دانشگاه صنعتی شریف

دانشکده‌ی مهندسی کامپیوتر

پایان‌نامه‌ی کارشناسی

گرایش مهندسی نرم‌افزار

عنوان:

خوشه بندی چنددیدگاهی و بی نظارت داده‌های ژنتیکی با شبکه‌ی خودرمزگذار گراف

نگارش:

دانیال ملک‌محمد

استاد راهنما:

دکتر محمدامین فضلی

شهریور ۱۳۹۹


بسم الله الرحمن الرحیم

به نام خدا

دانشگاه صنعتی شریف

دانشکده‌ی مهندسی کامپیوتر

پایان‌نامه‌ی کارشناسی

عنوان: خوشه بندی چنددیدگاهی و بی نظارت داده‌های ژنتیکی با شبکه‌ی خودرمزگذار گراف

نگارش: دانیال ملک‌محمد

کمیته‌ی ممتحنین

استاد راهنما: دکتر محمدامین فضلی امضاء:

استاد مشاور: دکتر محمدحسین رهبان امضاء:

استاد مدعو: استاد ممتحن امضاء:

تاریخ:

چکیده

در بسیاری از کاربردها چندین دسته ویژگی از جنس‌های مختلف برای نمونه‌ها موجود است که اطلاعاتشان می‌تواند برای یکدیگر تکمیل‌کننده باشد. به‌ویژه در حوزه‌ی بیوانفورماتیک تکنولوژی‌های اخیر امکان جمع‌آوری داده‌های ژنومیک زیاد و گوناگونی را فراهم کرده است. این امر موجب پراهمیت شدن روش‌هایی شده که به جای آموزش متریکی روی یک مجموعه داده چندین مجموعه داده از جنس های مختلف را به‌کار می‌گیرند. در این مقاله سعی‌شده که چندین روش مبتنی بر شبکه‌های عصبی گراف کانولوشن ارائه شود که هر یک از مجموعه دادگان را به عنوان یک گراف درنظربگیرد و با تجمعیع آن‌ها به تک گرافی جامع دست یابد. در این پژوهش از سه مجموعه دادگان Methylation mRNA DNA و Micro RNA استفاده شد که مربوط به ۵ مجموعه داده‌ی سرطانی بود. عمده‌ی پژوهش روی دادگان سرطان GBM انجام شد که دارای ۲۱۵ مریض بود و هر مریض به عنوان یکی از رئوس گراف مشابهت فرض شد. پس از بدست آوردن گراف نهایی روی آن خوشه‌بندی طیفی انجام شد تا برزسیتی مریض‌ها پیش‌بینی شود.

کلیدواژه‌ها:

Multi-view Learning, Data Fusion, Data Aggregation, Graph Convolutional Networks, Survival Analysis, Spectral Clustering, Graph Autoencoder, Autoencoder, Variational Autoencoder

فهرست مطالب

فهرست شکل‌ها

فهرست جدول‌ها

فصل ۱

مقدمه

تکنولوژی های تولید داده‌های ژنومیکی درحال پیشرفت سریعی هستند و این امر سبب ایجاد داده‌های زیاد و گوناگونی در حوزه‌ی بیولوژیکی و درمانی شده است. برای مثال مجموعه دادگان بزرگ TCGA [۱] اطلاعات ژنتیکی بیش از ۲۰ نوع سرطان و هزاران مریض را داراست. دردسترس بودن این چنین دادگان غنی سبب شده که روش های یادگیری مبتنی بر ترکیب که گوناگونی دادگان و روندهای زیستی را به‌کار می‌گیرند پراهمیت شوند. اما چالش‌هایی در این حوزه وجود دارند.

۱. داده‌های ژنومیکی معمولا دارای ابعاد ویژگی بسیار بزرگی هستند. مثلا مجموعه داده بیان ژنی که در این پژوهش بکار رفت دارای ۱۲۰۴۲ بعد بود که در مقایسه با ۲۱۵ نمونه بیمار عدد بزرگی است. این درحالی است که نمونه ها به صورت ذاتی در فضاها و منیفلدهای با ابعاد بسیار کوچکتری هستند. در روش ما برای حل این مسئله دادگان را با استفاده از خودرمزگذار ها به ابعاد بسیار کمتری مثل ۱۰۰ بعد می‌بریم.

۲. متفاوت بودن مقیاس، بایاس [۲] جمع‌آوری و نویز و ابعاد ویژگی در هر مجموعه داده: داده‌های ژنتیکی مختلف ابعاد متفاوتی دارند همچنین تفاوت ماهیت بیولوژیکی و تکنولوژی جمع آوری هریک سبب می‌شود نوع توزیع، ویژگی‌های آماری و حتی اندازه‌ی بزرگی اعداد هر مجموعه‌داده با بقیه متفاوت شود. از این‌رو ایده‌های ساده‌ای همچون میانگین‌گیری همه‌ي مجموعه‌دادگان یا الحاق ابعاد تمام ویژگی‌ها نمی‌توانند منجر به مجموعه‌داده‌ای واحد شوند که اثر همه‌ی دیدها در

[۱] The Cancer Genome Atlas
[۲] سوگیری

آن‌ها به یک‌اندازه باشد و مجموعه‌داده‌ای غالب نشود. برای حل این چالش به جای مطالعه‌ی خود دادگان، ماتریس شباهت [۳] نرمالایز شده‌شان(ماتریس مجاورت گراف مریض‌ها) را به‌کار می‌گیریم.

۳. آموزش روی هر یک از مجموعه دادگان به تنهایی منجر به دقت‌های متفاوتی می‌شود و هدف از روش‌های ترکیبی به‌کارگیری کل دادگان برای رسیدن به دقتی بالاتر از دقت آموزش روی تک تک دادگان است. اما این ترکیب شدن الزاما به معنای توافق همگانی [۴] نیست. توافق در مسائلی همچون رای‌گیری [۵] کاربرد دارد و با روش های ساده‌ای همچون میانگین، مد و... قابل انجام است. اما در این پژوهش فرض شده که هر مجموعه داده جنبه‌ای ناقص از واقعیت کلی را نشان‌می‌دهد. برای تمثیل فرض کنید واقعیت کلی که از آن بی خبر هستیم یک مکعب است و سه مجموعه‌داده داریم که هرکدام یک تصویر دوبعدی تقریبا عمود بر دوتای دیگر از این مکعب هستند. در این مثال با اجرای توافق همگانی به یک چهار ضلعی دست خواهیم یافت که غلط است. برای حل این چالش از ولگشت [۶] و واپخش [۷] در گراف بهره می گیریم تا گراف‌های ناقص و نویزی را ترکیب کرده و به تک گرافی جامع‌تر و دقیق‌تر برای تحلیل دست‌یابد.

۴. در اغلب اوقات خوشه‌ها در فضای ویژگی به صورت محدب از یکدیگر جداپذیر نیستند و روش‌های متداول خوشه‌بندی مثل kmeans خوشه‌های محدب ارائه می‌دهند. برای رفع این چالش kmeans را به جای اینکه در دامنه‌ی ویژگی‌ها اجرا کنیم در دامنه‌ی طیفی گراف شباهت مبتنی بر ویژگی‌ها انجام می‌دهیم. این روش مرسوم به خوشه‌بندی طیفی [۸] است.

۵. روش‌های متداول ترکیب مجموعه‌دادگان عمدتا مبتنی بر یادگیری ماشین کلاسیک هستند و از ایده‌ی یادگیری عمیق استفاده نمی‌کنند. این درحالی است که روش‌های مبتنی بر یادگیری عمیق در چندسال اخیر نتایج بسیار بهتری در اکثر حوزه‌ها کسب کرده‌اند. از این‌رو روش پیشنهادی ما مبتنی بر شبکه‌های یادگیری عمیق با لایه‌های GCN [۹] بود. که اخیرا بسیار موردتوجه قرار گرفته‌اند.

---

[۳] Similarity Matrix
[۴] Consensus
[۵] Voting
[۶] Random Walk
[۷] diffusion
[۸] Spectral Clustering
[۹] Graph Convoloutional Network

این نوشتار گزارش تلاش‌های ما برای آموزش بدون سرپرستی [۱۰] شبکه‌های عمیق روی مجموعه دادگان به‌کار گرفته‌شده در [۳] است. عمده‌ی بررسی‌ها روی داده‌ی سرطانی GBM بود که مربوط به ۲۱۵ بیمار مبتلا است. به ازای هر مریض اطلاعات مربوط به بیان ژنی (۱۲۰۴۲ بعد)، متیلیشن دی.ان.ای (۱۳۰۵ بعد) و میکرو آر.ان.ای (۵۳۴ بعد) موجود است. برچسب مریض‌ها هم تعداد روزهای برزیستی [۱۱] آن ها است. ابتدا از هرکدام از هرکدام از سه مجموعه داده یک ماتریس مشابهت به‌دست می‌آید. سپس تعدادی معماری شبکه‌ی خودرمزگذار با لایه‌های GCN آزموده می‌شود که از ماتریس لاپلاسین این سه گراف به عنوان کرنل [۱۲] و از خود دادگان به عنوان سیگنال درآن‌ها استفاده می‌شود. درنهایت ماتریس لاپلاسین ترکیب‌شده و embedding ها مورد مطالعه قرار میگیرند، با روش‌هایی مثلی مثل خوشه‌بندی طیفی خوشه‌بندی می‌شوند و صحتشان با استفاده از مدل Likelihood Cox Partial تست فرضیه می‌شود تا P Value بدست آمده با [۳] مقایسه شود. هدف این است که بیماران هر خوشه توزیع برزیستی یکسانی با یکدیگر و متفاوت با خوشه‌های دیگر داشته‌باشند.

[۱۰] Unsupervised
[۱۱] Survival
[۱۲] Kernel

فصل ۲

مفاهیم اولیه

۲ـ۱ واپخشی

در این نوشتار و مقاله‌های مشابه از هریک از مجموعه‌دادگان به‌عنوان یک دید [۱] یاد می شود. در پژوهش ما ۳ دید $X_۱, X_۲, X_۳$ وجود داشت که هریک $N \times d_v$ بعدی بود و سطر j ام در همه‌ی آن‌ها مربوط به نمونه بیمار jام است. از روی هر دید می‌توان یک ماتریس مشابهت ساخت. یک روش معمول و پایه‌ای استفاده از کرنل گاوسی است.

$$A_v(i,j) = exp(-\frac{\rho(x_i, x_j)^۲}{\sigma^۲}) \qquad (۱ـ۲)$$

که $\rho(x_i, x_j)$ فاصله‌ی اقلیدسی بیمار i و j است.

ماتریس $A_v \in \mathbb{R}^{N \times N}$ را می‌توان ماتریس مجاورت گرافی فرض کرد که راس‌هایش مریض ها هستند و وزن یال‌هایش میزان شباهت بین مریض‌ها مطابق دید v ام است. برای هر گراف ماتریس لاپلاسین به صورت زیر تعریف می شود.

$$L = D - A \qquad (۲ـ۲)$$

ماتریس قطری $D \in \mathbb{R}^{N \times N}$ ماتریس درجات نام‌دارد؛ درایه‌ی (i,i) آن درجه راس i ام است. همچنین

[۱] View

ماتریس لاپلاسین متقارن و نرمالایز شده به‌صورت زیر تعریف می‌شود.

$$\tilde{L} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} \qquad (۳\_۲)$$

حال می‌خواهیم فرآیند واپخشی در گراف را به صورت ریاضی تعریف کنیم. فرض کنید به هر راس گراف یک مقدار اسکالر نسبت داده‌ایم و قصد داریم این مقادیر را از طریق یال‌ها پخش کنیم. بردار $x \in \mathbb{R}^N$ را بیانگر این مقادیر تعریف کنید. با ضرب ماتریس لاپلاسین در آن مقادیر رئوس از طریق یال‌ها و باتوجه به وزنشان در گراف پخش می‌شوند. بردار

$$x' = Lx \qquad (۴\_۲)$$

مقادیر رئوس بعد از پخش شدن است. برای درک بهتر این فرآیند ضرب داخلی رئوس قبل و بعد از پخش شدن را درنظر. به راحتی می‌توان ثابت کرد

$$\langle x, x' \rangle = x^\top L x = \sum_{(u,v)\in E} A(u,v)(x(u) - x(v))^2 \qquad (۵\_۲)$$

طبق این رابطه فرآیند واپخشی به فراخور وزن یال‌ها اتفاق می‌افتد، یعنی همسایگان راس v مقادیر خود را از طریق یال واصلشان به این راس می دهند و هرچقدر وزن یال واصل و یا مقدار اسکالر همسایه بیشتر باشد، عدد بیشتری از آن همسایه به v می‌رسد. اگر به‌جای L از $\tilde{L}$ استفاده کنیم مقداری که از u به v می‌رسد به‌جای اینکه به $A(u,v)$ بستگی داشته باشد، متاثر از $\frac{A(u,v)}{deg(u)}$ خواهدبود (اصطلاحا برحسب مجموع یال‌های خروجی از هر راس نرمالایز می‌شود). همچنین اگر از L^k به جای L استفاده‌شود در فرآیند واپخشی به جای اینکه فقط مقادیر رئوس همسایه‌ی یک راس به آن سرازیر شوند، مقادیر رئوس با فاصله‌ی حداکثر k به آن سرازیر می‌شوند.

به سادگی می‌توان مفهوم واپخشی در گراف را از انتساب فقط یک اسکالر به هر راس، به انتساب یک بردار d بعدی به هر راس تعمیم داد (هربعد این بردار مستقل از ابعاد دیگر مطابق آنچه گفته شد پخش شود).

$$X' = LX \quad \& \quad X \in \mathbb{R}^{N\times d} \qquad (۶\_۲)$$

در فرآیند واپخشی دو نوع دامنه وجود دارد: دامنه‌ی فضای برداری d بعدی ویژگی‌ها (دامنه‌ی سیگنال)

و دامنه‌ی فضای برداری N بعدی رئوس گراف (دامنه‌ی طیفی [۲]). فرآیند واپخشی همانگونه که دیدیم در دامنه‌ی اسپکترم به صورت یک عمل خطی تعریف شد اما تفسیر آن در دامنه‌ی سیگنال به این سادگی نیست.

برای درک بهتر فرآیند واپخشی جالب است بدانید که تمامی تعاریف این حوزه مشابه فوریه در مبحث سیگنال است؛ دامنه‌ی سیگنال و دامنه‌ی طیفی در وابخشی به ترتیب دامنه‌ی زمانی و دامنه‌ی فرکانسی در مبحث سیگنال است. انجام فرآیند واپخشی با ماتریس لاپلاسین هم همان به‌کارگیری فیلتر فرکانسی است.

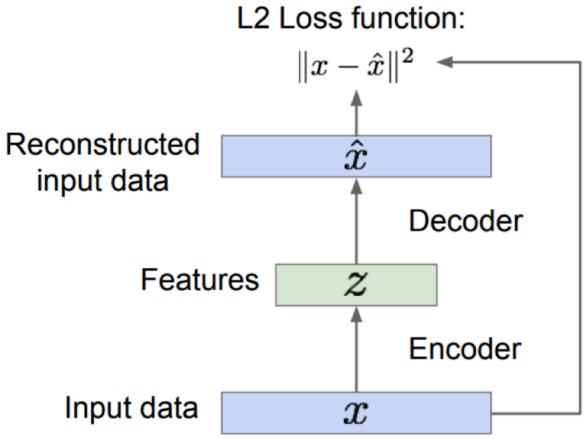
۲ـ۲ خودرمزگذار

خودرمزگذارها [۳] گونه‌ای از شبکه‌های عمیق هستند که به صورت بدون سرپرستی یک embedding (معمولا در ابعادی کمتر از ابعاد داده‌ي اصلی) را یاد می‌گیرند. این شبکه‌ها دارای دو قسمت هستند: بخش رمزگذار [۴] و رمزگشا [۵] .داده ابتدا به رمزگذار داده می‌شود تا طی چندین لایه ابعادش کاهش یابد، سپس رمزگشا این نمایش در ابعاد پایین را دوباره به ابعاد اولیه می برد و سعی می‌کند داده‌ی اصلی را بازسازی‌کند. شکل ۲ـ۱ نمای کلی این شبکه‌ها را نشان می‌دهد.

یک تابع ضرر متداول برای این شبکه ها $(X - X')^2$ می‌تواند باشد که X ورودی و X' خروجی حاصل از بازسازی رمزگشا است.

تجربه نشان داده‌است که خودرمزگذار ساده الزاما نمونه‌ها را به صورت خوش تعریف در فضای embedding رمزگذاری نمی‌کند، بلکه ممکن است هر داده را به نقطه‌ای تصادفی در فضای embed-ding ببرد به صورتی که فاصله‌ی نقاط و محلیت در این فضا معنای خاصی نداشته‌باشد، به عبارتی دیگر خودرمزگذار صرفا به عنوان یک تابع هش عمل کند. این درحالی است که ما در بسیاری از کاربردها علاقه‌مندیم که فضای embedding دارای خواصی همچون پیوستگی و همواری [۶] باشد، فاصله‌ها بامعنی باشند، داده‌های مشابه در فضای embedding نزدیک هم باشند و حرکت در یک جهت خاص از embedding موجب نوعی تغییر پیوسته در ویژگی‌ای از نمونه‌ها شود. برای حل این مشکل از شبکه‌ی

[۲] Spectral
[۳] Autoencoder
[۴] Encoder
[۵] Decoder
[۶] Smoothness



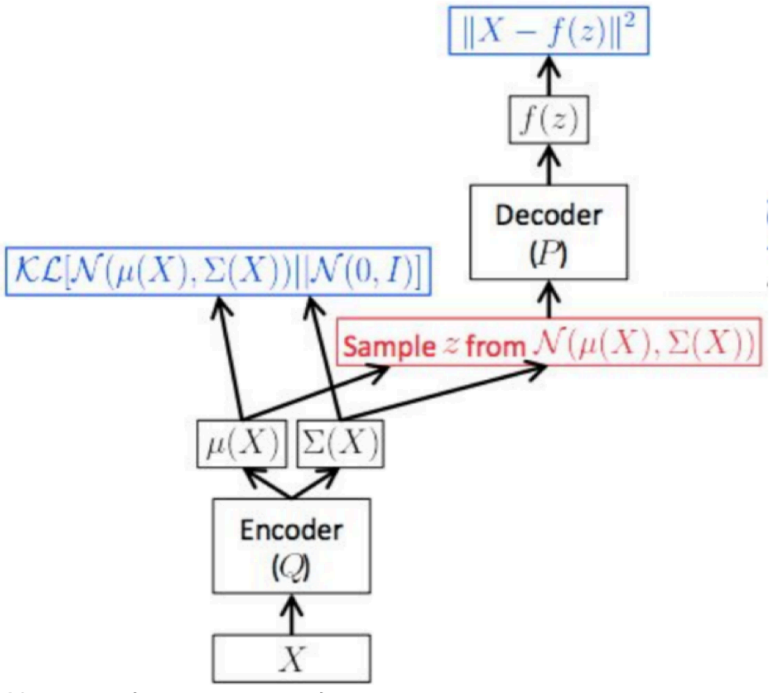
شکل ۲ـ۱: نمای کلی یک خودرمزگذار [۱]

خودرمزگذار متغیر [۷] استفاده می‌شود. این شبکه نیز دارای دو قسمت رمزگذار و رمزگشا است با این تفاوت که نقاط در فضای embedding به‌صورت احتمالاتی تصور می‌شوند؛ رمزگذار داده را به جای اینکه به یک نقطه در فضای embedding ببرد، به یک توزیع نرمال چندبعدی [۸] در آن فضا متناظر می‌کند. از این‌رو قسمت رمزگذار دارای دو شبکه است: یکی برای ارائه‌ی μ یا میانگین توزیع و دیگری برای ارائه‌ی ماتریس کواریانس Σ توزیع. سپس رمزگشا مطابق توزیع یک نمونه تصادفی ایجاد می‌کند و از روی آن داده‌ی اصلی را بازسازی می‌کند. شکل ۲ـ۲ نمای کلی این شبکه‌ها را نشان می‌دهد. تابع هدف در این شبکه‌ها به صورت

$$E_z[\log p_\theta(x^{(i)}|z)] - D_{KL}(q_\phi(z|x^{(i)})||p_\theta(z)) \qquad (۲ـ۷)$$

که عبارت اول برای بازسازی است و عبارت دوم سعی می‌کند توزیعی که هرنمونه در فضای embedding پیدا می‌کند شبیه توزیع پیشین (معمولا $\mathcal{N}(۰, \sigma^۲ I)$) باشد.

[۷] Variational Autoencoder
[۸] Multivariate Normal



شکل ۲ـ۲: نمای کلی یک خودرمزگذار متغیر [۲]

۲ـ۳ خوشه‌بندی طیفی

تعریف ۲ـ۱ (برش) *در گراف مفهوم برش*[۹] *برای دو مجموعه رئوس* A *و* B *به‌صورت*

$$cut(A, B) = \sum_{i \in A, j \in B} w_{ij} \qquad (۸ـ۲)$$

تعریف می‌شود.

تعریف ۲ـ۲ (RatioCut) *برای* k *مجموعه های* A_k .. $A_۱$ *به صورت*

$$RatioCut(A_۱, ..., A_k) = \sum_{i=۱}^{k} \frac{cut(A_i, \overline{A_i})}{|A_i|} \qquad (۹ـ۲)$$

تعریف می‌شود که $|A_i|$ *تعداد رئوس مجموعه‌ی* A_i *است.*

بر اساس RatioCut می‌توان یک نوع خوشه‌بندی برای گراف تعریف کرد:

یافتن افرازی k مجموعه‌ای (k خوشه) به صورت A_k .. $A_۱$ که مقدار $RatioCut(A_۱, ..., A_k)$ شان کمینه باشد.

[۹] Cut

خوشه‌بندی طیفی در حقیقت نسخه‌ی relaxed شده‌ی این مسئله است که در ادامه توضیح داده‌می‌شود.

ماتریس لاپلاسین طبق ۲ـ۵ مثبت نیمه معین بوده و متقارن بودنش نیز پرواضح است از این رو همواره L را می‌توان به‌صورت

$$L = QSQ^\top \qquad (۱۰ـ۲)$$

تجزیه کرد که Q ماتریس متعامد و S ماتریس قطری است. در واقع هرستون Q یکی از بردار ویژه‌های L و درایه‌ی متناظرش در S مقدار ویژه‌اش است. یک الگوریتم برای خوشه‌بندی طیفی بدین صورت است که k تا بردار ویژه‌ی مربوط به k تا کوچک‌ترین مقدار ویژه‌های ماتریس لاپلاسین را نگه می‌داریم و بقیه را دور می‌ریزیم.

$$L' = Q_{[۱:k]} S_{[۱:k]} Q^\top_{[۱:k]} \qquad (۱۱ـ۲)$$

سپس در فضای k بعدی جدید متشکل از این k تا بردار ویژه $(Q_{[۱:k]})$ الگوریتم kmeans را اجرا می‌کنیم تا رئوس گراف خوشه‌بندی شوند.

یک راه دیگر برای اجرای خوشه‌بندی طیفی حل

$$min_Y\{\mathbf{Trace}(Y^\top LY)\} \qquad (۱۲ـ۲)$$

با شرط

$$Y \in \mathbb{R}^{N\times k} \;\&\; Y^\top Y = I_k \qquad (۱۳ـ۲)$$

است (هر ستون ماتریس Y معرف یکی از خوشه‌ها می‌شود).

به عنوان نکته‌ی آخر اگر در دو الگوریتم مذکور برای خوشه‌بندی طیفی از ماتریس لاپلاسین نرمالایز شده به جای ماتریس لاپلاسین استفاده کنیم Ncut کمینه می‌شود.

تعریف ۲ـ۳ (Ncut)

$$Ncut(A_۱, ..., A_k) = \sum_{i=۱}^{k} \frac{cut(A_i, \overline{A_i})}{vol(A_i)} \qquad (۱۴ـ۲)$$

تعریف می‌شود که

$$vol(A) = \sum_{i\in A} d_i \qquad (۱۵ـ۲)$$

فصل ۳

کارهای پیشین

۳ـ۱ روش Cross Diffusion

پژوهش [۴] در سال ۲۰۱۲ این روش را برای ترکیب چندین مجموعه‌داده ارائه داد. در آن ابتدا براساس هر مجموعه‌داده یک ماتریس مشابهت می‌سازیم.(یک روش معمول رابطه‌ی ۳ـ۱ است.)

$$\mathcal{P}(i,j) = \frac{W(i,j)}{\sum_{k \in V} W(i,k)} \qquad (۱ـ۳)$$

سپس این ماتریس مشابهت را به ماتریس مارکوف تبدیل می‌کنیم. در ادامه برای $t = ۱..T$ بار ادغام را به صورت

$$P^{(i)}_{t+۱} = \mathcal{P}^{(i)} \times (\frac{۱}{m-۱} \sum_{j \neq i} P^{(j)}_t) \times {\mathcal{P}^{(i)}}^{\top} \qquad (۲ـ۳)$$

انجام می‌دهیم. توجه‌کنید که $\mathcal{P}^{(i)}$ ماتریس مارکوف حاصل از دید i ام است و

$$P^{(i)}_{t=۱} = \mathcal{P}^{(i)} \qquad (۳ـ۳)$$

درنهایت ماتریس نهایی به صورت

$$P^{(c)} = \frac{۱}{m} \sum_{i=۱}^{m} P^{(i)}_T \qquad (۴ـ۳)$$

بدست می‌آید. برای درک شهودی این روش حالتی که دو مجموعه‌داده داشته‌باشیم را درنظر بگیرید.

$$P^{(۱)}_{t+۱} = \mathcal{P}^{(۱)} \times P^{(۲)}_t \times {\mathcal{P}^{(۱)}}^{\top} \qquad (۵ـ۳)$$

$$P^{(۲)}_{t+۱} = \mathcal{P}^{(۲)} \times P^{(۱)}_{t} \times \mathcal{P}^{(۲)^\top} \quad (۶\_۳)$$

زمانی که $t = ۱$ باشد، سمت راست رابطه‌ی ۳ـ۶ احتمال مسیرهای به طول ۳ ای را درنظر می‌گیرد که یال اول و سوم آن‌ها مطابق گراف دید ۱ و یال دوم مطابق گراف دید ۲ است. حال اگر T را برای مقادیر بزرگتری درنظر بگیریم مسیرهایی درنظر گرفته می‌شوند که به طور یک درمیان طبق گراف دید ۱ و دید ۲ می‌شوند و بدین ترتیب گراف دو مجموعه‌داده ادغام می‌شوند. در [۳] از همین این روش استفاده‌شد.

۳ـ۲ خودرمزگذار متغیر گراف

این نوع از شبکه‌ی خودرمزگذار توسط کیپف و ماکس‌ولینگ در سال ۲۰۱۶ معرفی‌شد [۵] و به‌سرعت موردتوجه قرار گرفت. در این پژوهش از یک گراف بدون جهت استفاده شد که هر راس آن یک مقاله بود و وجود یال بین دو راس به معنای ارجاع دهی یکی توسط دیگری بود. علاوه‌بر آن تمام مقالات دارای یک بردار صفرویکی بودند که هر درایه وجود یا عدم وجود کلمه‌ای از دیکشنری در آن مقاله را نشان می‌داد. در این پژوهش حدود ٪۱۰ از یال‌های گراف ارجاع‌دهی به عنوان داده‌ی آزمایش حذف شدند و با استفاده‌از بردار کلمات مذکور این یال‌های با دقت ٪۹۰ بازسازی شدند. این خودرمزگذار در هر ایپوک [۱] یادگیری تمام بردارهای مربوط به وجود یا عدم وجود کلمات دیکشنری در هرمقاله را می‌گیرد(این ماتریس $X \in \mathbb{R}^{N \times d}$ توصیف مقالات در دامنه‌ی سیگنال است).

تفاوت اصلی این شبکه با شبکه‌های عادی خودرمزگذار استفاده از لایه‌های GCN بود. یک لایه MLP ساده به صورت

$$MLP(X) = ReLU(XW) \quad (۷\_۳)$$

اما در لایه‌ی GCN عمل واپخشی نیز انجام می‌شود.

$$GCN(X, A) = ReLU(\tilde{A}XW) \quad (۸\_۳)$$

$$\tilde{A} = D^{-\frac{۱}{۲}} A D^{-\frac{۱}{۲}} = I - \tilde{L} \quad (۹\_۳)$$

[۱] Epoch

بخش رمزگذار دارای دو لایه GCN است و بخش رمزگشا هدفش بازسازی ماتریس مجاورت A است، به این صورت که اگر z_i و z_j نمایش دو راس i و j در embedding باشند،

$$p(A_ij = ۱|z_i, z_j) = \sigma(z_i^\top z_j) \qquad (۱۰\_۳)$$

است. از این‌رو رمزگشا $\hat{A} = \sigma(ZZ^\top)$ را به عنوان خروجی ارائه می‌کند و تلاش دارد $\hat{A}$ شبیه A شود.

۳_۳ استفاده از خودرمزگذار برای پیشبینی برزیستی سرطان کبد

پژوهش [۶] در سال ۲۰۱۷ از خودرمزگذارهای ساده برای ترکیب ۳ مجموعه داده‌ی RNA-seq، miRNA-seq و methylation برگرفته از TCGA استفاده کرد تا برزیستی ۳۶۰ نمونه بیمار مبتلا را پیشبینی کند. در این پژوهش هرسه مجموعه داده الحاق شدند و خود رمزگذار با تابع هزینه‌ی

$$logloss(x, x') = \sum_{k=۱}^{d}(x_k log(x'_k) + (۱ - x_k) log(۱ - x'_k)) \qquad (۱۱\_۳)$$

، رگولاریزیشن L۱ روی وزن‌های شبکه، رگولاریزیشن L۲ روی خروجی تابع‌های فعال‌سازی شبکه و لایه‌های dropout با نرخ ٪۵۰ آموزش داده‌شد. معماری خودرمزگذار به‌صورت سه لایه‌های با ابعاد به ترتیب ۵۰۰, ۱۰۰, ۵۰۰ بود. بعد از آموزش شبکه از embedding به عنوان ویژگی‌های جدید استفاده گردید و با مدل تک بعدی Cox Partial Likelihood ابعادی از آن که p value قابل توجه (log-rank p-value < ۰/۰۵) داشتند انتخاب شدند. سپس در محیط ویژگی جدید خوشه‌بندی kmeans انجام شد تا برچسب‌های خوشه‌بندی به‌دست آیند. در نهایت ابعادی از ویژگی اولیه که بیشترین کورولیشن با برچسب‌های خوشه‌بندی را داشتند انتخاب شدند تا الگوریتم SVM روی آن‌ها آموزش داده‌شود (۱۰۰ بعد از mRNA ، ۵۰ بعد از methylation و ۵۰ بعد از miRNA).

۳ـ۴ چهارچوب SIMLR

در سال ۲۰۱۷ [۷] چهارچوب مبتنی بر بهینه‌سازی [۲] SIMLR [۳] را برای ترکیب چندین کرنل ارائه داد. در این پژوهش با استفاده از روش بهینه‌سازی تابع هزینه‌ی

$$-\sum_{i,j,l} w_l K_l(c_i, c_j) S_{ij} + \beta \|S\|_F^2 + \gamma \, \mathbf{tr}(L^\top (I_N - S) L) + \rho \sum_l w_l \log w_l \qquad (۱۲ـ۳)$$

با شروط

$$L^\top L = I_C, \; \sum_l w_l = ۱, \; w_l \geqslant ۰, \sum_j S_{ij} = ۱ \; \& \; S_{ij} \geqslant ۰ \qquad (۱۳ـ۳)$$

کمینه می‌شود که K_l ها کرنل‌هایی هستند که می‌خواهیم با یکدیگر ترکیبشان کنیم، w_l ضریب اهمیتی است که به هر کرنل انتساب داده می‌شود و S ماتریس مشابهتی است که قصد داریم به دستش بیاوریم. عبارت اول در تابع هزینه موجب می‌شود که اگر فاصله‌ی بین دو سلول i و j در کرنل‌ها زیاد باشد میزان مشابهت آن‌دو در ماتریس S یعنی S_{ij} کم شود. عبارت دوم یک رگولاریزیشن برای جلوگیری از تبدیل S به ماتریس همانی است. عبارت سوم موجب می‌شود که گراف با ماتریس مجاورت S تقریبا به صورت c خوشه باشد که یال‌های درون خوشه‌ای سنگین و یال‌های بین خوشه‌ای ضعیف باشند(هر ستون ماتریس L بردار معرف یکی از این خوشه‌ها است). عبارت آخر نیز از اینکه یک کرنل خیلی مهم و بقیه بی اهمیت شوند جلوگیری می‌کند و به مساوات اثر کرنل‌ها در تشکیل S کمک می‌کند.

تعریف ریاضی این مسئله محدب نیست اما بهینه‌کردن هریک از سه متغیر به شرط ثابت بودن دوتای دیگر یک مسئله‌ی محدب است و به همین دلیل [۷] به صورت متناوب آن را بهینه می‌کند. [۴] بعد از به‌دست آمدن S ماتریس انتقال P به‌صورت

$$P_{ij} = \frac{S_{ij} \mathbb{۱}_{\{j \in A_K(i)\}}}{\sum_l S_{il} \mathbb{۱}_{\{l \in A_K(i)\}}} \qquad (۱۴ـ۳)$$

تعریف می‌شود تا به صورت

$$H_{ij}^{(t+۱)} = \tau H_{ij}^{(t)} P + (۱ - \tau I_N) \qquad (۱۵ـ۳)$$

برای چند مرحله روی آن فرآیند واپخشی رخ دهد. این امر سبب کاهش نویز های موجود در S می‌شود. در نهایت از $H_{ij}^{(T)}$ به‌عنوان ماتریس مشابهت برای تحلیل‌هایی همچون خوشه‌بندی استفاده می‌شود.

[۲] Optimization
[۳] Single Cell Interpretation via Multi-kernel Learning
[۴] Alternating

۳ـ۵ مشابهت دارویی با شبکه‌ی توجه‌وار چنددیدگاهی خودرمزگذار گراف

الهام‌بخش ترین مقاله برای پژوهش ما [۸] بود که در معماری‌های نسخه‌ی ۲۰ به بعد از آن استفاده کردیم (البته روش‌های پیشنهادی این مقاله به صورت یادگیری بانظارت [۵] یا یادگیری نیمه‌نظارت [۶] و ترارسانی [۷] است ولی تمامی معماری‌های ما به صورت یادگیری بی‌نظارت [۸] انجام شد). در این پژوهش برای تعدادی دارو چندین دید (مثلا CPI [۹] یا TTD [۱۰]) وجود دارد که با ترکیب آن‌ها سعی می‌شود پیشبینی بهتری از DDI [۱۱] داروها به‌دست آید. نوآوری اصلی این پژوهش استفاده از ساختار توجه برای کرنل لاپلاسین لایه‌های GCN در خودرمزگذار گراف کیپف [۵] است. سابقاً در پژوهش کیپف واپخشی با لاپلاسین فقط یک گراف (گراف ارجاع‌دهی مقالات) انجام می‌شد اما در این پژوهش از

$$\hat{A} = \sum_u diag(g^u) A^u \qquad (۱۶ـ۳)$$

به عنوان کرنل واپخشی استفاده‌شد که A^u ماتریس مشابهت دید u ام است. $g \in \mathbb{R}^N$ نیز بردار خروجی ساختار توجه است (برخلاف [۷] که به هر دید یک ضریب توجه می‌دهد، این پژوهش به هر راس گراف در هردید یک ضریب توجه می‌دهد) که یک شبکه‌ی تک لایه‌ی MLP با تابع فعال‌سازی softmax و A^u ها به‌عنوان ورودی است.

$$\sum_u diag(g^u) = I \qquad (۱۷ـ۳)$$

بخش رمزگذار یک شبکه‌ی دولایه به‌صورت

$$Z^{(u)} = f(X^{(u)}, A^{(u)}; W^u_۱, W^u_۲) = Softmax(\hat{A}^u \; ReLU(\hat{A}^u X^{(u)} W^{(u)}_{\cdot} W^u_۱) \qquad (۱۸ـ۳)$$

است که $Z^{(u)}$ ها با یکدیگر الحاق می‌شوند تا امبدینگ Z تشکیل شود. بخش رمزگشا بسته به نوع یادگیری (نیمه‌نظارتی، بانظارت و یا ترارسانی) متفاوت است. مثلا می‌تواند به‌صورت

$$X' = f'(X, \hat{A}) = Sigmoid(\hat{A} Z W_۳) \qquad (۱۹ـ۳)$$

[۵] Supervised Learning
[۶] Semi-Supervised Learning
[۷] Transductive
[۸] Unsupervised Learning
[۹] Drug Chemical Protein Interactome
[۱۰] Protein and Nucleic Acid Targets
[۱۱] Drug Drug Interaction

با تابع هزینه‌ی

$$\sum ||X - X'||^{۲} \qquad (۲۰\_۳)$$

باشد و همچنین تابع ضرر بخش یادگیری بانظارت نیز به آن افزوده شود (شبکه‌ی مربوط به پیشبینی هم مثل بخش رمزگشا ورودیش Z است). در این پژوهش زمانی که سیگنال ورودی X وجود نداشت از بردار one-hot رئوس استفاده شد همچنین در مدلی دیگری از آن برچسب‌ها به عنوان سیگنال ورودی به‌کار برده شدند.

فصل ۴

روش پیشنهادی

در این پژوهش از داده‌گان [۳] استفاده شد. قسمت‌هایی از کد این مقاله(پکیج SNFtool) که به‌زبان R نوشته‌شده برای تحلیل ضروری بودند و از سوی دیگر آموزش شبکه‌های عصبی در محیط pytorch و به زبان پایتون انجام می‌شد. برای حل این مشکل از پکیج rpy۲ استفاده‌کردیم تا بتوانیم کدهای به زبان R را در محیط پایتون اجرا کنیم.

ابتدا هر دید را به صورت

$$\tilde{f} = \frac{f - E(f)}{\sqrt{Var(f)}} \qquad (۱-۴)$$

نرمالایز کردیم. سپس از روزی هریک ماتریس مشابهتی به صورت

$$A(i,j) = \exp(-\frac{\rho^{۲}(x_i, x_j)}{\mu\epsilon_{i,j}}) \qquad (۲-۴)$$

ساختیم که ρ فاصله‌ی اقلیدسی بود [۱] ، $\epsilon_{i,j}$ از طریق رابطه‌ی

$$\epsilon_{i,j} = \frac{mean(\rho(x_i, N_i)) + mean(\rho(x_j, N_j)) + \rho(x_i, x_j)}{۳} \qquad (۳-۴)$$

بدست آمد و $\rho(x_i, N_i)$ میانگین فاصله‌ی x_i از همسایگانش بود. البته قبل از این نرمالیز کردن ماتریس مشابهت را تنک کردیم: برای هر راس، در سطر مربوطش فقط K همسایه‌ی نزدیکش را درنظر گرفتیم و بقیه‌ی درایه ها را صفر کردیم. (μ و K ابرپارامتر هستند و به صورت تجربی تنظیم می شدند اما به‌طور معمول حدود آن‌ها به‌صورت $۱۰ \leqslant K \leqslant ۳۰$ و $۰/۸ \leqslant \mu \leqslant ۰/۳$ و $۲۰ \leqslant T \leqslant ۱۰$ بود.)

[۱] البته کورولیشن را هم امتحان کردیم اما نتیجه‌ي مطلوبی نداد

در ادامه نسخه های مختلف کدهایی که ایجاد کردیم را به ترتیب زمانی توضیح می‌دهیم. فایل iphyton مربوط به‌هر نسخه به همراه دیگر کدها و نتایج در این نشانی موجود است. عمدتا تفاوت هر نسخه با بقیه در ایده‌های اصلی است و تغییرات جزئی از قبیل تعداد ابعاد هرلایه از شبکه یا مقادیر ابرپارامترها در همان نسخه بررسی شده است.

۴ـ۱ توضیح کدهای مربوط به آموزش شبکه‌های عمیق

۴ـ۱ـ۱ نسخه‌ی ۱۰

برای شروع ساده‌ترین ایده یعنی خودرمزگذار ساده آزمایش شد. هر سه مجموعه داده الحقاق [۲] شدند و به عنوان ورودی به شبکه استفاده‌شدند. بعد از آموزش نمایش داده‌ها در embedding بررسی شد. [۳] برای خوشه‌بندی embedding ها هم از Kmeans و هم از Spectral Clustering استفاده شد و روی خوشه‌ها میزان P Value مربوط به Cox Partial Likelihood محاسبه شد و با نتایج [۳] مقایسه می‌شد. در طی این نسخه متوجه شدیم embedding بسیار متاثر از مجموعه داده‌ی ژنتیکی است و دلیلش هم منطقی بود، چراکه با داشتن ۱۲۰۴۲ بعد بود درمقایسه با ۱۳۰۵ و ۵۳۴ بعد تاثیر بسیار زیادی روی تابع هزینه می‌گذاشت. (تابع هزینه $(X - X')^2$ بود.) برای رفع این مشکل معماری بعدی ارائه شد.

۴ـ۱ـ۲ نسخه‌ی ۱۱

کماکان از خودرمزگذار ساده استفاده شد اما تابع هزینه به صورت

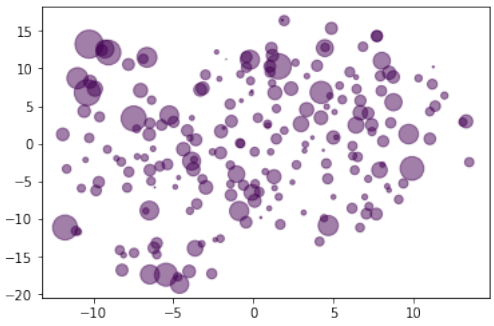
$$\frac{(X_{[gene]} - X'_{[gene]})^2}{dim(gene)} + \frac{(X_{[methy]} - X'_{[methy]})^2}{dim(methy)} + \frac{(X_{[mirna]} - X'_{[mirna]})^2}{dim(mirna)} \qquad (۴ـ۴)$$

تغییر داده‌شد تا میزان تاثیر مجموعه داده‌گان روی embedding متاثر از اندازه‌ی ابعادشان نباشد. پس از بررسی نتایج برای نسخه‌های مختلف این معماری (تغییر ابعاد لایه‌های مختلف و یا تغییر تعداد لایه‌ها) کماکان نتیجه نگرفتیم و حدسمان این بود که دلیلش بیش برازش [۴] باشد چون ابعاد داده در مقایسه

[۲] Concatenate

[۳] دراکثر نسخه‌های کد این پژوهش هم فاصله‌ی cosine و هم فاصله‌ی اقلیدسی در embedding ها بررسی شد اما عموما فاصله‌ی اقلیدسی نتیجه‌ی بهتری می‌داد.

[۴] Overfitting



شکل ۴ـ۱: نمای امبدینگ معماری نسخه‌ی ۱۲ در دوبعد با استفاده از TSNE

با تعداد نمونه‌ها خیلی زیاد بود (۱۳۸۸۱ در مقابل ۲۱۵). برای رفع این مشکل اولین ایده استفاده از لایه‌های Dropout بود.

۴ـ۱ـ۳ نسخه‌ی ۱۲

در نسخه‌ی ۱۲ به معماری خودرمزگذار لایه‌های Dropout اضافه‌شد و دوباره اندازه‌های مختلف ابعاد برای لایه‌های شبکه بررسی شد. علاوه‌بر بررسی های قبلی که روی embedding انجام می‌شد، با استفاده از PCA و TSNE تمام مریض‌ها را در embedding تصویرسازی کردیم. همانطور که از شکل ۴ـ۱ پیداست هر مریض یک دایره است و دایره‌های بزرگتر زمان برزیستی‌شان بیشتری دارند. کماکان نتایج خوبی نگرفتیم و مسئله‌ی بیش‌برازش مشهود بود.

۴ـ۱ـ۴ نسخه‌ی ۱۳

به تابع هزینه‌ی مدل قبلی رگولاریزیشن l1 روی ماتریس‌های وزن رمزگذار اضافه کردیم. توجیهمان این بود که چون ابعاد نمونه‌ها بسیار زیاد است، احتمالا همه‌ی ویژگی‌ها مهم نیستند و بتوان با تعداد کمتری از ویژگی‌ها embedding مناسبی ساخت. همانگونه که می‌دانیم رگولاریزیشن $L1$ سعی می‌کند ماتریس را تنک و درایه‌های بی تاثیر را صفر کند.

۴ـ۱ـ۵ نسخه‌ی ۱۴

برای کاهش بیشتر بیش برازش قبل از اینکه خروجی بخش رمزگذار را به رمزگشا بدهیم، یک نویز از جنس $\mathcal{N}(۰, \sigma^۲ I)$ به آن اضافه کردیم. توجه کنید این معماری خودرمزگذار متغیر نبود چرا که کماکان تابع هزینه‌ی آن

$$(X - X')^۲ + \lambda l_۱(W) \qquad (۵ـ۴)$$

و فاقد عبارت KL Divergence بود.

۴ـ۱ـ۶ نسخه‌ی ۱۵

در نسخه‌های ۱۳ و ۱۴ برای اینکه فقط تاثیر یک تغییر جدید درمقایسه با خودرمزگذار ساده بررسی شود، از dropout استفاده نشد. برای همین در این نسخه به صورت همزمان از dropout ، رگولاریزیشن l۱ و نویز گاوسی روی embedding بهره بردیم.

۴ـ۱ـ۷ نسخه‌ی ۱۶

به معماری نسخه‌ی قبلی تابع هزینه‌ی مربوط به Kl Divergence را اضافه کردیم و شبکه تبدیل به یک خودرمزگذار متغیر شد که ماتریس کواریانس مربوط به توزیع نمونه‌گیری در embedding به‌صورت $\mathcal{N}(۰, \sigma^۲ I)$ بود که σ یک عدد ثابت است.

۴ـ۱ـ۸ نسخه‌ی ۱۷

در این نسخه معماری خودرمزگذار نویززدا [۵] بررسی شد. تفاوت این خودرمزگذر با خودمزگذار ساده، افزودن نویز $\mathcal{N}(۰, \sigma^۲ I)$ به داده‌ی ورودی و تلاش شبکه برای بازسازی داده‌ی اولیه بدون نویز است.

۴ـ۱ـ۹ نسخه‌ی ۱۸

به خودرمزگذار متغیر نسخه‌ی ۱۶ dropout و رگولاریزیشن l۱ اضافه‌شد.

[۵] Denoising Autoencoder

۴ـ۱ـ۱۰ نسخه‌ی ۲۰ تا ۲۳

از این نسخه به بعد ایده‌ی خودرمزگذار کلاسیک (که فقط روی فضای ویژگی عمل می‌کند) کنار گذاشته‌شد و ایده‌ای مشابه [۸] پیاده‌سازی شد. ساختار توجهی [۶] اضافه‌کردیم که به هر مریض ۳ عدد نسبت دهد که مجموعشان ۱ و همه بزرگتر از ۰ باشند.(خروجی لایه‌ی softmax هستند و هرکدام از آن‌ها بیانگر اهمیت آن مریض در هریک از ۳ مجموعه‌داده است.) سپس ضرایب مربوط به هردید را به صورت ماتریس قطری درآوردیم تا ماتریس‌های لاپلاسین را باتوجه به آن‌ها جمع بزنیم.

$$\hat{L} = g_1 L^{(1)} g_1 + g_2 L^{(2)} g_2 + g_3 L^{(3)} g_3 \qquad (۴ـ۶)$$

این ساختار توجه به‌جای اینکه به هر دید یک ضریب اهمیت بدهد، برای هرفرد در هر دید یک ضریب در نظر می‌گرفت. علت اینکه برخلاف [۸] g_i از دو طرف در $L^{(i)}$ ضرب می‌شود، این است که ماتریس لاپلاسین باید متقارن و مثبت نیمه‌معین باشد تا بتوانیم خوشه‌بندی طیفی را روی آن اجرا کنیم. سپس از $\hat{L}$ به عنوان کرنل واپخشی در لایه‌های GCN بخش رمزگذار استفاده می‌شد. تابع فعال‌سازی لایه‌ی آخر بخش رمزگذار نیز softmax بود تا هربعد embedding متناظر با یکی از خوشه‌ها باشد(طبق [۳] داده‌ی gbm ۳ خوشه‌ای بود) و اندازه‌ی هر نمونه بیمار در آن بعد بیانگر احتمال تعلقش به خوشه‌ی متناظرش شود. این خاصیت تابع softmax که سعی می‌کند یک بعد را نزدیک به ۱ و بقیه‌را تقریبا صفر کند آن را برای هدف مذکور مناسب‌تر می‌کرد. همچنین لازم به‌ذکر است که این معماری کاملا بدون استفاده‌ی مستقیم از ویژگی‌ها آموزش دید و بردار one-hot رئوس به عنوان ورودی سیگنال به شبکه داده‌شد. بخش رمزگشا دارای یک لایه با تابع فعال‌سازی سیگموید بود تا همان بردارهای one-hot بازسازی شوند. (تابع هزینه به صورت $(\tilde{X} - \tilde{X}')^2$ تعریف شد)در این معماری و نسخه‌های بعدی هم قسمت embedding و هم قسمت $\hat{L}$ بررسی شد(روش‌های مختلف خوشه‌بندی مثل خوشه‌بندی طیفی). در معماری نسخه‌ی ۲۰ embedding کم‌بعد بود و حدس زده‌شد که ۳بعد برای رمز کردن این‌همه اطلاعات کافی نیست لذا ابعاد آن را از ۳ به ۱۰ افزایش دادیم. در نسخه‌ی ۲۲ این عدد به ۴۰ افزایش یافت و در نسخه‌ی ۲۳ بعد embedding ۱۳۰ شد.

[۶] Attention

۴ـ۱ـ۱۱ نسخه‌ی ۲۴ تا۲۷

تابع هزینه به فاصله‌ی l۱ تغییر یافت ($|\tilde{X} - \tilde{X}'|$) و برای embedding های به‌طول ۳و۱۰و۴۰و۱۳۰ آزمایش شد.

۴ـ۱ـ۱۲ نسخه‌ی ۲۸ تا ۳۰

تابع هزینه‌ی Log Sigmoid به جای فاصله‌ی نرم ۱ برای چندین اندازه‌ی embedding (۳،۱۰،۴۰) بررسی شد.

۴ـ۱ـ۱۳ نسخه‌ی ۳۱ تا ۳۴

همانطور که پیشتر گفته‌شد از معماری ۲۰ به بعد طبق [۸] تابع فعال‌سازی بخش رمزگذار soft-max و بخش رمزگشا sigmoid بود. اما تجربه‌مان نشان داد که softmax آموزش شبکه را دشوار می کند. ازاین‌رو تابع فعال‌سازی رمزگذار را به relu تغییر دادیم و برای embedding های به اندازه‌ی ۳،۱۰،۴۰،۱۳۰ شبکه‌را بررسی کردیم.

۴ـ۱ـ۱۴ نسخه‌ی ۳۵

این‌بار تابع فعال‌سازی بخش رمزگشا را به Softmax و تابع هزینه‌را به Cross Entropy تغییر دادیم.

۴ـ۱ـ۱۵ نسخه‌ی ۳۶ تا ۳۹

معماری بخش توجه را به صورتی تغییر دادیم که به جای اینکه ماتریس مجاورت ها را به عنوان ورودی بگیرد، ماتریس X که برارهای onehot رئوس گراف بود را دریافت کند. سپس شبکه‌را برای em-bedding های به اندازه‌های ۱۰،۴۰،۷۰،۱۵۰ بررسی کردیم.

۴ـ۱ـ۱۶ نسخه‌ی ۴۰

تابع فعال‌سازی رمزگذار را relu و رمزگشا را softmax کردیم. همچنین تابع هزینه را به فاصله‌ی نرم ۱ تغییر دادیم.

۴ـ۱ـ۱۷ نسخه‌ی ۴۱

در این نسخه به جای اینکه بردار one-hot رئوس را به‌عنون سیگنال ورودی به شبکه دهیم، یکی از سه داده را (مثلا mirna) به شبکه می دادیم و برخلاف قبل که ماتریس $\hat{L}$ حاصل جمع وزن دار سه ماتریس بود، آن را حاصل جمع توجه‌وار دو ماتریس وزن‌دار دیگر کردیم (methyوgene) .

۴ـ۱ـ۱۸ نسخه‌ی ۴۲

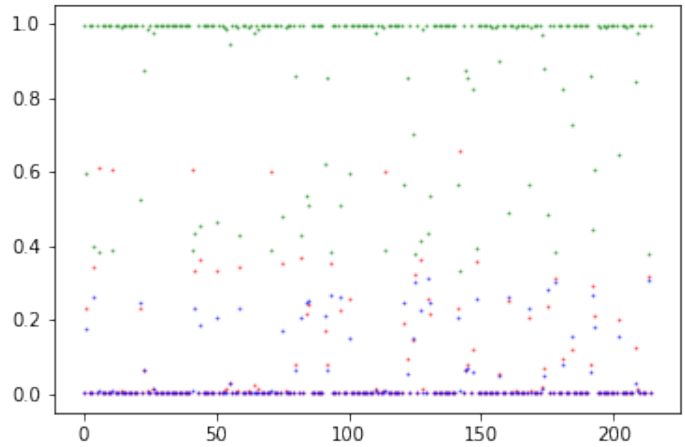
همان معماری قبلی را به‌کار بردیم اما به شبکه‌ی توجه مجموعه‌داده‌ی سیگنال (mirna) را به جای L_i ها ورودی دادیم.

۴ـ۱ـ۱۹ نسخه‌ی ۴۳ و ۴۴

دوباره تابع فعال‌سازی بخش رمزگذار را به softmax تغییر دادیم.

۴ـ۱ـ۲۰ نسخه‌ی ۴۵ تا ۴۹

از آنجا که قبلا تمام خودرمزگذارهایی که پیاده‌سازی کرده‌بودیم(نسخه‌ی۱ تا ۲۰) تابع هزینه‌شان فاصله‌ی نرم۱ یا نرم۲ داده با نسخه‌ی بازسازی‌اش بود، این بار تصمیم گرفتیم مطابق ایده‌ی [۶] از تابع هزینه‌ی logloss استفاده‌کنیم. معماری به‌صورت کلی یک خودرمزگذار متغیر روی خودویژگی‌ها(الحاق سه مجموعه داد) با لایه‌های dropout بود که تابع هزینه‌ی logloss داشت. همچنین مطابق [۶] با Like-Cox Partial lihood تک متغیره میزان اهمیت تک تک ابعاد embedding را بررسی کردیم.



شکل ۴ـ۲: نمودار وزن‌های ساختار توجه. قرمز مربوط به gene سبز مربوط به methy و آبی مربوط به mirna است. بعد افق شماره‌ی راس گراف هارا نشان می‌دهد و بعد عمود بزرگی عدد احتمالی. برای مثال نقطه‌ی آبی رنگ با مختصات (۴۳, ۰/۷) به این معنا است که $g^{mirna}(۲۵) = ۰/۷$ یعنی ضریب توجه برای راس ۴۳ ام در دید mirna ۰/۷ است و لذا در نتیجه مجموع دو دید methy و gene برای این راس ۰/۳ است. همانطور که از نمودار پیداست، اثر مذکور رخ داده و دید methy غلبه کرده‌است.

۴ـ۱ـ۲۱ نسخه‌ی ۵۰

در معماری‌های ۲۰ تا ۴۹ که از ساختار توجه برای ترکیب خطی ماتریس‌های $L^{(i)}$ استفاده کردیم. نمودارهای g_{mirna} و g_{methy} و g_{gene} را رسم می‌کردیم ۴ـ۲ و در اکثر مواقع g مربوط به یکی از داده‌ها بزرگ و غالب می‌شد و دو مجموعه‌داده‌ی دیگر عملا بی‌اثر می‌شدند. برای جلوگیری از این مسئله ایده‌ای مشابه [۷] را پیاده سازی کردیم.

$$\sum_{i=۱}^{i=۳} mean(g_i) \times log(mean(g_i)) \qquad (۴ـ۷)$$

عبارت ۴ـ۷ را به تابع هزینه افزودیم تا از غلبه‌ی یک $L^{(i)}$ خاص روی $\tilde{L}$ جلوگیری شود.

۴ـ۱ـ۲۲ نسخه‌ی ۵۱

از این نسخه به بعد از پکیج rpy۲ استفاده کردیم تا بخش‌هایی از کد که در محیط R باید اجرا می‌شدند در همان محیط پایتون در فضای colab قابل اجرا باشند. پیشتر بعد هربار آموزش شبکه، ماتریس embedding و $\hat{L}$ به صورت فایل از محیط پایتون استخراج می‌شدند و به منظور تحلیل در محیط R دوباره بارگذاری می‌شدند که فرآیند زمان‌بری بود. این پکیج کمک کرد که کدهای R در فضای پایتون قابل اجرا باشند و همچنین این امکان را فراهم کرد که حتی در حین آموزش شبکه نیز تمام آنالیزها روی embedding و $\tilde{L}$ به صورت لحظه به لحظه قابل نمایش باشد.
معماری را نیز تغییر دادیم و با اقتباس از [۷] عبارت

$$\lambda\, trace(L^\top \tilde{A} L) \qquad (۸ـ۴)$$

را به تابع هزینه اضافه کردیم. L خروجی یک لایه MLP است و ابعادش به صورت $N \times c$ است که c تعداد خوشه‌های طیفی گراف با ماتریس لاپلاسین $\hat{L}$ است. درواقع هر ستون این ماتریس، بردار معرف یکی از خوشه‌ها است. از آنجاکه این بردار معرف‌ها باید متعامد یکه [۷] باشند، عبارت

$$\mu(L^\top L - I) \qquad (۹ـ۴)$$

را هم به تابع هزینه افزودیم.

۴ـ۱ـ۲۳ نسخه‌ی ۵۲

به معماری نسخه‌ی قبلی رگولازیزشن l۲ روی وزن‌های بخش رمزگذار افزودیم و همچنین با اقتباس از [۹] روی ماتریس $\hat{L}$ رگولاریزیشن l۱ گذاشتیم. طبقی توضیحی که در آن مقاله داده‌شده بود، تنک کردن ماتریس لاپلاسین موجب مشخص‌تر شدن خوشه‌ها می‌شود. یعنی یال‌های درون خوشه‌ای تقویت و یال‌های بین خوشه‌ای تضعیف می‌شوند.

۴ـ۱ـ۲۴ نسخه‌ی ۱ تا ۹

نسخه‌های ۱ تا ۹ ابتدایی ترین مدل‌ها بودند و کدشان به صورت مرتب آرشیو نشد. ولی ایده‌ی کلی بدین صورت بود که دقیقا معماری کیپف [۵] را پیاده‌سازی کردیم. بدین صورت که مثلا داده‌ی mirna

[۷] Orthonormal

و methy را که ابعداشان کمتر بود الحاق کردیم و به عنوان سیگنال ورودی به شبکه دادیم (به‌جای بردارهای وجود کلمات) و از L_{gene} در لایه‌های GCN استفاده شد (به جای ماتریس ارجاع‌دهی مقالات). علت ناموفق بودن این نسخه‌ها عدم ترکیب کافی مجموعه‌دادگان و تفاوت ماهیتی مسئله‌ی ما با مسئله‌ی کیپف بود:

۱. در معماری کیپف هدف ترکیب چندین مجموعه‌داده نبود بلکه می‌خواستیم یک ماتریس مجاورت ناقص را تکمیل کنیم. برای همین وقتی این معماری را روی مجموعه‌داده‌ی خودمان بکار بردیم درنهایت همان ماتریس مجاورت gene بازسازی شد. البته خود embedding را هم مطالعه کردیم ولی اطلاعات مفید و قابل اعتمادی نداشت.

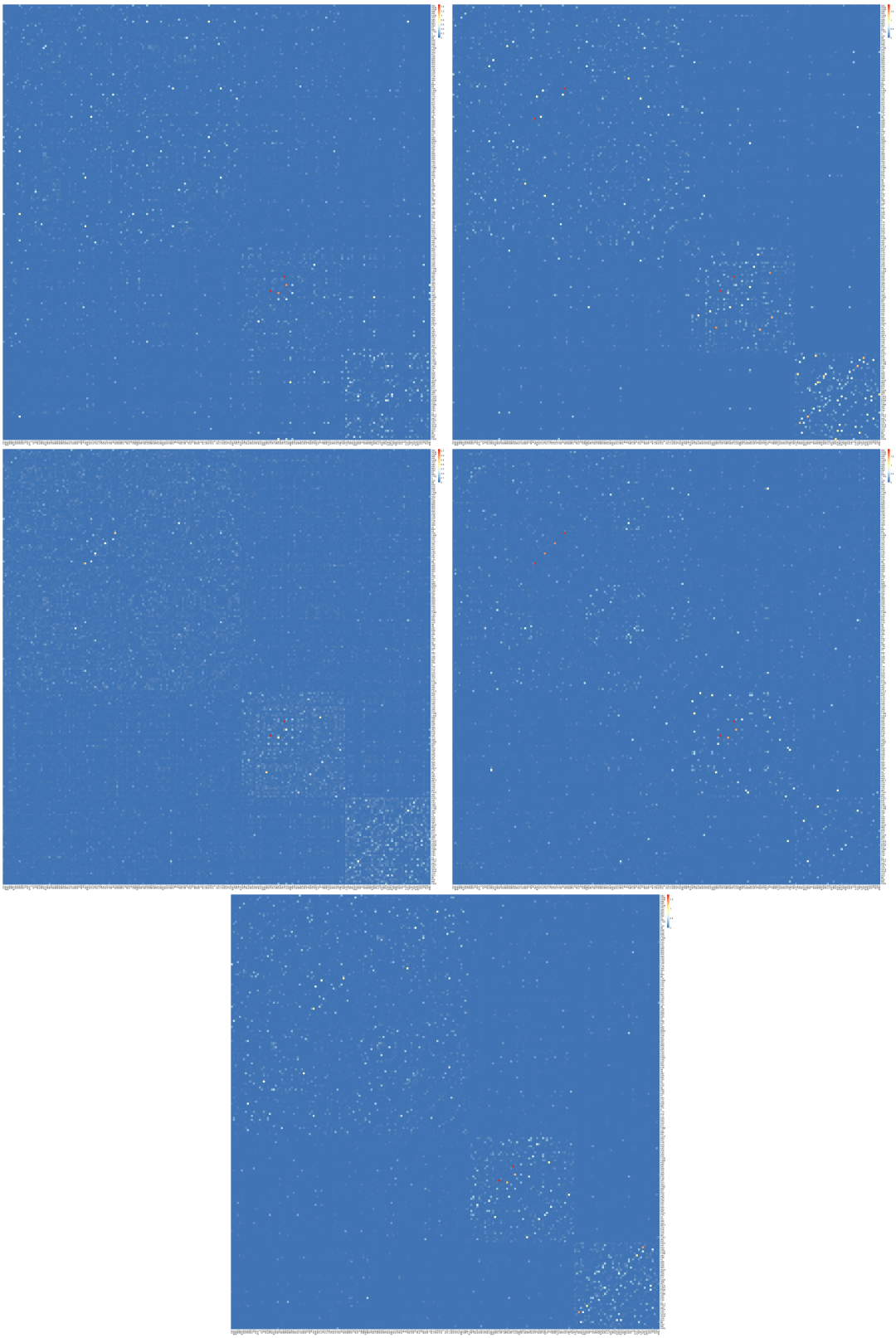
۲. در معماری کیپف ماتریس مجاورت تمامی یال‌هایش درست بود (یال غلط نداشت) و مشکلش صرفا نداشتن تعدادی یال درست بود. اما در مسئله‌ی ما هرسه ماتریس‌های مجاورت هم دارای یال غلط و هم فاقد یال درست بودند.

۴ـ۲ کد مربوط به آنالیزهای بعد آموزش شبکه

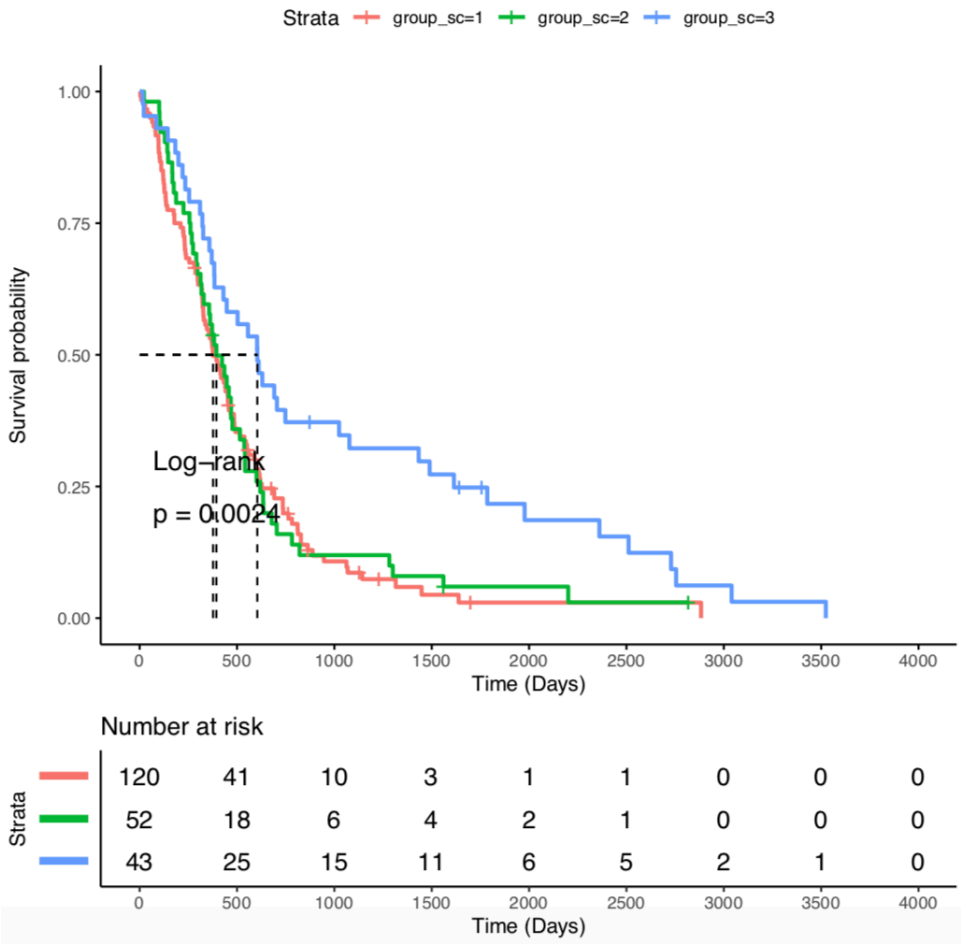
این کد نیز در آدرس اینترنتی که ارائه شد موجود است و به زبان R نوشته شده است. (تا قبل از اینکه پکیج $rp2y$ استفاده‌شود انواع آنالیزها در این محیط انجام می‌شد.) در ادامه قسمت‌هایی از آنالیزهای این کد روی معماری نسخه ۱۵ آورده شده‌است ($K = ۳۰ \ \& \ \mu = ۰/۳ \ \& \ T = ۱۷$) .

Gene	Methy	Mirna	Cross Diffusion	Embedding
0.00678	0.99879	0.26649	0.00680	0.00142

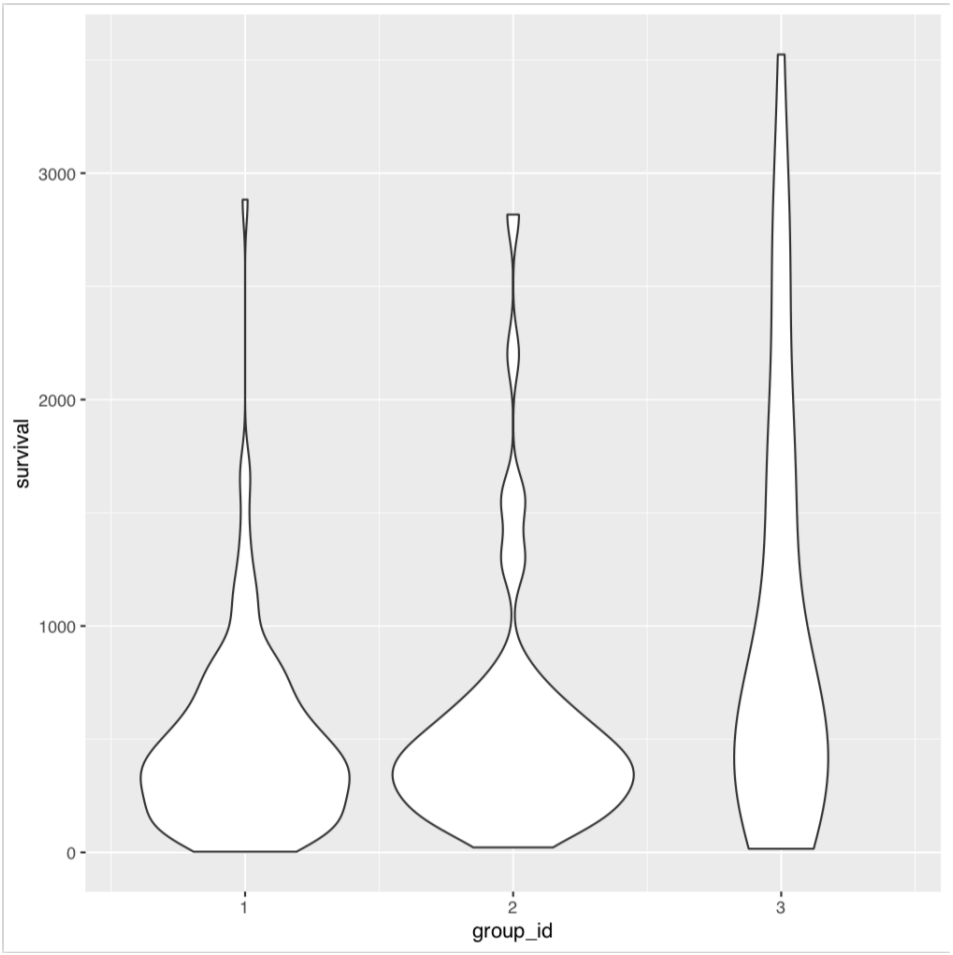
جدول ۴ـ۱: مقدار P Value برای Cox Partial Likelihood خوشه‌بندی روی دیدهای مختلف.



شکل ۴ـ۳: تصاویر حاصل از اجرای خوشه‌بندی طیفی روی دیدهای مختلف. با ترتیب سطری از راست به چپ و از بالاترین سطر به پایین‌ترین: gene ، methy ، mirna ، ترکیب سه دید با snftool و embedding حاصل از آموزش شبکه‌ی ۱۵.



شکل ۴ـ۴: نمودار Kaplan-Meier برای خوشه‌بندی طیفی embedding حاصل از معماری ۱۵



شکل ۴ـ۵: نمودار violin برای خوشه‌بندی طیفی embedding حاصل از معماری ۱۵

فصل ۵

نتیجه‌گیری

در این نوشتار مجموعه‌ای از تلاش‌ها و ایده‌های مبتنی بر واپخشی و خودرمزگذار گراف برای ترکیب چندین مجموعه داده‌ی ژنتیکی ارائه کردیم تا خوشه‌بندی مناسبی از جهت برزیستی ایجاد کنیم. متاسفانه نتوانستیم به روش جدید و صحیحی دست یابیم که با اطمینان آن را ارائه دهیم. اما به هرحال نگارش این تلاش‌ها ممکن است برای پژوهش‌های آتی در این زمینه ایده‌بخش و مفید باشد. در طی این پژوهش متوجه شدیم آموزش شبکه‌های دارای لایه‌ی GCN فرآیند آموزش خودرمزگذارها را دشوارتر از آن‌چه که هست می‌کند و تعداد ایپوک های بسیار بیشتری طول خواهد کشید تا به دقت مطلوب برسد، مخصوصا اگر از بیشتر از ۲ــ۳ لایه متوالی استفاده شود. مشکل بعدی ناشی از تفاوت هدف شبکه‌ها با تحلیل ما بود. عمده‌ی شبکه‌هایی که طراحی شدند به دنبال ترکیب دیدها و ارائه‌ی embedding بودند. این درحالی است که هدف نهایی ما خوشه‌بندی بود و ما به دنبال معماری بودیم که اگر هرچندبار از اول آن را روی مجموعه‌دادگان آموزش دهیم به خوشه‌بندی تقریبا یکسان و با P Value بهتری از خط مبنا [۱] ۰/۰۰۰۲ دست یابد. اما همواره در روش‌های ما خوشه‌بندی یک آنالیز پسا آموزشی بود و روی تابع هزینه‌و نحوه‌ی یادگیری شبکه تاثیر نداشت (بجز نسخه‌های اخیر مثل ۵۰ و ۵۱ که از [۷] ایده‌برداری کردند). اما مقایسه‌ی embedding خودرمزگذارها کار چالش برانگیزی است چراکه با هربار آموزش دادن آن‌ها روی یک مجموعه‌دادگان ثابت یک embedding کاملا جدید به‌دست می‌آید و نباید انتظار داشت که خوشه‌های آن ثابت بماند. به همین دلیل توصیه می‌شود که پژوهش‌های آتی روی این مسئله به‌دنبال شبکه‌های مبتنی بر خوشه‌بندی باشند و مفهوم خوشه‌بندی را به صورت مستقیم در معماری و

[۱] Baseline

تابع هزینه وارد کنند تا در طی آموزش embedding مناسبی برای این منظور ایجاد شود. مثلا [۱۰] یک معماری از این نوع است و تبدیل آن به یک شبکه‌ی چنددیدگاهی [۲] می‌تواند یک مسیر پژوهشی برای آینده باشد. یکی از پژوهش‌هایی که این ایده را پیش گرفت، [۱۱] بود اما نواقصی دارد:

۱. در رابطه‌ی (۳) از آن مقاله که تابع هزینه تعریف شده امکان دارد ضرایب مربوط به چندتا از دیدها نزدیک صفر و تعدادی دیگر بسیار بزرگ شود و عملا ترکیب دیدها به مساوات صورت نگیرد لذا افزودن عباراتی همچون

$$\lambda \sum_{v} a^{(v)} \log a^{(v)}$$

از این مشکل جلوگیری می‌کند

۲. تابع هزینه‌ی این پژوهش به دنبال یافتن $Y \in \mathbb{R}^{N \times c}$ ای است که بیانگر بهترین خوشه‌بندی در مجموع برای تمام $L^{(v)}$ ها باشد. لذا ایده‌ی ترکیب کردن دیدها در آن به صورت توافق همگانی است و نه تکمیل کردن یکدیگر. تلاش برای ترکیب دادگان به وجهی که یکدیگر را تکمیل کنند می‌تواند موضوع پژوهش آتی باشد.

[۲] multi-view

مراجع

[1] M. Soleymani. Variational auto-encoder (vae) [slideset]. 2019.

[2] M. Soleymani. Generative models [slideset]. 2019.

[3] A. G. Bo Wang, Aziz M Mezlini. Similarity network fusion for aggregating data types on a genomic scale.

[4] Z. T. Bo Wang, Jiayan Jiang. Unsupervised metric fusion by cross diffusion. 2012.

[5] M. W. Thomas N. Kipf. Variational graph auto-encoders. 2016.

[6] L. X. G. Kumardeep Chaudhary, Olivier B. Deep learning based multi-omics integration robustly predicts survival in liver cancer. 2017.

[7] E. P. S. B. Bo Wang, Junjie Zhu. Visualization and analysis of single-cell rna-seq data by kernel-based similarity learning. 2017.

[8] F. W. Tengfei Ma, Cao Xiao. Drug similarity integration through attentive multi-view graph auto-encoders. 2018.

[9] S. Y. Canyi Lu and Z. Lin. Convex sparse spectral clustering: Single-view to multi-view. 2018.

[10] H. L. Uri Shaham, Kelly Stanton. Spectralnet: Spectral clustering using deep neural networks. 2018.

[11] J. L. Zhenyu Huang, Joey Tianyi Zhou. Multi-view spectral clustering network. 2019.

واژه‌نامه

الف

الحاق ..... Concatenate

ایپاک ..... Epoch

ب

بدون‌سرپرستی ..... Unsupervised

برزیستی ..... Survival

بیش برازش ..... Overfitting

ت

توافق همگانی ..... Consensus

چ

چنددیدگاهی ..... Multi-view

خ

خودرمزگذار ..... Autoencoder

خودرمزگذار متغیر ..... Variational Autoencoder

خودرمزگذار نویززدا ..... Denoising Autoencoder

خوشه‌بندی طیفی ..... Spectral Clustering

خط مبنا ..... Baseline

د

دید ..... View

ر

رای‌گیری ..... Voting

رمزگذار ..... Encoder

رمزگشا ..... Decoder

ط

طیفی ..... Spectral

م

ماتریس شباهت ..... Similarity Matrix

متعامد یکه ..... Orthonormal

ن

نرمال چندبعدی ..... Multivariate Normal

هـ

همواری ..... Smoothness

و

واپخش ..... Diffusion

ولگشت ..... Random Walk