

سوال اول

شبکه های SOM از دو لایه تشکیل می شوند:

۱. لایه اول شامل نورون های ورودی است که ابعاد آن به تعداد ویژگی های داده ورودی است.

۲. لایه دوم نورون های خروجی که نحوه چینش نودها در لایه خروجی شکل خروجی را مشخص می کند.

فرآیند یادگیری وزن ها در شبکه ی کوهونن با استفاده از کمترین فاصله انجام می شود. بدین صورت که فاصله ی هر داده ی ورودی تا تمام نورون های لایه ی خروجی محاسبه می شود و سپس با توجه به اینکه کدام نورون کمترین فاصله را با داده ی مذکور دارد، نورون برنده انتخاب می شود. این عمل در واقع همان انتخاب نزدیکترین مرکز خوشه در الگوریتم- k means است. پس از آنکه نورون برنده مشخص شد، وزن های آن نورون به نحوی تغییر می کنند که به مقدار آن داده نزدیک شوند.

به روزرسانی وزن ها به دو شکل دسته ای و بازگشتی انجام می شود. در حالت بازگشتی در هر مرحله به ازای هر داده، وزن نورون برنده به روزرسانی می شود. در حالت دسته ای در آخر هر اپیاک با توجه به میانگین مقادیر داده های هر نورون، وزن نورون ها بروزرسانی می شود. همانطور که گفته شد نورون برنده، نورونی است که وزن هایش کمترین فاصله را با داده ی مد نظر دارند. فاصله ی ذکر شده می تواند فاصله ی اقلیدسی یا منهتن باشد. پس از آنکه نورون برنده مشخص شد با استفاده از رابطه ی (1) وزن آن تغییر می کند:

$$w_{j+1} = w_j + \beta (x - w_j) \quad (1)$$

که β در این رابطه برابر با نرخ یادگیری است. در شبکه ی کوهونن تنها وزن نورون برنده تغییر نمی کند بلکه تا یک شعاع همسایگی نورون برنده، وزن نورون ها متناسب با یک ضریب تغییر میکند. بنابراین رابطه ی (1) به صورت رابطه (2) بازنویسی می شود:

$$w'_j = w_j + \beta \text{NS}[x - w_j] \quad (2)$$

در این رابطه NS بیانگر ضریبی است که به هر نورون همسایه‌ی نورن برنده تعلق می‌گیرد که وزن‌هایش در راستای مقدار x تغییر کند. شکل‌های همسایگی متفاوتی می‌توان برای این مسئله تعریف کرد مانند: مربعی، دایره‌ای، شش‌ضلعی. همچنین می‌توان با توابع مختلف به همسایه‌ها وزن داد. به طور مثال می‌توان به صورت خطی وزن همسایه‌ها را کاهش داد یا از تابع گوسی استفاده کرد. در این حالت همسایه‌های نزدیک با ضریب بیشتری نسبت به همسایه‌های دورتر تغییر می‌کنند. دلیل بروز رسانی نورون‌های همسایه این است که انتظار می‌رود خوشه‌های نزدیک به هم داده‌هایشان بیشتر شبیه به هم باشد. به همین دلیل وزن نورون‌های نزدیک نورون برنده را در جهت آن داده تغییر می‌دهند.

در این شبکه‌ها روند یادگیری بدین صورت است که نرخ یادگیری و شعاع همسایگی به صورت تدریجی کاهش می‌یابد و وزن‌ها به صورت تدریجی به سمت خوشه‌بندی صحیح حرکت می‌کنند. یادگیری این شبکه‌ها در دوفاز ordering و convergence انجام می‌شود:

- در فاز اول سعی می‌شود توپولوژی نقشه‌ی ویژگی شکل بگیرد.
- در فاز دوم سعی می‌شود نمایش داده‌ها دقیقتر شود.

به همین منظور در فاز اول مقدار اولیه‌ی نرخ یادگیری حدود 0.1 در نظر گرفته می‌شود و در فاز دوم نرخ یادگیری اولیه حدود 0.01. نرخ یادگیری باید در هر مرحله به مقداری کمتر از مرحله‌ی قبلیش تغییر کند. بنابراین برای تغییر نرخ یادگیری می‌توان از روابط ۳ یا ۴ استفاده کرد.

$$\beta_t = \beta_0(1 - t/T) \quad (3)$$

$$\beta_t = \beta_0 \text{Exp}[-t/T] \quad (4)$$

با استفاده از رابطه‌ی ۳ نرخ یادگیری به صورت خطی کاهش می‌یابد و با رابطه‌ی ۴ به صورت نمایی کاهش می‌یابد.

در رابطه با تابع NS که گفته شد می‌تواند تابع گاوسی در نظر گرفته شود لازم به ذکر است که در این قسمت نیز باید بعد از هر مرحله شعاع همسایگی کاهش یابد تا در نهایت یادگیری همگرا شود. شعاع همسایگی آنقدر کوچک می‌شود که در فاز دوم عمال تنها نورون برنده بروز رسانی می‌شود. برای کاهش شعاع همسایگی میتوان مانند کاهش نرخ یادگیری عمل کرد. با آزمایش و خطا به این نتیجه رسیده‌اند حالت بهینه آن است که در فاز اول یادگیری شعاع همسایگی اولیه به قدری بزرگ در نظر گرفته شود که کل نقشه‌ی ویژگی را دربر بگیرد و در فاز دوم شعاع همسایگی بسیار کوچک باشد تا تنها نورون برنده تغییر کند.

نکته: اگر در هر دوفاز تنها نورون برنده بروز رسانی شود و همسایه‌ها بروز رسانی نشوند در واقع یادگیری صورت نمی‌گیرد و همیشه یک تعداد محدودی نورون برنده می‌شوند و دیگر نورون‌ها به اصطلاح می‌میرند.

یک مسئله‌ی مهم دیگر در یادگیری شبکه‌ی خودسازمانده انتخاب وزن‌های اولیه است. برای انتخاب وزن‌ها می‌توان به دو شیوه عمل کرد:

۱. مقداردهی تصادفی به وزن‌ها.

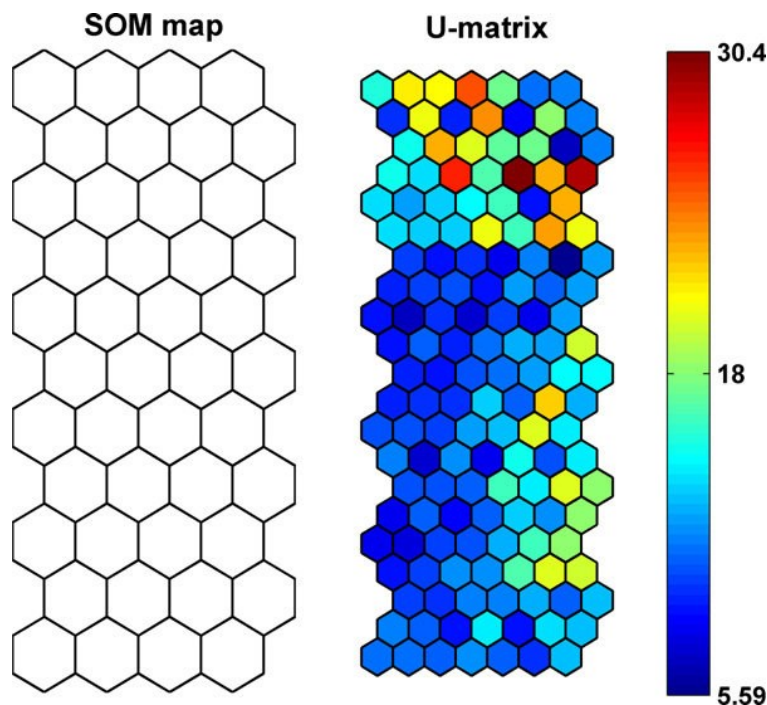
۲. استفاده از تعدادی از داده‌های ورودی برای مقداردهی؛ یعنی مقادیر داده‌ها به عنوان وزن اولیه‌ی نورون‌های لایه‌ی خروجی تنظیم شوند.

حالت دوم باعث تسریع سرعت یادگیری و همگرایی می‌شود.

از شبکه‌ی کوهونن به دو روش می‌توان برای کاهش بعد داده‌های مسئله استفاده کرد:

- در روش اول پس از آن که یادگیری شبکه کامل شد، فاصله‌ی هر داده را با وزن‌های تمام نورون‌ها (یعنی مراکز خوشه‌ها) محاسبه می‌کنیم و به عنوان ویژگی‌های جدید استخراج‌شده در نظر می‌گیریم.
- در روش دوم پس از یادگیری کامل شبکه، به ازای هر داده مختصات نورون برنده‌ی آن داده را در نقشه‌ی ویژگی به عنوان ویژگی جدید در نظر می‌گیریم و این‌گونه ابعاد داده‌ها را کاهش می‌دهیم.

ابزار **مصورسازی** در شبکه کوهونن، **U-Matrix** است. ابعاد U-Matrix با ابعاد شبکه SOM یکسان است. پس از آموزش شبکه، U-Matrix به این صورت ساخته می‌شود که برای هر نورون، میانگین فاصله آن را با همسایه‌های محاسبه کرده و در سلول مربوط به آن در U-Matrix می‌نویسیم. سپس هر کدام از اعدادی که در U-Matrix وجود دارند را به یک رنگ مپ می‌کنیم. به عنوان مثال در شکل ۱ رنگ آبی نشان‌دهنده میانگین فاصله کمتر و رنگ قرمز نشان‌دهنده میانگین فاصله بیشتر است. بنابراین در قسمت‌های آبی رنگ، میانگین فاصله‌ی نورون‌ها با نورون‌های همسایه‌شان خیلی کم است یعنی تراکم داده‌هایی که به این قسمت از نقشه مپ شده‌اند بیشتر بوده است.



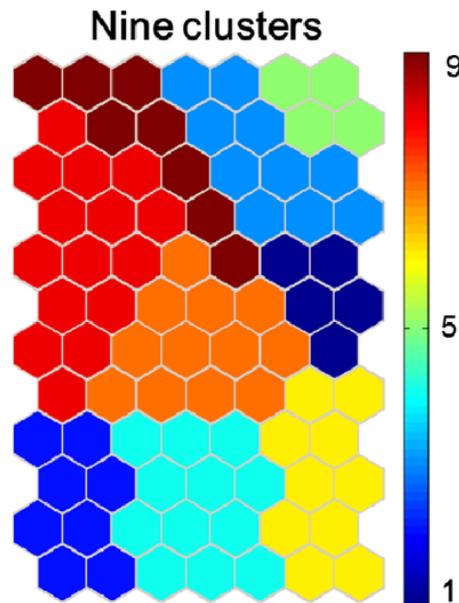
شکل ۱ نمایش U-Matrix و طیف رنگی آن

به دو روش می‌توان U-Matrix را تفسیر کرد:

۱- می‌توان از روی آن ناحیه‌هایی از فضای داده که تراکم در آن بیشتر است را تشخیص بدهیم. زیرا وقتی تراکم داده‌ها در یک محل زیاد است یعنی بردار وزنی این داده‌ها از نظر فاصله اقلیدسی به هم نزدیک هستند پس این داده‌ها مشابه هستند.

۲- می‌توانیم یک تخمین گسسته از توزیع احتمال داده‌ها داشته باشیم. هر جا که تراکم نوروها بیشتر است، تراکم داده‌ها هم بیشتر است

یک روش دیگر برای **مصورسازی** شبکه به این صورت است. با فرض داشتن لیبل‌های کلاسترها، هر کلاستر را با یک رنگ نشان می‌دهیم و رنگ هر نرون به رنگ پرتکرارترین کلاستر مپ شده به آن نرون در می‌آید. به عنوان مثال در شکل زیر با انجام این عمل، مشاهده می‌شود که نوروهای مربوط به یک کلاستر در کنار هم قرار می‌گیرند.



شکل ۲ خوشه بندی به کمک SOM

سوال دوم

توضیحات:

مجموعه داده rcv1 شامل حدود ۸۰۰۰۰۰ داده است که هر داده ۴۷۲۳۶ ویژگی دارد و می‌تواند به یک یا چند کلاس از ۱۰۳ کلاس موجود تعلق داشته باشد. به دلیل حجم بالای داده‌ها امکان کار کردن با کل داده‌ها وجود نداشت به همین دلیل از نمونه‌های ۱۰۰۰۰ تایی و ۱۲۰۰۰ تایی این مجموعه داده استفاده شد.

هم‌چنین به دلیل اینکه تعداد ویژگی‌ها زیاد است و کد موجود در ویدئوها روی این داده‌ها بسیار زمانبر بود، از کتابخانه susi برای پیاده‌سازی شبکه خودسازمانده کوهونن استفاده شد.

سوال سوم

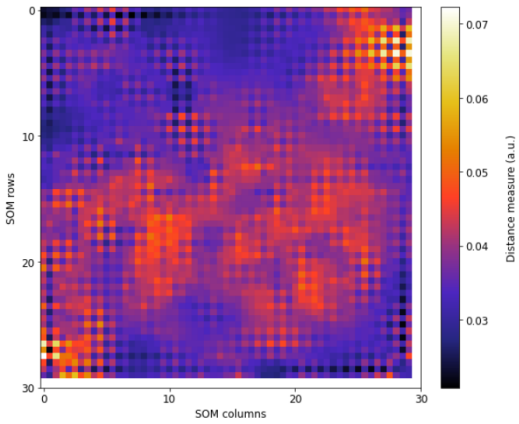
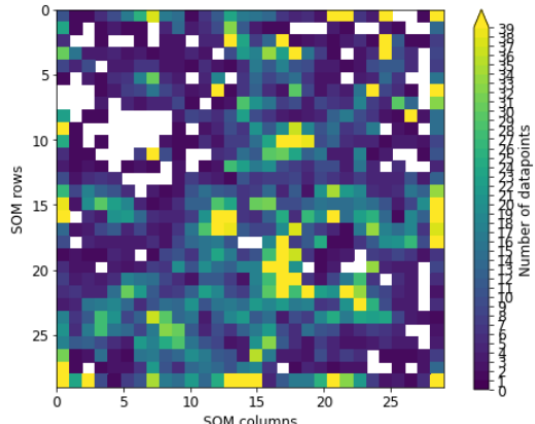
در جدول زیر نقشه ویژگی و هیستوگرام دو بعدی برای ۴ مدل مختلف آورده شده است. تفاوت این مدل‌ها در ابعاد نقشه و هم چنین در تعداد داده‌های نمونه‌گیری شده است.

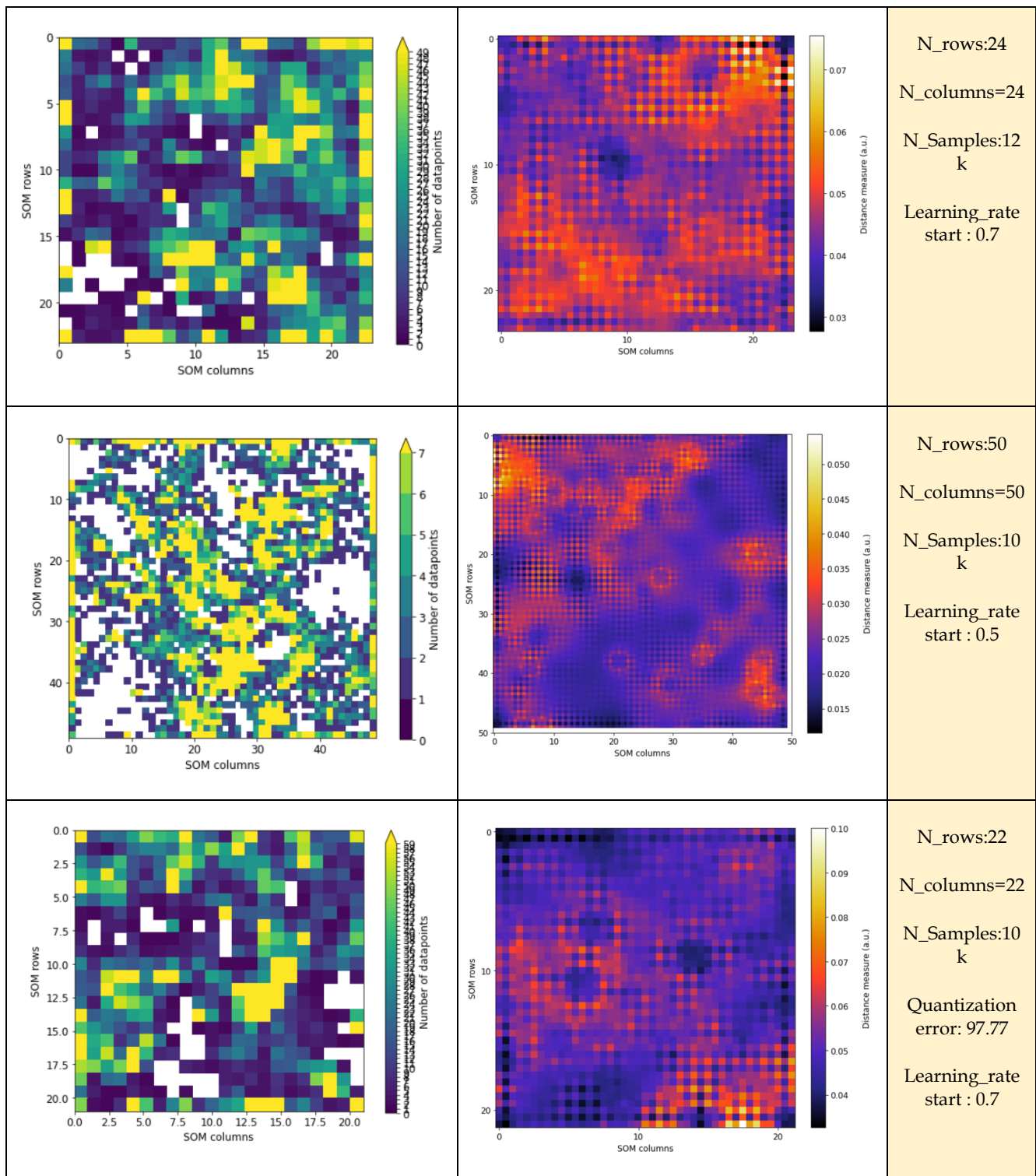
حدود ابعاد نقشه با توجه به یک مقاله که در رابطه با تعداد ابعاد بهینه برای نقشه‌های کوهون بحث کرده است به دست آمده است. طبق ادعای این مقاله نقشه بهینه یک مربع به این ضلع است:

$$\sqrt{5 \times \sqrt{\text{number_of_rows}}}$$

که در آن `number_of_rows` نشان‌دهنده تعداد سطرهای داده است. بنابراین برای یک نمونه ۱۰ هزارتایی از داده‌ها، یک نقشه ۲۲*۲۲ پیشنهاد می‌شود و برای یک نمونه ۱۲ هزارتایی یک نقشه ۲۴*۲۴. این دو نقشه به همراه دو نقشه‌ی دیگر با ابعاد بالاتر بررسی شده‌اند و در جدول زیر آورده شده است.

هم‌چنین یک نمودار دیگر تحت عنوان هیستوگرام دو بعدی نمایش داده شده است. در این نمودار هر خانه نشان دهنده یکی از نورون‌های شبکه کوهون است و رنگ آن خانه نشانگر تعداد داده‌هایی است که به آن نورون مپ شده‌اند. هر چقدر تعداد داده‌های مپ شده به آن نورون بیشتر باشند، رنگ خانه‌ی متناظر با آن روشن‌تر است. لازم به ذکر است که اگر رنگ خانه‌ای سفید باشد به این معنی است که هیچ داده‌ای به نورون متناظر با آن خانه مپ نشده است.

مشخصات مدل	نقشه ویژگی (U matrix)	هیستوگرام دو بعدی (2D histogram)
<p>N_rows:30</p> <p>N_columns=30</p> <p>N_Samples:10 k</p> <p>Quantization error: 97.75</p> <p>Learning_rate start : 0.7</p>		



سوال چهارم:

ویژگی‌های مجموعه داده: بیش از ۸۰۰ هزار داده در آن وجود دارد که هر کدام دارای ۴۷۲۳۶ بعد یا ویژگی هستند. هم چنین هر داده ممکن است به یک یا چند تا از ۱۰۳ کلاس موجود متعلق باشد بنابراین داده‌ها multi-label نیز هستند.

تفسیر `u_matrix`:

با توجه به نمودارهای `u-matrix` ناحیه‌هایی از نمودار که به طیف رنگ بنفش و سیاه تعلق دارند نشان‌دهنده این است که در آن نواحی تراکم داده‌ها بالاست و داده‌های شبیه به هم که میانگین فاصله‌شان با همسایگانشان کم است به آن نقاط مپ شده‌اند. نقاطی که دارای طیف رنگ نارنجی و زرد هستند در آن همسایگی نقاطی از داده مپ شده اند که از هم دورند و میانگین فاصله‌شان با همسایگانشان زیاد است. بنابراین می‌توانیم نتیجه بگیریم تراکم داده‌ها در نواحی بنفش و تیره بیشتر است.

تفسیر خوشه‌بندی:

با توجه به نمودارهای رسم شده در سوال بعدی برای خوشه‌بندی، می‌توان مشاهده کرد که داده‌ها به خوبی جداپذیر نیستند زیرا در برخی از موارد دیده می‌شود که داده‌های دو خوشه با هم ادغام شده‌اند. هم‌چنین می‌توان مشاهده کرد که شکل خوشه‌ها پیچیدگی چندانی ندارد و اشکال پیچیده‌ای مثل مارپیچ و ... در آن دیده نمی‌شود. از لحاظ نویزی بودن می‌توان دید که داده‌ها نويز کمی دارند زیرا داده‌های پرت خیلی کمی در خوشه‌بندی دیده می‌شود.

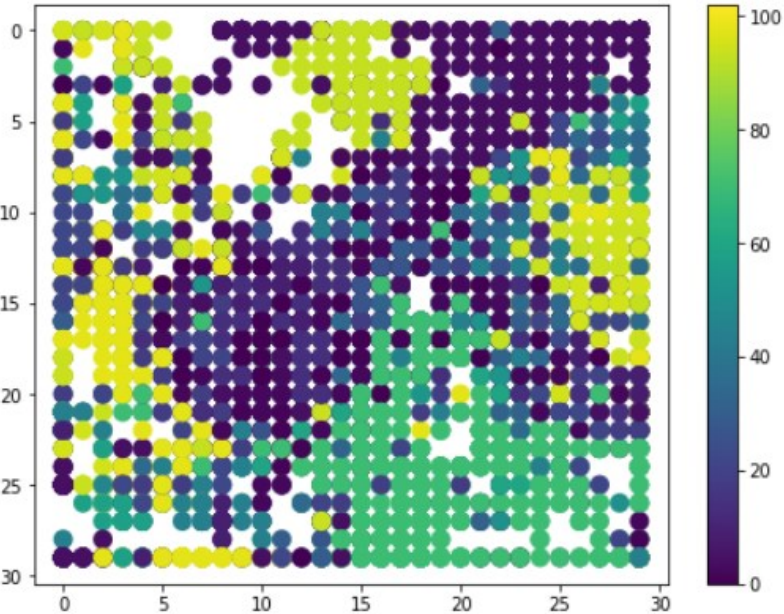
تفسیر هیستوگرام دوبعدی:

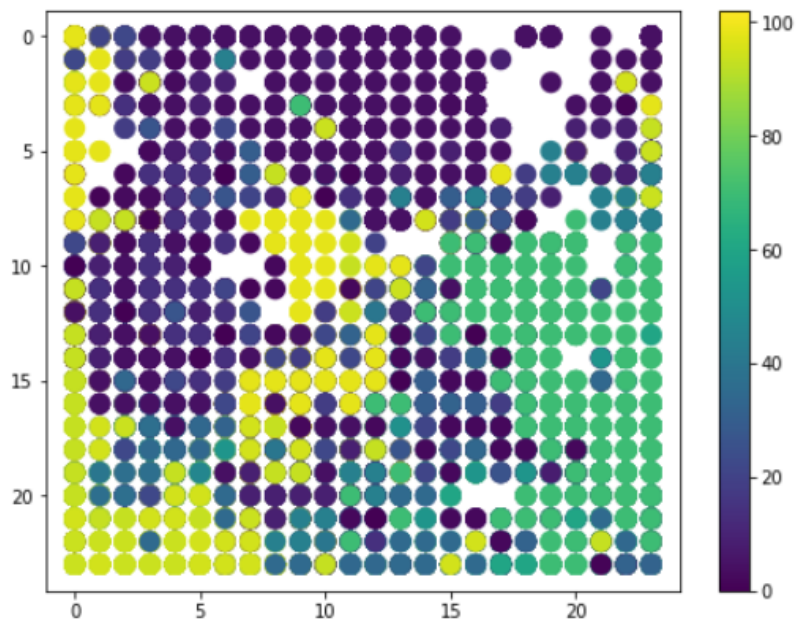
با توجه به هیستوگرام‌های دوبعدی می‌توان مشاهده کرد که هر چقدر ابعاد نقشه بیشتر شده است، تعداد خانه‌های سفید، یعنی خانه‌هایی که هیچ داده‌ای به نوروں متناظر با آن مپ نشده است، افزایش یافته است. بنابراین می‌توان دید که با افزایش بی‌رویه‌ی ابعاد نقشه فقط تعداد نوروں‌های مرده افزایش پیدا می‌کند و تاثیر مثبتی در عملکرد شبکه دیده نخواهد شد.

سوال پنجم:

خوشه‌بندی‌های انجام شده به همراه مشخصات مدل و معیار **purity** در جدول زیر آورده شده است. در واقع خوشه‌بندی به این صورت است که هر نورون نشان‌دهنده یک خوشه است و داده‌هایی که به آن نورون مپ می‌شوند متعلق به آن خوشه هستند. برای مصورسازی خوشه‌بندی، در هر نمودار نورون‌های موجود در لایه خروجی به تصویر درآمدی است و سپس باتوجه به لیبل داده‌هایی که به هر نورون مپ شده‌اند، رنگ پرتکرارترین لیبل به آن نورون اختصاص داده شده است. همانطور که مشاهده می‌شود داده‌هایی که لیبل‌های مشابه داشته‌اند در کنار هم قرار گرفته‌اند و خوشه‌هایی را تشکیل داده‌اند.

نحوه محاسبه **purity** به این شکل است که برای هر نورون لیبل داده‌های مپ شده به آن را در نظر گرفته و تعداد پرتکرارترین لیبل را می‌شماریم. این مقدار را برای تمام نورون‌ها محاسبه کرده و با هم جمع می‌کنیم و در نهایت تقسیم بر تعداد کل داده‌ها می‌کنیم. از آنجایی که هر داده می‌تواند به چند کلاس از ۱۰۳ کلاس تعلق داشته باشد، یک بار به این روش عمل شد که فقط یکی از لیبل‌های هر داده در نظر گرفته شد و یک بار به این روش عمل شده که تمام لیبل‌ها در نظر گرفته شده‌اند که مقادیر **purity** در حالت دوم بیشتر است.

مشخصات مدل	نمودار خوشه‌بندی
N_rows:30	
N_column=30	
N_Samples:10k	
Learning_rate_start : 0.7	
Purity_1: 0.87	
Purity_2: 0.62	



N_rows:24

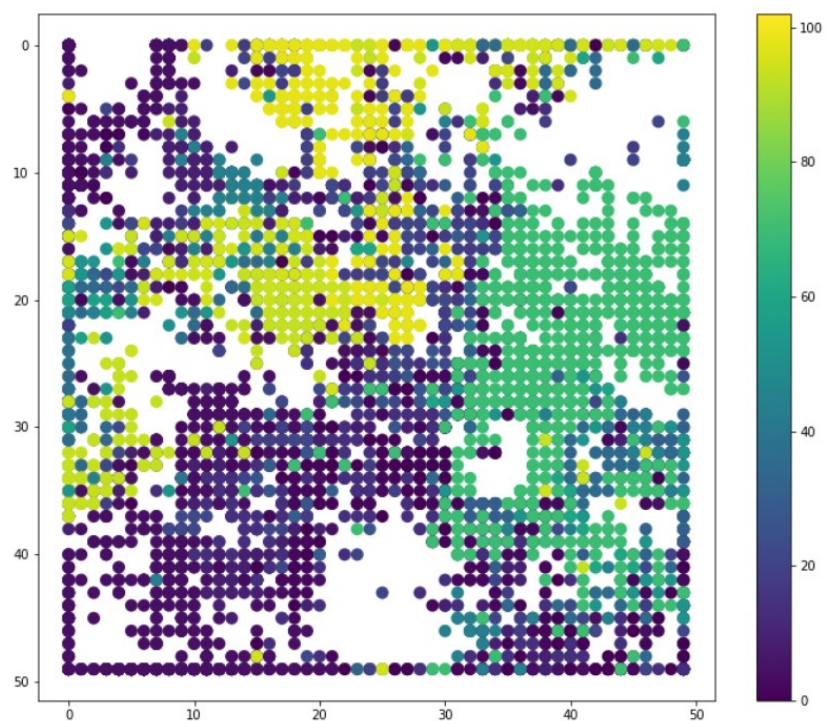
N_columns=24

N_Samples:12k

Learning_rate_start : 0.7

Purity_1: 0.88

Purity_2: 0.60



N_rows:50

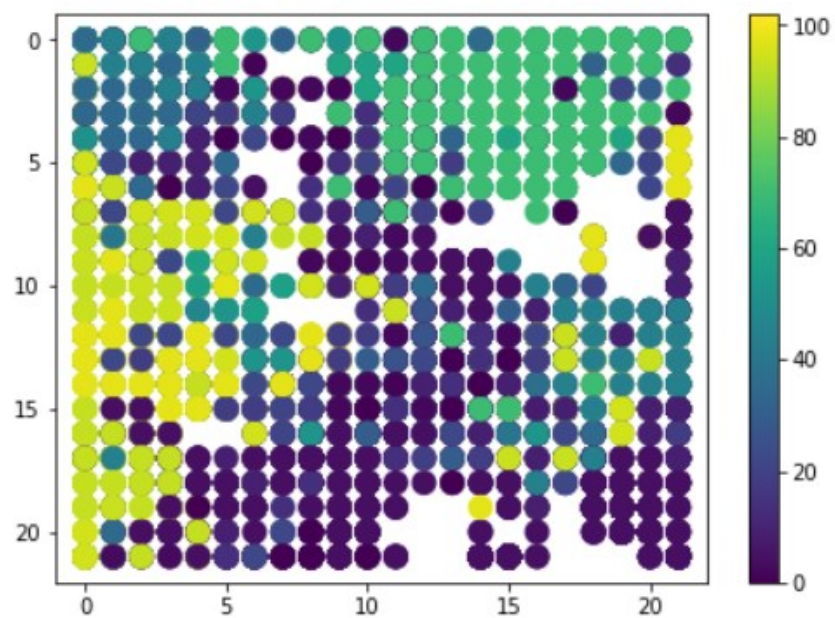
N_columns=50

N_Samples:10k

Learning_rate_start : 0.5

Purity_1: 0.89

Purity_2: 0.67



N_rows:22

N_columns=22

N_Samples:10k

Learning_rate_start : 0.7

Purity_1 : 0.88

Purity_2: 0.61