

# شبکه‌های عصبی و یادگیری عمیق دکتر صفابخش



**دانشگاه صنعتی امیرکبیر**  
( پلی تکنیک تهران )  
دانشکده مهندسی کامپیوتر

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

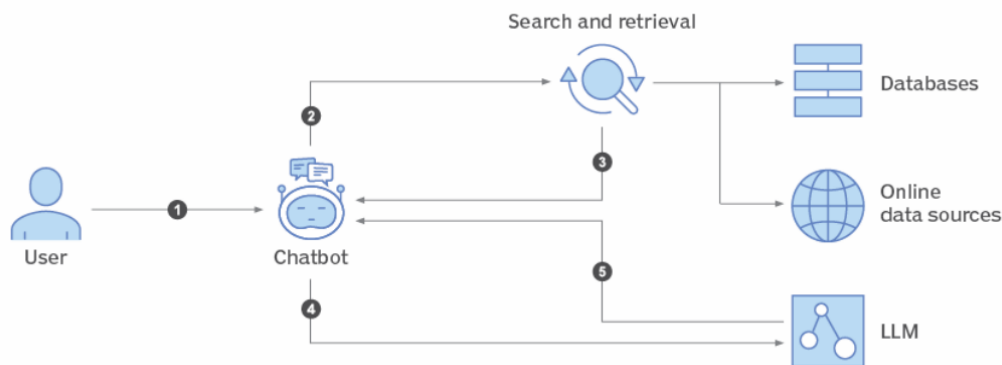
تمرین سوم  
شبکه خودسازمانده (SOM)

۴ اردیبهشت ۱۴۰۳

## سوال اول - عملی نظری

برای آموزش مدل‌های زبانی بزرگ (Large Language Model) که حاوی میلیون‌ها و میلیارد‌ها پارامتر هستند، از حجم قابل توجهی داده استفاده می‌شود. اما در تمامی این مدل‌ها یک تاریخ قطع آموزش وجود دارد که مدل زبانی هیچ اطلاعاتی در خصوص داده‌های تولید شده پس از این زمان ندارد. به عنوان مثال، تاریخ قطع آموزش مدل GPT-3.5-turbo-instruction سپتامبر ۲۰۲۱ است و از همین رو این مدل ممکن است به سوالات مربوط به رویدادهای سال ۲۰۲۲، ۲۰۲۳ و ۲۰۲۴ پاسخ صحیح ندهد. چنین داده‌هایی که بعد از تاریخ قطع آموزش تولید شده‌اند و یا بخشی از داده‌ی آموزشی اولیه‌ی مدل زبانی نیستند را داده‌ی خارجی می‌گوییم. تکنیک تولید تقویت شده با بازیابی (RAG) رویکردی است که با استخراج داده‌ی خارجی متناسب با فرمان، دریافت شده و افزودن آن به عنوان ورودی به مدل زبانی تلاش می‌کند که فرمان ورودی را تقویت کرده و به مدل زبانی کمک می‌کند تا جواب مرتبط و متناسبی بسازد. به عنوان مثال در پاسخ به یک فرمان متنی مانند «چه کسی شرکت توییتر را در سال ۲۰۲۲ خرید؟» تمامی داده‌های خارجی متناسب با این فرمان را استخراج می‌کند و آن‌ها را به عنوان ورودی به مدل زبانی GPT-3.5-turbo-instruct می‌دهد تا مدل زبانی بتواند با دانش دریافت شده پاسخ متناسبی تولید کند. این رویکرد نیاز به آموزش مجدد و با بازتنظیم (Fine tune) مدل زبانی را برطرف می‌سازد. در این پروژه می‌خواهیم با استفاده از شبکه‌های خودسازمان‌ده این تکنیک را پیاده‌سازی کنیم.

## How an LLM using RAG works



شکل ۱: فرآیند کلی RAG در یک مدل زبانی بزرگ

وظیفه اصلی RAG جست‌وجو معنایی (Semantic search) در پایگاه داده‌های اطلاعاتی و بازیابی اطلاعات خارجی دارای تناسب محتوایی با فرمان داده‌شده به یک مدل زبانی است. برای تسهیل جست‌وجوی معنایی، ابتدا داده‌های خارجی استخراج شده به بازنمایی‌های عددی یا برداری تبدیل می‌شوند که به این بازنمایی، تعبیه‌ی متن (Text embedding) می‌گوییم. در زمان بازیابی نیز ابتدا فرمان متنی به بازنمایی برداری تبدیل می‌شود و سپس نزدیک‌ترین بردارهای داده‌ی خارجی متناسب با آن استخراج می‌شود. شکل «۱» دیگرام کلی این فرآیند را نشان می‌دهد. چالش اصلی این رویکرد این است که جست‌وجوی معنایی ذکر شده به دلیل نیازمندی به محاسبه‌ی فاصله‌ی بردار فرمان با حجم عظیمی از بردارهای داده‌ی خارجی، به منابع پردازشی و

محاسباتی زیاد و زمان قابل توجهی نیاز دارد. بنابر این پیدا کردن رویکردی که جست‌وجوی معنایی را به‌صورت کارا انجام دهد بسیار حائز اهمیت است. برای افزایش کارایی جست‌وجو معنایی، یک رویکرد رایج این است که بردارهای داده‌های خارجی را خوشه‌بندی کنیم و در زمان جست‌وجو نیز ابتدا خوشه مشابه با بردار فرمان ورودی را پیدا می‌کنیم و سپس شباهت بردارهای داده‌های خارجی متعلق به آن خوشه با بردار فرمان را محاسبه می‌کنیم و اگر شباهت بردارها از یک آستانه بیشتر باشد، آنها را به‌عنوان اطلاعات مرتبط در نظر می‌گیریم.

۱. در این پروژه قصد داریم برای خوشه‌بندی داده‌های خارجی از شبکه خودسازمان‌ده استفاده کنیم. بررسی کنید که در این شبکه‌ها نسبت به سایر روش‌های خوشه‌بندی که در یادگیری ماشین به‌کار گرفته می‌شود، چه مزایا و معایبی دارد؟ به نظر شما، چرا استفاده از شبکه خودسازمان‌ده به صورت با نظارت صورت نمی‌گیرد؟ فرآیند یادگیری این مدل‌ها را توضیح دهید.

### پاسخ

قبل از بررسی مزایا و معایب شبکه SOM نیاز است که یک سری پیش‌نیازها را توضیح دهیم. پیش از هر چیزی ابتدا می‌بایست انواع الگوریتم‌های یادگیری ماشین و دلیل استفاده از آنها را توضیح دهیم. الگوریتم‌های یادگیری ماشین به ۳ دسته مختلف تقسیم می‌شوند:

(آ) یادگیری با نظارت (Supervised Learning)

(ب) یادگیری نیمه نظارتی (Semi-supervised Learning)

(ج) یادگیری بدون نظارت (Unsupervised Learning)

در یادگیری با نظارت، داده و لیبل‌های متناظر با آنها را داریم. در یادگیری نیمه نظارتی، صرفاً بخشی از داده‌ها لیبل دارند و لیبل بقیه داده‌ها مشخص نیست. دسته آخر که مورد بحث ماست، یادگیری بدون نظارت است که داده‌های موجود، لیبل ندارند و به ازای داده‌های مختلف، خروجی مناسب را نمی‌دانیم و از الگوهای پنهان در داده‌ها اطلاعی نداریم. در این صورت است که به سمت الگوریتم‌های بدون نظارت می‌آیم تا به الگوریتم این اجازه را بدهیم که هرچه را می‌تواند یاد بگیرد و اطلاعات پنهان در داده‌ها را مشخص کند. الگوریتم‌های خوشه‌بندی در این دسته قرار می‌گیرند و دلیل قرارگیری در این دسته آن است که ما هیچ اطلاعاتی در مورد داده‌های ورودی نداریم و به دنبال ایجاد وابستگی میان آنها هستیم. الگوریتم‌های خوشه‌بندی این امکان را برای ما فراهم می‌سازد تا داده‌های شبیه به هم را در یک دسته قرار دهد. در این باره در صفحه ۱۴۱ [۱] گفته شده است:

« تکنیک‌های خوشه‌بندی زمانی اعمال می‌شوند که کلاسی برای پیش‌بینی وجود نداشته باشد، بلکه زمانی که نمونه‌ها باید به گروه‌های طبیعی تقسیم شوند، اعمال می‌شوند. »

پس اگر با داده‌هایی مواجه بودیم که اطلاعاتی در مورد آنها نمی‌دانیم، خوشه‌یابی بهترین روش برای درک وابستگی‌ها میان داده‌هاست. الگوریتم‌های خوشه‌یابی را می‌توان به‌صورت زیر دسته‌بندی کرد:

- (a) Density-based
- (b) Distribution-based
- (c) Centroid-based
- (d) Hierarchical-based

در الگوریتم‌های خوشه‌یابی مبتنی بر چگالی، داده‌ها بر اساس تراکم و غلظت داده‌ها در نقاط مختلف تقسیم‌بندی می‌شود.

## پاسخ

در خوشه‌یابی توزیع شده، اساس خوشه‌یابی به صورت احتمالی است. یعنی برای تمام نقاط یک احتمال تعلق به یک خوشه خاص در نظر گرفته می‌شود که با دور شدن داده از مرکز آن خوشه، احتمال تعلق داده به خوشه مربوطه کاهش پیدا می‌کند.

پرباربردترین و سریع‌ترین نوع خوشه‌یابی، خوشه‌یابی Centroid است. این الگوریتم نقطه‌ها را بر اساس چندین مرکز در داده‌ها جدا می‌کند و هر نقطه بر اساس مجذور فاصله‌اش تا مرکز داده به یک خوشه اختصاص می‌یابد. استفاده از خوشه‌بندی سلسله‌مراتبی محدود تر از سایر روش‌هاست. بدین صورت است که برای داده‌هایی که ذاتاً به صورت سلسله‌مراتبی هستند استفاده می‌شود. مانند داده‌های مربوط به یک پایگاه داده. الگوریتم‌های مختلفی برای خوشه‌یابی وجود دارد که می‌توان چندتا از آنها را به صورت زیر نام برد:

- (a) SOM
- (b) K-means
- (c) DBSCAN
- (d) Gaussian Mixture
- (e) BIRCH
- (f) Affinity Propagation
- (g) Mean-Shift
- (h) OPTICS

در این سوال به بررسی دو مورد از مهم‌ترین الگوریتم‌ها یعنی SOM و K-means می‌پردازیم.

(آ) **K-Means**: الگوریتم K-Means یک الگوریتم بدون نظارت، مبتنی بر مرکز و تکراری (iterative) است که داده‌های ورودی را دریافت می‌کند و آنها را به K دسته تقسیم می‌کند. مقدار K می‌بایست از قبل مشخص باشد. هدف در الگوریتم K-Means به حداقل رساندن مجموع فواصل بین دو نقطه داده شده و خوشه مربوط به آنهاست و تا زمانی که مینیمم فاصله را پیدا نکند، الگوریتم متوقف نمی‌شود. ذکر این نکته الزامی است که در این الگوریتم، آموزشی‌ای صورت نمی‌گیرد و صرفاً یک کار تکراری چندین بار تکرار می‌شود تا زمانی که بهینه‌ترین حالت پیدا شود. شکل «۲» نحوه عملکرد الگوریتم K-Means را نشان می‌دهد.

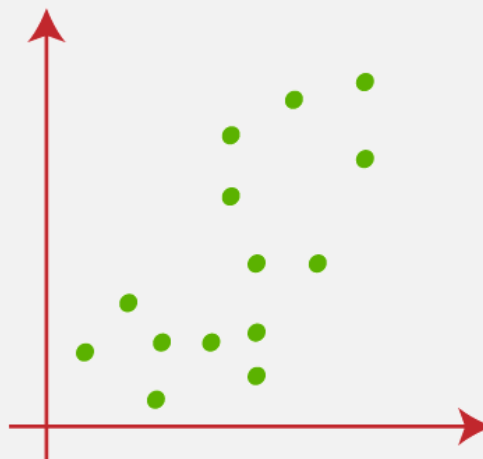


شکل ۲: ساختار الگوریتم K-Means

## پاسخ

مراحل انجام الگوریتم K-Means به صورت زیر است:

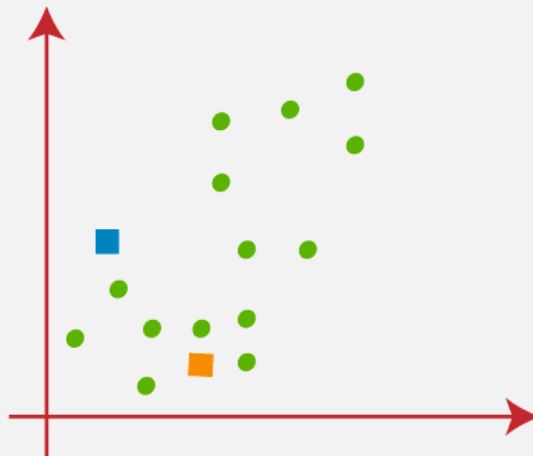
- (آ) مرحله ۱: انتخاب مقدار  $K$  بر اساس تعداد خوشه‌ها. اگر تعداد خوشه‌ها را نمی‌دانیم، عددی بزرگ را انتخاب می‌کنیم.
  - (ب) مرحله ۲: انتخاب  $K$  نقطه به صورت رندم و تصادفی.
  - (ج) مرحله ۳: قرار دادن هر نقطه در نزدیک‌ترین مرکز آن. (مرکز  $K$  خوشه‌ای که از قبل تعیین شده است).
  - (د) مرحله ۴: واریانس را حساب کرده و مرکز جدید را برحسب واریانس انتخاب کرده
  - (ه) مرحله ۵: تکرار مرحله ۳. یعنی قرار دادن هر نقطه در مرکز جدید تعیین شده
  - (و) مرحله ۶: اگر هر تخصیص مجددی رخ داد به مرحله ۴ باید برویم در غیر این صورت به مرحله ۷
  - (ز) مرحله ۷: پایان الگوریتم
- برای درک بهتر، در ادامه با رسم شکل مراحل بالا را توضیح خواهیم داد. فرض شود که داده‌های ورودی ما به صورت زیر باشد:



شکل ۳: داده‌های ورودی

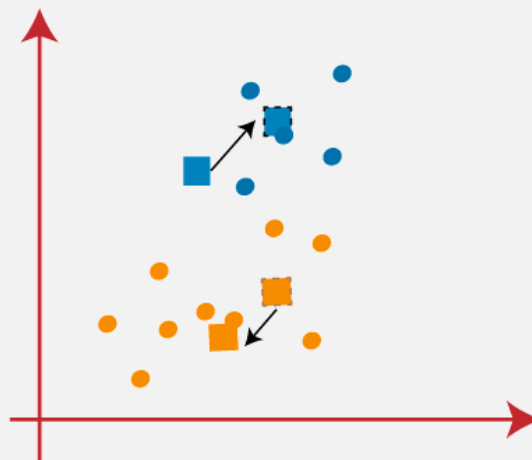
در اینجا چون تعداد خوشه‌ها برای ما مشخص است، مقدار  $K$  را ۲ فرض می‌کنیم. و دو نقطه به صورت رندم در صفحه به عنوان نقاط شروع الگوریتم انتخاب می‌کنیم. شکل «۴»

پاسخ



شکل ۴: نقاط ابتدایی الگوریتم

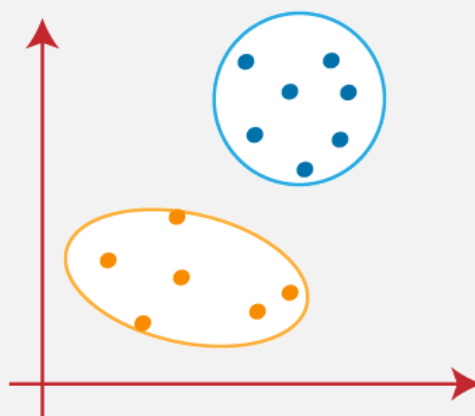
اکنون هر نقطه را به نزدیک‌ترین مرکز اختصاص می‌دهیم. این عملیات با محاسبه فاصله بین نقطه‌ها انجام می‌شود. سپس به مرحله آپدیت مرکز می‌رویم و برحسب واریانس محاسبه کرده، مرکز نقاط را آپدیت می‌کنیم. شکل «۵»



شکل ۵: آپدیت مراکز

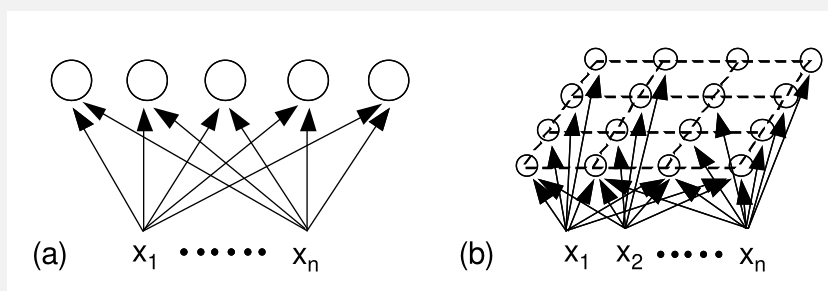
این فرآیند را آنقدر ادامه می‌دهیم تا مینیمم ترین فاصله نقاط از مراکز به دست آید و داده‌ها خوشه‌بندی شود. شکل «۶»

پاسخ



شکل ۶: داده‌های خوشه‌بندی شده

(آ) SOM: SOM بر خلاف K-Means یک شبکه عصبی است که بر اساس یادگیری بدون نظارت کار می‌کند. شبکه SOM کاربردهای مختلفی دارد. کاربرد اصلی شبکه SOM نگاشت داده‌های با بعد بالا به داده‌هایی با بعد پایین است. اما از این شبکه برای خوشه‌یابی نیز استفاده می‌شود شبکه SOM تنها از دو لایه تشکیل می‌شود. (لایه ورودی + خروجی) که لایه خروجی می‌تواند به صورت پشت سر هم و یا در یک ساختار شبکه‌ای قرار گیرند. شکل «۷»



شکل ۷: ساختار شبکه SOM

در فرآیند آموزش وزن‌های متصل از ورودی به خروجی آموزش داده می‌شوند و در نهایت، هر نرون نماینده یک دسته (خوشه) از داده‌های ورودی است. الگوریتم یادگیری در SOM، رقابتی (Competitive) است. یعنی نرون‌های خروجی با هم بر سر نماینده شدن برای داده‌های ورودی رقابت می‌کنند و نرون برنده وزنش به نسبت نرون بازنده بیشتر اصلاح می‌شود. در SOM نیز همانند K-Means اگر تعداد خوشه‌ها را از قبل نمی‌دانستیم می‌بایست عدد بزرگی را برای آن در نظر بگیریم. [۲]

الگوریتم رقابتی در SOM را می‌توان به دو صورت انجام داد.

i. ارسال سیگنال به نرون‌های دیگر

ii. محاسبه فاصله تا ورودی

معمولاً در تمامی شبکه‌های SOM متداول است که از روش دوم استفاده شود اما در ادامه توضیح هر دو روش را خواهیم داد.

## پاسخ

برای روش اول داریم:  
ابتدا می‌بایست هر نرون خروجی‌اش را به صورت زیر تولید کند:

$$u_j = \sum_{i=1}^n w_{ij} x_i$$

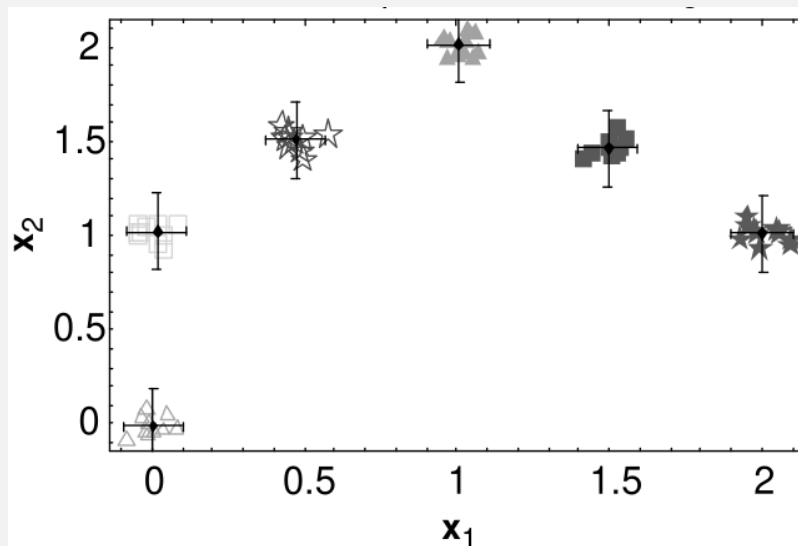
پس از محاسبه  $u_j$  هر نرون خروجی  $u_j$  خودش را با علامت معکوس به تمامی نرون‌های دیگر می‌فرستد. نرون‌ها پس از دریافت سایر  $u_j$  ها، می‌بایست مقادیر  $u_j$  های خودشان را با سایر  $u_j$  های وارد شده جمع کنند. حالا اگر مقدار حاصل از یک آستانه کمتر شود، (مثلا صفر) نرون مربوطه از رقابت خارج می‌شود. این فرآیند تا زمانی ادامه پیدا خواهد کرد که فقط یک نرون باقی بماند و آن نرون به عنوان نرون برنده مشخص می‌شود. همانطور که گفته شد معمولا از این روش استفاده نمی‌شود و از روش دوم استفاده می‌شود. یعنی محاسبه فاصله تا ورودی. در این روش فاصله بردار ورودی طبق یکی از روابط تعیین فاصله (در اینجا فاصله اقلیدسی) برای تمامی وزن‌ها محاسبه می‌شود و نرونی که کمترین فاصله با بردار ورودی را داشته باشد به عنوان نرون برنده مشخص می‌شود.

$$d_j = \|x - w_j\| = \sqrt{\sum_{i=1}^n (x_i - w_{ij})^2}$$

پس از محاسبه همه فاصله‌ها، مقدار وزن نرون برنده به صورت زیر آپدیت می‌شود:

$$\Delta w_j = \beta(x - w_j) = \beta d_j$$

در نهایت، پس از همگراشدن شبکه، داده‌ها همگی در خوشه‌های مربوط به خودشان قرار می‌گیرند. شکل «»



شکل ۸: داده‌های خوشه‌بندی شده شبکه SOM پس از آموزش

از مزایا و معایب این دو الگوریتم می‌توان به موارد زیر اشاره کرد:



پاسخ

## (آ) K-Means:

i. مزایا:

- الگوریتم K-Means نسبت به شبکه SOM از سرعت بالا تری برخوردار است. دلیل این افزایش سرعت، سادگی پیاده‌سازی آن به نسبت SOM است.
- بر روی داده‌های بزرگ به خوبی کار می‌کند

ii. معایب:

- کاملاً وابسته و حساس به مقداردهی اولیه برای نقاط مربوط به K هاست.
- برای خوشه‌یابی داده‌هایی که ساختار غیر محدب دارند، نامناسب است.
- وابستگی شبکه به مشخص کردن تعداد خوشه‌ها پیش از اجرای الگوریتم

## (ب) SOM:

i. مزایا:

- کاربردهای گسترده به جز خوشه‌یابی
- بدست آوردن روابط پیچیده میان داده‌ها

ii. معایب:

- بار محاسباتی بیشتر و سرعت کمتر نسبت به الگوریتم K-Means
- وابسته بودن شبکه به پارامترهای مختلف ورودی مثل نرخ یادگیری، سائز همسایگی و ...

\*

## References

- [1] Data Mining: Practical Machine Learning Tools and Techniques, 2016.
- [2] Neural Networks for Applied Sciences and Engineering, 2006 by Taylor & Francis Group, LLC
- [3] Frey BJ, Dueck D. Clustering by passing messages between data points. science. 2007 Feb 16;315(5814):972-6.
- [4] Sculley D. Web-scale k-means clustering. InProceedings of the 19th international conference on World wide web 2010 Apr 26 (pp. 1177-1178).
- [5] MacQueen J. Some methods for classification and analysis of multivariate observations. InProceedings of the fifth Berkeley symposium on mathematical statistics and probability 1967 Jun 21 (Vol. 1, No. 14, pp. 281-297).
- [6] Comaniciu D, Meer P. Mean shift: A robust approach toward feature space analysis. IEEE Transactions on pattern analysis and machine intelligence. 2002 May;24(5):603-19.
- [7] Ng A, Jordan M, Weiss Y. On spectral clustering: Analysis and an algorithm. Advances in neural information processing systems. 2001;14.

۲. مجموعه داده ارائه شده در این پروژه شامل رویدادهای سه سال متوالی از ۲۰۲۲ تا ۲۰۲۴ است که از سایت ویکی‌پدیا جمع‌آوری شده است. داده‌ی مربوطه را بارگزاری کنید و پیش‌پردازش‌های متنی شامل حذف کلمات ایست (Stop word)، واحدسازی کلمات (Tokenization) و تبدیل به بردارهای GloVe را روی آن انجام دهید.

پاسخ

فایل کد در مسیر Code/Q1-P3.ipynb موجود است. ابتدا کتابخانه‌های مورد نیاز برای کار با متن را نصب می‌کنیم، کتابخانه‌هایی مانند:

- (a) nltk
- (b) textblob
- (c) minisom

پس از نصب کتابخانه‌ها، دیتاست موجود (فایل WikipediaEvents.csv) را می‌خوانیم. ابعاد این دیتاست 1, 473 است.

```
shape of dataset: (473, 1)
[4]:
```

	text
0	January 1, 2022 – The Regional Comprehensive E...
1	January 2, 2022 – Abdalla Hamdok resigns as Pr...
2	January 4, 2022 – The five permanent members o...
3	January 5, 2022 – A nationwide state of emerge...
4	January 6, 2022 – The CSTO deploys a "peacekee...

شکل ۹: دیتاست ورودی

در ادامه، عملیات Pre-processing شامل کوچک کردن تمامی حروف، حذف علائم نگارشی، اعداد و کاراکترهای newline را حذف می‌کنیم. این عملیات‌ها در تابع `clean_text` نوشته شده است. پس از اعمال تابع نام‌برده بر روی دیتاست موجود، همه متن به حروف کوچک تبدیل می‌شود، تاریخ تمامی سال‌ها حذف می‌شود و فقط ماه مورد نظر باقی می‌ماند، همچنین کاراکترهای اضافی مانند - و , نیز حذف شده است. دیتاست پس از انجام این مراحل به صورت زیر می‌شود:

```
[17]: 0    january    the regional comprehensive economic...
      1    january    abdalla hamdok resigns as prime min...
      2    january    the five permanent members of the u...
      3    january    a nationwide state of emergency is ...
      4    january    the cstos deploys a peacekeeping mis...
      ...
      468    october    georgian presidential election
      469    november    namibian general election
      470    november    romanian presidential election
      471    november    lee hsien loong prime minister of s...
      472    december    croatian presidential election
      Name: text, Length: 473, dtype: object
```

شکل ۱۰: دیتاست ورودی پس از حذف علائم نگارشی و کوچک سازی کلمات

## پاسخ

برای پیدا کردن کلمات ایست، در دیتاست، از ماژول stopwords استفاده می‌کنیم. کلمات پرتکرار پیدا شده در این دیتاست به صورت زیر است:

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'you're', 'you've', 'you'll', 'you'd', 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'she's', 'her', 'hers', 'herself', 'it', 'it's', 'its', 'itself', 'they', 'them', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'that'll', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'don't', 'should', 'should've', 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', 'aren't', 'couldn', 'couldn't', 'didn', 'didn't', 'doesn', 'doesn't', 'hadn', 'hadn't', 'hasn', 'hasn't', 'haven', 'haven't', 'isn', 'isn't', 'ma', 'mightn', 'mightn't', 'mustn', 'mustn't', 'needn', 'needn't', 'shan', 'shan't', 'shouldn', 'shouldn't', 'wasn', 'wasn't', 'weren', 'weren't', 'won', 'won't', 'wouldn', 'wouldn't']
```

## شکل ۱۱: کلمات ایست

همچنین پس از پیدا کردن کلمات، برای حذف آنها تابع remove\_stopwords را نوشته‌ایم. پس از حذف، خروجی به صورت زیر می‌شود:

```
[31]: 0      january regional comprehensive economic partne...
      1      january abdalla hamdok resigns prime minister ...
      2      january five permanent members un security cou...
      3      january nationwide state emergency declared ka...
      4      january cstc deploys peacekeeping mission kaza...
      ...
      468     october georgian presidential election
      469     november namibian general election
      470     november romanian presidential election
      471     november lee hsien loong prime minister singap...
      472     december croatian presidential election
      Name: text, Length: 473, dtype: object
```

## شکل ۱۲: خروجی پس از حذف کلمات ایست

پس از این مرحله، نوبت به Tokenization می‌رسد. برای انجام آن از ماژول punkt استفاده می‌کنیم. پس از انجام Tokenization خروجی به صورت زیر می‌شود:

```
[36]: 0      [january, regional, comprehensive, economic, p...
      1      [january, abdalla, hamdok, resigns, prime, min...
      2      [january, five, permanent, members, un, securi...
      3      [january, nationwide, state, emergency, declar...
      4      [january, cstc, deploys, peacekeeping, mission...
      ...
      468     [october, georgian, presidential, election]
      469     [november, namibian, general, election]
      470     [november, romanian, presidential, election]
      471     [november, lee, hsien, loong, prime, minister,...
      472     [december, croatian, presidential, election]
      Name: text, Length: 473, dtype: object
```

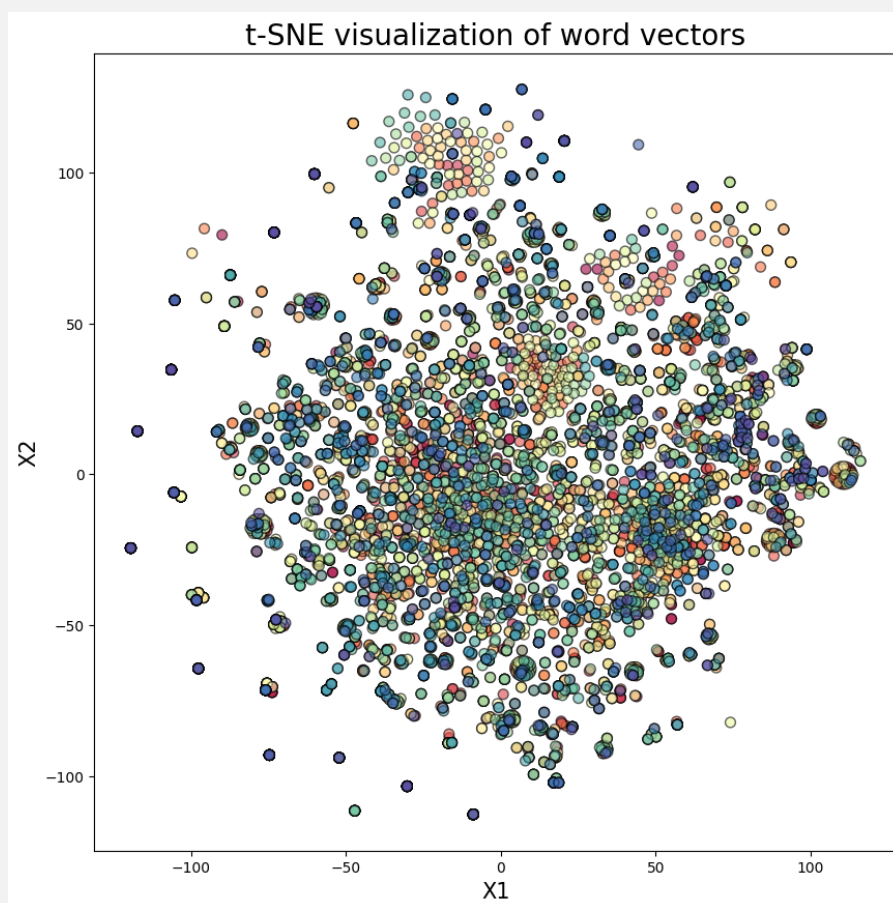
## شکل ۱۳: خروجی عملیات Tokenization

پس از Tokenization می‌بایست تبدیل کلمات به بردار را انجام دهیم. برای انجام این کار، از مدل ازپیش آموزش داده شده GloVe استفاده می‌کنیم. این مدل را می‌توان از اینجا دانلود کنید. پس از fine-tune کردن مدل، از تابع نوشته شده convert\_to\_vector برای تبدیل کلمات به بردار استفاده می‌کنیم.



پاسخ

به دلیل آنکه تعداد کلمات زیاد است و اسامی آن‌ها بر روی آن نوشته شده است و تصویر را شلوغ کرده است، لیبل‌ها را بر روی آن بر می‌داریم و دوباره آن را رسم می‌کنیم:



شکل ۱۶: نمایش برداری کلمات بدون اسامی

۳. پارامترهای ورودی مدل minisom را توضیح دهید. پیرامترهای شبکه خودسازمان‌ده خود را تنظیم کنید و شبکه را بر روی داده‌های مربوطه آموزش دهید. (مقادیر تمامی پارامترها را در گزارش خود اضافه کنید.) سپس به‌ازای هر داده‌ی ورودی واحد، منطق (Best matching unit) با آن را به‌دست آورید و به‌عنوان نمایه‌ی داده‌ی مربوطه ذخیره کنید.

پاسخ

برای آموزش شبکه از کلاس MiniSom استفاده می‌کنیم. ورودی‌های این کلاس و مقادیر اولیه آن‌ها به صورت زیر هستند:

```
__init__(self, x, y, input_len, sigma=1.0, learning_rate=0.5,
decay_function=asymptotic_decay, neighborhood_function='gaussian'
topology='rectangular', activation_distance='euclidean',
random_seed=None)
```

در ادامه به توضیح هر پارامتر می‌پردازیم.

## پاسخ

(آ)  $x, y = (2, 3)$ 

$x, y$  ابعاد نوروهای خروجی را مشخص می‌کند. مثلاً اگر مقدار آن را به ترتیب ۱ و ۵ قرار دهیم بدین معناست که نوروهای خروجی در ساختار پشت‌سرهم در یک ساختار ۱ بعدی قرار می‌گیرند و ۵ نورو در خروجی شبکه داریم. پس در نتیجه ۵ خوشه داریم.

در این مسئله چون از تعداد خوشه‌ها اطلاعی نداریم مقدار آن را (۲,۳) در نظر گرفتیم. یعنی ساختار نوروهای خروجی ۲ بعدی است و ۶ نورو (کلاس) در خروجی شبکه داریم.

(ب) `input_len=vectors_2d.shape[1]`

پارامتر `input_len` طول (بعد) داده‌های ورودی شبکه را مشخص می‌کند و چون در مسئله ما داده‌ها ۲ بعدی هستند، پس این پارامتر را برابر با `vectors_2d.shape[1]` یعنی مقدار ۲ قرار می‌دهیم.

(ج) `sigma=0.5`

پارامتر بعدی `sigma` است. این پارامتر، به نوعی نماینده انحراف معیار است و شعاع همسایگی را مشخص می‌کند. برای مثال در تکرار  $t$  مقدار  $\sigma(t)$  به صورت زیر محاسبه می‌شود:

$$\sigma(t) = \frac{\sigma}{1 + \frac{t}{T}}, \quad T = \frac{\text{number of iteration}}{2}$$

هر چقدر مقدار این پارامتر را کوچکتر بگیریم، دقت خوشه‌بندی بالاتر می‌رود و خطای آموزش کمتر می‌شود. مقدار پیش‌فرض این پارامتر، ۱ است که ما در این مسئله آن را ۰/۵ در نظر گرفتیم.

(د) `learning_rate=0.1`

پارامتر `learning_rate` نرخ یادگیری شبکه را مشخص می‌کند. این ضریب در اصلاح وزن نورو برنده شده خودش را نشان می‌دهد (در سوال اول به‌طور کامل توضیح داده شد). هر چقدر مقدار این پارامتر بزرگ باشد، شبکه ممکن است دچار ناپایداری شود. رنج نرمال این پارامتر در بازه ای بین ۰/۱ تا ۰/۱ قرار دارد. در این کلاس به طور پیش‌فرض مقدار نرخ یادگیری ۰/۵ فرض شده است اما ما در این مثال مقدار آن را ۰/۱ قرار دادیم.

(ه) `decay_function=asymptotic_decay`

این پارامتر، مقدار  $\sigma$  و `learning_rate` را در هر دوره کاهش می‌دهد. مقدار پیش‌فرض آن بر روی تابع `asymptotic_decay` تنظیم شده است. این تابع به صورت زیر نوشته شده است:

```
def asymptotic_decay(learning_rate, t, max_iter):
    return learning_rate / (1+t/(max_iter/2))
```

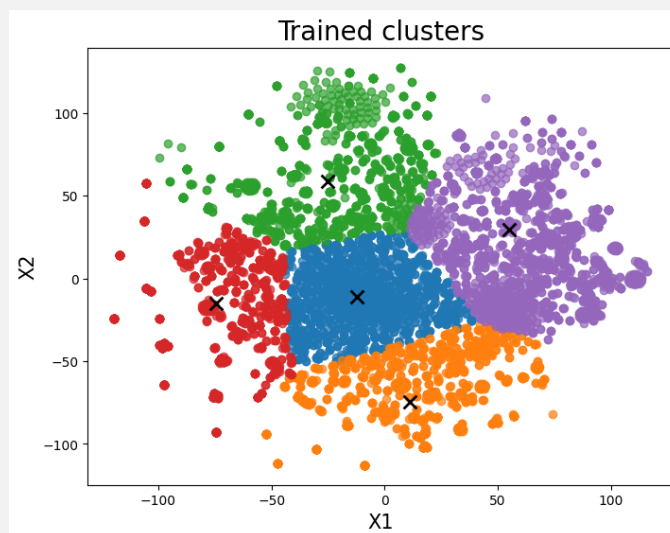
پاسخ

سپس با استفاده از شبکه minisom شبکه را در ۵۰۰۰۰ دوره آموزش می‌دهیم و خوشه‌های مناسب را پیدا می‌کنیم. این عملیات را برای ۵۰ داده رندوم از میان مجموعه داده‌ها مجدداً تکرار می‌کنیم و نتایج آن را گزارش می‌دهیم. به ازای آموزش شبکه با همه داده‌ها در ۵۰۰۰۰ دوره آموزشی و با ۵ نرون در خروجی مقدار خطای شبکه به صورت زیر بدست آمده است:

```
[ 50000 / 50000 ] 100% - 0:00:00 left
quantization error: 33.85513106766581
```

شکل ۱۷: خطای شبکه

خروجی شبکه پس از آموزش به صورت زیر به دست می‌آید:



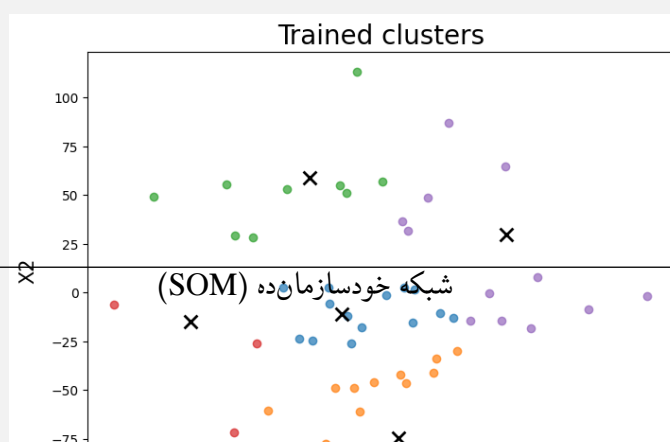
شکل ۱۸: داده‌های خوشه‌بندی شده پس از آموزش

همین کار را برای ۵۰ داده رندوم تکرار می‌کنیم و نتایج را به صورت زیر گزارش می‌دهیم. خطا پس از ۵۰۰۰۰ دوره آموزشی به صورت زیر به دست می‌آید:

```
[ 50000 / 50000 ] 100% - 0:00:00 left
quantization error: 33.85513106766581
```

شکل ۱۹: خطای شبکه برای داده‌های رندوم

همچنین خروجی خوشه‌یابی شده نیز به صورت زیر به دست می‌آید:



۴. برای ۵۰ رویداد که به صورت تصادفی از مجموعه داده انتخاب شده‌اند، نقشه خروجی را رسم کنید. نقشه‌ی به دست آمده را تفسیر کنید.

۵. فرآیند جست‌وجو را به صورت زیر برای سه رویداد دلخواه از سه سال گذشته انجام دهید. (می‌توانید از پرسش‌های موجود در فایل sample\_questions.txt کمک بگیرید.) و خروجی مربوطه را در گزارش خود اضافه کنید.

- تبدیل پرسش به بردار
- پیدا کردن نمایه‌ی متناسب با پرسش مربوطه
- پیدا کردن تمامی داده‌های خارجی نمایه‌ی مورد نظر
- محاسبه معیار شباهت کسینوسی و خروجی دادن بردارهای داده‌های خارجی با شباهت بیشتر از آستانه. (چرا معیار کسینوسی در این مسئله انتخاب مناسبی است؟)