As previously mentioned, all datasets we use are undirected. We included networks from various applications. Datasets statistics are listed in Table 1. The sparsity of the datasets is one minus the quotient of the number of ones in the adjacency matrix ($2|E|$) divided by the whole number of entries in the adjacency matrix which is $|V|^2$. The datasets are:

- Among biological networks, we use the well known protein-protein interaction network in budding yeast (denoted *Yeast*) with 2,361 nodes and 6,914 links [32].
- From social networks, a network of Facebook users with 5,793 nodes and 30,753 links (known as *FacebookLI* in the original paper) who, according to their Facebook profile page, worked in an international IT Corporation that provides products and services to customers around the world [33].
- Graph of Autonomous Systems (AS) peering information inferred from Oregon route-views between March 31 2001 and May 26 2001. This dataset consists of 11,492 nodes and 23,409 links.
- Arxiv *COND-MAT* and *GrQc* in co-authorship networks [34]. *COND-MAT* is from the e-print arXiv and covers scientific collaborations between authors papers submitted to Condense Matter category and consists of 23,133 nodes and 93,497 links. Arxiv *GrQc* collaboration network is from the e-print arXiv and covers scientific collaborations between authors papers submitted to General Relativity and Quantum Cosmology category and has 5,242 nodes and 14,490 links.
- August 24 2002 snapshot from Gnutella peer-to-peer network (denoted *P2P*¹) which has 26,518 nodes and 65,369 links [34], [35].

¹Note that P2P dataset is originally a directed network, however we converted it to an undirected one by simply changing unmatched zeroes to ones.

Table 1: Datasets Statistics

Dataset	Nodes	Links	Sparsity	Avg. Degree
<i>Yeast</i> [32]	2,361	6,914	0.99752	2.9
<i>GrQc</i> [34]	5,242	14,490	0.99891	2.8
<i>FacebookLI</i> [33]	5,793	30,753	0.99817	5.3
<i>AS</i> [34]	11,492	23,409	0.99965	2.0
<i>COND-MAT</i> [34]	23,133	93,497	0.99965	4.0
<i>P2P</i> [34], [35]	26,518	65,369	0.99981	2.5

2) **Competing Methods**: We refer to our method as *SLPMF*. Methods that we compare our performance to are:

- *Unsupervised Scores*: We picked 3 popular scores: Adamic-Adar (AA), Shortest-Path (SH) and Preferential-Attachment (PA) [36].
- *NMF with KL-Divergence loss*: There are a variety of methods in this category. We use [30] (denoted *NMF-KL*) as a recent and efficient method. The reason we chose KL-Divergence loss is that by considering the Poisson generative model for the original matrix, the maximum likelihood inference of the parameters leads to the minimization of KL-Divergence loss [5]. However, the method in [30] is suited for collaborative filtering rather than link prediction. Hence, the factorization in this work is based on equation (2) and the output is not necessarily a symmetric matrix. Our method assumes that the matrix is a graph adjacency matrix rather than a user-item matrix. We will show the effect of different factorizations in next section.
- *GLFM*: This method, presented in [9], uses Bernoulli-Logistic model as its generative model.

As mentioned in earlier sections, there are many methods using latent feature models for link prediction. However, many popular ones are outperformed by methods in [8] and [9]. Hence, we decided to compare our performance to [9] instead of all methods which is the closest to ours in general methodology. Also, the work in [1], as discussed in Section II-B, has a very long run-time that it was infeasible for us to get its results.

3) **Setup**: For all of the test cases, we randomly removed 10% of the entries of the adjacency matrix in the following way: First we choose a random entry, say (i, j) . We add (i, j) and (j, i) to the unknowns list and if (i, j) equals to 1, we set the value of (i, j) and (j, i) to zero. The resulted matrix will be the adjacency matrix of the observed graph G . Obviously, with random removal of entries, about $\frac{2|E|}{|V|^2} * \frac{|V|^2}{10} = \frac{|E|}{5}$ of the removed entries are links from the original graph. For evaluation of the methods, we use AUC (Area under ROC curve) as the measure of the quality of predictions because ROC curves are insensitive to changes in class distribution [37].

We chose a random initialization for F . All results show the average values over 10 cross-validation folds in each experiment. For implementation of our method, we used MATLAB. For unsupervised methods, we used *LPmade* package [38] and for the other methods, we used codes provided by the authors. All experiments were run on a PC with Intel Core i7 2.6GHz CPU and 12GB of RAM.

Table 2: AUC Scores Comparison - k is the latent space dimension for supervised methods (*NMF-KL*, *GLFM*, *SLPMF*). Bold numbers represent the best results

Network	k	AA	PA	SH	<i>NMF-KL</i>	<i>GLFM</i>	<i>SLPMF</i>
<i>Yeast</i>	10	0.6917 ± 0.005	0.6634 ± 0.024	0.6925 ± 0.028	0.7577 ± 0.011	0.7648 ± 0.010	0.8230 ± 0.005
<i>GrQc</i>	10	0.7554 ± 0.026	0.7302 ± 0.014	0.7411 ± 0.003	0.8118 ± 0.017	0.7331 ± 0.021	0.8600 ± 0.012
<i>FacebookLI</i>	10	0.8014 ± 0.013	0.7752 ± 0.011	0.7871 ± 0.019	0.9135 ± 0.009	0.8539 ± 0.023	0.9250 ± 0.008
<i>AS</i>	15	0.7182 ± 0.010	0.6697 ± 0.016	0.6934 ± 0.011	0.7610 ± 0.005	0.7013 ± 0.021	0.8835 ± 0.015
<i>COND-MAT</i>	25	0.7301 ± 0.012	0.7132 ± 0.014	0.7124 ± 0.013	0.8865 ± 0.016	0.7393 ± 0.011	0.9238 ± 0.017
<i>P2P</i>	30	0.6111 ± 0.005	0.6201 ± 0.005	0.6249 ± 0.020	0.6644 ± 0.031	0.6950 ± 0.014	0.7744 ± 0.021

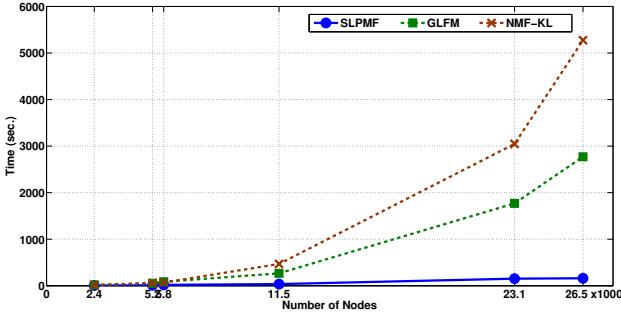


Fig. 1. Run-time Comparison - Each point corresponds to one dataset. *GLFM* and *NMF-KL* run-times become much worse on larger datasets while *SLPMF* run-time increases linearly.

4) Choosing the dimension of latent space: Clearly, the more the dimension of the latent space, the more degrees of freedom we can cover. However, smaller dimensions are preferable in terms of speed and memory issues. One approach to identify a good number (the same as in [10], [21]) of latent space dimension is to extract a hold out set (e.g. 10% of the observed graph links which is our train set) from the observed graph and run the algorithm on the rest of the dataset. The latent dimension achieving the most prediction will be a suitable one. We repeated this task for all three supervised methods (*NMF-KL*, *GLFM* and *SLPMF*) and chose the dimension that works best for all in each dataset.

B. Accuracy

The results are shown in Table 2. It can be seen that our proposed method outperforms unsupervised scores in all datasets. We also see better AUC for *SLPMF* comparing to *GLFM* and *NMF-KL*. The reason that *GLFM* cannot achieve the optimal performance of Bernoulli-Logistic generative model as it should, can be investigated in *GLFM* optimization strategy which is one of the main contributions of it [9]. In *GLFM*, an approximation to Hessian matrix through MM method is employed. However, we observed high condition numbers on this approximation (as we had in Newton-based optimizations for our own method) in our datasets which causes the descent algorithm to become slower and more inaccurate [31]. AUC of *NMF-KL* is better than *GLFM* however this method suffers from two issues that puts it behind our method: First, it is not suited for link prediction as we described and loses some of its performance because of this issue. Second, this algorithm as pointed by the authors, runs at $O(nmkd)$ time where n and

m are the matrix dimensions, k is the latent space dimension and d is the average number of gradient descent iterations. Hence, *NMF-KL* also converges slower than our method.

C. Scalability

To test for scalability, we compare the run-time of our method with *GLFM* and *NMF-KL*. Figure 1 shows the average run-times under different datasets. The graphs represent the time until convergence (where change in AUC is less than 0.001) with maximum of 10 iterations (loop in line (3) of Algorithm 1) for each method. It can be observed that *GLFM* and *NMF-KL* run-times become much worse as we test on larger datasets while our method's run-time increases linearly to the number of network nodes, as it was investigated in Section III-C1. As pointed out in the first experiment, high condition numbers in Hessian-approximations and slower optimization procedure add-up to longer run-times of *GLFM*. Also, higher algorithmic order of *NMF-KL* causes a significant drop in its run-time for larger datasets.

Figure 2 shows the resulted AUC from different time-stamps during the run-time on all datasets. Each method was evaluated for maximum of 10 iterations. As it is observed, our method converges faster and in the same running time achieves better AUC than other two methods.

D. Robustness to Data Loss

We also tested the performance of our method under different fractions of missing entries. We changed the amount of missing data from 10% to 50% on all datasets.

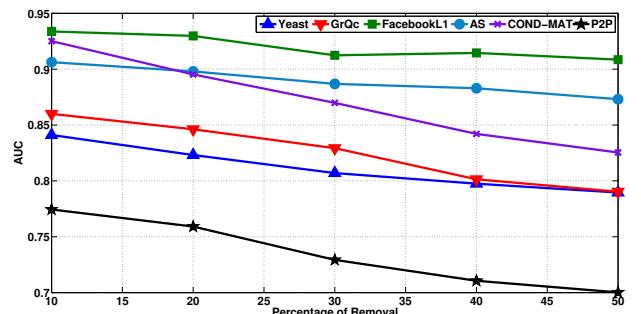


Fig. 3. Robustness to Data Loss - 10% average decrease in AUC with the amount of missing data being as high as 50% show that *SLPMF* has consistent performance with varying amount of missing data.

Figure 3 shows the results. All datasets present the same slow trend in reduction of AUC as the amount of missing

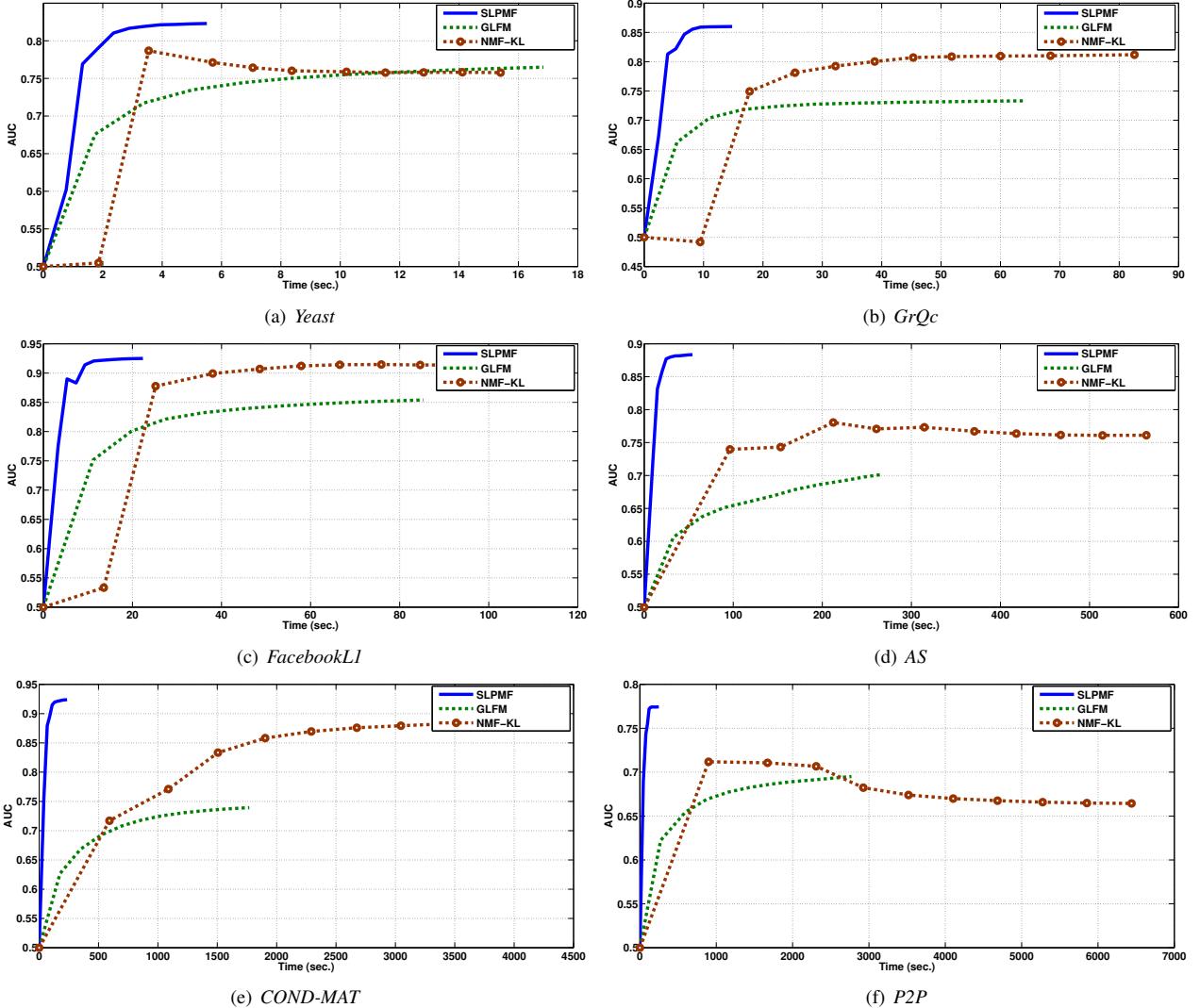


Fig. 2. AUC vs. Run-time Comparison - Results are taken for maximum of 10 iterations for each algorithm. The results show that *SLPMF* converges faster and in the same running time, *SLPMF* achieves better solutions.

entries increases. We see 10% average decrease in AUC with the amount of missing data being as high as 50%. Hence, the proposed method shows a reasonable independence from the available information.

V. CONCLUSION

In this paper, we proposed a scalable method for link prediction problem using matrix factorization techniques. We used a Poisson generative model for the networks and utilized maximum likelihood to estimate the parameters. To the best of our knowledge, previous methods for link prediction did not consider such generative model and were introduced in other models like Bernoulli-Logistic. However, according to experimental investigations in recent papers and real-world evidences, it seems networks can be modeled and analyzed as Poisson processes. By considering the sparsity of real-world networks, we showed that our method can scale easily to large-

scale datasets. Experiments show that we can achieve better performance compared to unsupervised and supervised methods. In addition, our method is significantly more scalable.

There are several directions for future work. The existence of faster coordinate and gradient descent methods can be investigated, e.g. by better line search algorithms and exploiting the sparsity of latent features. Also, parallelization of the optimization procedures can help us to speed up the algorithm.

REFERENCES

- [1] A. K. Menon and C. Elkan, "Link prediction via matrix factorization," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, vol. 6912. Springer Berlin Heidelberg, 2011, pp. 437–452.
- [2] E. Candes, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, p. 15, 2005.
- [3] E. Candes and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.

- [4] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.
- [5] Y.-X. Wang and Y.-J. Zhang, "Nonnegative matrix factorization: A comprehensive review," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 25, no. 6, pp. 1336–1353, 2013.
- [6] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Advances in Neural Information Processing Systems 20*, ser. NIPS '08, 2008.
- [7] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American Society for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, May 2007.
- [8] S. Gao, L. Denoyer, P. Gallinari, and J. Guo, "Latent factor blockmodel for modelling relational data," in *Proceedings of the 35th European conference on Advances in Information Retrieval*, ser. ECIR'13. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 447–458.
- [9] W.-J. Li, D.-Y. Yeung, and Z. Zhang, "Generalized latent factor models for social network analysis," in *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Two*, ser. IJCAI'11. AAAI Press, 2011, pp. 1705–1710.
- [10] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, "Mixed membership stochastic blockmodels," *Journal of Machine Learning Research*, vol. 9, pp. 1981–2014, Jun. 2008.
- [11] K. Lange, D. R. Hunter, and I. Yang, "Optimization Transfer Using Surrogate Objective Functions," *Journal of Computational and Graphical Statistics*, vol. 9, no. 1, 2000.
- [12] P. D. Hoff, "Multiplicative latent factor models for description and prediction of social networks," *Computational and Mathematical Organization Theory*, vol. 15, no. 4, pp. 261–272, Dec. 2009.
- [13] K. Miller, T. Griffiths, and M. Jordan, "Nonparametric Latent Feature Models for Link Prediction," in *Advances in Neural Information Processing Systems 22*, 2009, pp. 1276–1284.
- [14] J. Zhu, "Max-margin nonparametric latent feature models for link prediction," in *Proceedings of the 29th International Conference on Machine Learning*, ser. ICML '12. Omnipress, 2012.
- [15] T. H. McCormick, M. J. Salganik, and T. Zheng, "How Many People Do You Know?: Efficiently Estimating Personal Network Size," *Journal of the American Statistical Association*, vol. 105, no. 489, pp. 59–70, Mar. 2010.
- [16] M. Kim and J. Leskovec, "The network completion problem: Inferring missing nodes and edges in networks," in *SDM*. SIAM / Omnipress, 2011, pp. 47–58.
- [17] J. Leskovec, D. Chakrabarti, J. M. Kleinberg, C. Faloutsos, and Z. Ghahramani, "Kronecker graphs: An approach to modeling networks," *Journal of Machine Learning Research*, vol. 11, pp. 985–1042, 2010.
- [18] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.
- [19] S. Hanneke and E. P. Xing, "Network completion and survey sampling," *Journal of Machine Learning Research - Proceedings Track*, vol. 5, pp. 209–215, 2009.
- [20] A. Jalali, Y. Chen, S. Sanghavi, and H. Xu, "Clustering partially observed graphs via convex optimization," in *Proceedings of the 28th International Conference on Machine Learning*, ser. ICML '11. Omnipress, 2011, pp. 1001–1008.
- [21] J. Yang and J. Leskovec, "Overlapping community detection at scale: a nonnegative matrix factorization approach," in *Proceedings of the sixth ACM international conference on Web search and data mining*, ser. WSDM '13. ACM, 2013, pp. 587–596.
- [22] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," *TKDD*, vol. 5, no. 4, p. 21, 2012.
- [23] P. Siyari, H. R. Rabiee, M. Salehi, and M. E. Mehdiabadi, "Network reconstruction under compressive sensing," in *Proceedings of ASE International Conference on Social Informatics*. IEEE Computer Society, 2012, pp. 19–25.
- [24] H. Mahyar, H. R. Rabiee, and Z. S. Hashemifar, "UCS-NT: An unbiased compressive sensing framework for network tomography," in *Proceedings of 38th International Conference on Acoustics, Speech, and Signal Processing, ICASSP '13*, 2013.
- [25] H. Mahyar, H. R. Rabiee, Z. S. Hashemifar, and P. Siyari, "UCS-WN: An unbiased compressive sensing framework for weighted networks," in *Proceedings of 47th Conference on Information Sciences and Systems, CISS '13*, 2013.
- [26] A. K. Menon and C. Elkan, "A log-linear model with latent features for dyadic prediction," in *Proceedings of the 2010 IEEE International Conference on Data Mining*, ser. ICDM '10. IEEE Computer Society, 2010, pp. 364–373.
- [27] C. A. Cameron and P. K. Trivedi, *Regression Analysis of Count Data (Econometric Society Monographs)*. Cambridge, United Kingdom: Cambridge University Press, Sep. 1998.
- [28] I. Psorakis, S. J. Roberts, M. Ebden, and B. Sheldon, "Overlapping community detection using bayesian nonnegative matrix factorization," *Physical Review E*, vol. 83, no. 6, 2011.
- [29] D. Wind and M. Morup, "Link prediction in weighted networks," in *Machine Learning for Signal Processing (MLSP), 2012 IEEE International Workshop on*, 2012, pp. 1–6.
- [30] C.-J. Hsieh and I. S. Dhillon, "Fast coordinate descent methods with variable selection for non-negative matrix factorization," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '11. ACM, 2011, pp. 1064–1072.
- [31] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.
- [32] D. Bu, Y. Zhao, L. Cai, H. Xue, X. Zhu, H. Lu, J. Zhang, S. Sun, L. Ling, N. Zhang, G. Li, and R. Chen, "Topological structure analysis of the protein-protein interaction network in budding yeast," *Nucleic Acids Research*, vol. 31, pp. 2443–2450, 2003.
- [33] M. Fire, R. Puzis, and Y. Elovici, "Organizational mining using online social networks," *ArXiv Preprint*, 2012.
- [34] J. Leskovec, J. M. Kleinberg, and C. Faloutsos, "Graph evolution: Densification and shrinking diameters," *TKDD*, vol. 1, no. 1, 2007.
- [35] M. Ripeanu, A. Iamnitchi, and I. T. Foster, "Mapping the gnutella network," *IEEE Internet Computing*, vol. 6, no. 1, pp. 50–57, 2002.
- [36] L. Lu and T. Zhou, "Link prediction in complex networks: A survey," *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 6, pp. 1150 – 1170, 2011.
- [37] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, Jun. 2006.
- [38] R. N. Lichtenwalter and N. V. Chawla, "Lpmade: Link prediction made easy," *Journal of Machine Learning Research*, vol. 999888, pp. 2489–2492, Nov. 2011.

APPENDIX

A. Appendix 1: Gradient Descent under Bernoulli-Logistic Generative Model

Bernoulli-Logistic generative model has the following form:

$$P(M|F) = \prod_{i,j} (\sigma(F_i^T F_j))^{M_{ij}} (1 - \sigma(F_i^T F_j))^{1-M_{ij}} \quad (22)$$

where $\sigma(\cdot)$ is sigmoid function. Negative log-likelihood is:

$$lP(M|F) = \sum_{i,j} -M_{ij}(F_i^T F_j) + \log(1 + F_i^T F_j) + C \quad (23)$$

By $f(F_i) = lP(M|F)$ and computing the gradient, we get:

$$\nabla f(F_i) = \sum_{i,j} -M_{ij} F_j + \frac{F_j}{1 + F_i^T F_j} \quad (24)$$

Obviously, (24) cannot be computed with the technique used in (15) and requires $O(|V|)$ time to get computed.

Scalable Link Prediction: A Case for Nonnegative Matrix Factorization under Generalized KL-Divergence Loss

ارسال شده به ژورنال

IEEE Journal of Selected Topics in Signal Processing: Special Issue on Signal Processing for Social Networks - IF: ۳.۲۹۷

Abstract: Link prediction is a fundamental problem that appears in many applications such as recommender systems and biological networks. In this paper, we exploit the sparsity (*i.e.*, small number of links) of real-world networks in order to introduce a new link prediction method which is improved in scalability and accuracy than the existing state-of-the-art algorithms. We take a probabilistic approach by assigning a Poisson generative model to the underlying network, and present link prediction problem as a variant of matrix factorization problem. To the best of our knowledge, no previous method has used Poisson process for link prediction to improve scalability and accuracy. It is shown that the proposed method leads to fast inference of the parameters, and hence easily scales to large real-world networks. Moreover, it is possible to use the proposed method in several applications other than link prediction such as collaborative filtering, latent semantic indexing and dimensionality reduction. In addition to robustness and fast convergence, experiments show that the proposed method outperforms the popular unsupervised and supervised methods while being significantly more scalable.

Abstract

During the last decade, there have been a great number of researches on complex networks. Data aggregation is the first step in the analysis of these networks. However, due to the large scale of them, almost never is there complete information about a network's different aspects. Therefore, analysis of a complex network is usually done based on the incomplete data. Although a good sampling approach in a way that the achieved sample is a good representative of the whole network has its own challenges, analysis of incomplete data causes a significant alteration in the estimation results. Consequently, one of the first problems emerging after sampling is the possibility of predicting the hidden part of the network. The main approach to solve such problem is a model-based one in which, there is an assumption for the parametric model of the network and the effort is to estimate this model's parameters. Depending on the data that we exploit to infer the hidden part of a network, presented methods may differ in their general approach. The methods presented in this thesis rely on the sparse structure of the real-world networks that is the most common pattern in almost all real-world networks.

As the first problem, we address the Network Reconstruction Problem: Given a network with missing edges, how is it possible to uncover the network structure based on certain observable quantities extracted from partial measurements? We propose a novel framework based on a newly emerged paradigm in sparse signal recovery called Compressive Sensing (CS). The results demonstrate that our framework can perform accurately even on low number of cascades (e.g. when the number of cascades is around half of the number of existing edges in the desired network). In addition, we compared the performance of our framework with NetInf; one of the state-of-the-art methods in inferring the networks of diffusion. The results suggest that the proposed method outperforms NetInf by an average of 10% improvement based on the F-measure. Furthermore, the proposed method can be generalized to other areas such as detecting congested links in computer networks.

Second, we propose a scalable method for link prediction problem. Link prediction is a fundamental problem that appears in many applications such as recommender systems and biological networks. In this thesis, we exploit the sparsity (i.e., small number of links) of real-world networks in order to introduce a new link prediction method which is improved in scalability and accuracy than the existing state-of-the-art algorithms. We take a probabilistic approach by assigning a Poisson generative model to the underlying network, and present link prediction problem as a variant of matrix factorization problem. To the best of our knowledge, no previous method has used Poisson process for link prediction to improve scalability and accuracy. It is shown that the proposed method leads to fast inference of the parameters, and hence easily scales to large real-world networks. In addition to robustness and fast convergence, experiments show that the proposed method outperforms the popular unsupervised and supervised methods while being up to 15 times faster than previous methods. Moreover, it is possible to use the proposed method in several applications other than link prediction such as collaborative filtering, latent semantic indexing and dimensionality reduction. We also present greedy and parallel versions of the proposed method which are suitable for collaborative filtering applications.

Keywords: *Complex Networks, Topology Inference, Network Sampling, Network Reconstruction, Compressed Sensing, Network Model, Link Prediction, Matrix Factorization*



Sharif University of Technology
Computer Engineering Department

Master of Science Thesis
Software Engineering

**Network Topology Inference from
Incomplete Data**

By
Payam Siyari

Supervisor
Hamid R. Rabiee

Summer 2013