به نام خدا

گزارش تمرین سری دوم درس هوش محاسباتی

دکتر مزینی

محمد یارمقدم

96462104

سوال یک

مورد A

در این سوال هدف پیاده سازی شبکه kohonen **با ضریب یادگیری و شعاع همسایگی ثابت است.**

بدین منظور ابتدا داده ورودی سوال را تهیه میکنیم که یک آرایه 1600 **تایی از** RGB **های مپ ورودی خواهد بود. برای مقدار دهی اولیه به رنگ های دیتای ورودی از مقادیر رندوم استفاده میکنیم. پس از تکمیل آرایه، اعضا را بر بزرگ ترین عنظر تقسیم میکنیم تا نرمالایز شود و محاسبات ساده تر صورت گیرد.**

در قسمت بعد یک نمونه از کلاس شبکه موردنظر با ضریب یادگیری و تعداد iteration **های** epoch **میسازیم و سپس تابع** train **را روی آن فراخوانی میکنیم.**

در قسمت کانستراکتور کلاس موردنظر مپ موردنظر برای آموزش دیتای ورودی به صورت رندوم مقداردهی اولیه میشود.

سپس مقادیر epoch **و شعاع همسایگی و ضریب یادگیری طبق مقادیر پاس داده شده از** main **برنامه مقداردهی میشوند.**

در تابع train **برای هر کدام از پیکسل های دیتای ورودی تفاوت رنگ با هر کدام از خانه های مپ** kohonen **طبق فرمول زیر محاسبه میشود.**

به این شکل که جذر جمع تفاوت سه رنگ دو پیکسل محاسبه شده و از روی آن اختلاف دو پیکسل بدست میاید و سپس طبق مقایسه با یک متغیر که در ابتدا مقدار بی نهایت دارد کوچک ترین این اختلاف بدست می آید

سپس تابع update_weights **صدا زده میشود و نزدیک ترین پیسکل به پیکسل مپ در دیتای ورودی با این پیکسل مج میشود. بدین ترتیت تمامی پیکسل های فضای ورودی بر روی نقشه خروجی مپ شده و تصویر خروجی در نهایت بر اساس آرایه** RGB **های پیکسل های چاپ میشود.**

مورد B

کد این مورد نیز مانند قبل است با این تفاوت که ضریب یادگیری ثابت مشکلاتی به همراه دارد. اگر یک ضریب یادگیری مناسب انتخاب شود، مدل میتواند به خوبی آموزش دیده و خروجی مناسب را تولید کند. اما در اکثر موارد ضریب ثابت باعث میشود که همگرایی رخ نداده و به اندازه باشد که مینیموم محلی کوهونن را هر بار رد کرده و نتواند رنگ های مشابه را کنار هم نگه دارد و اجازه نمیدهد رنگ ها کنار هم فیت شوند. اما کم کردن ضریب یادگیری و شعاع همسایگی باعث میشود ابتدا رنگ های کلی را کنار هم قرار دهد و سپس جزیی تر بررسی کند. یعنی به طور مثال طیف رنگ قرمز را یک سمت و طیف آبی را یک سمت قرار دهد.

از سوی دیگر شعاع همسایگی اگر در ابتدا بزرگ و متغیرباشد باعث میشود که ابتدا هر نورون کل مپ را به سمت خود بکشد و سپس با کوچک کردن شعاع تنها 3 یا 4 نورون اطراف خود را به سمت خود بکشد باعث میشود در مپ گرده ایجاد نشود و ضریب یادکیری متغیر هم از عدم همگرای جلوگیری می کند.
برای حل این مشکل با پیشروی در epoch ها ضریب یادگیری هم آپدیت میکنیم و به روز نگه میداریم.

به این پدیده map distortion میگویند که به معنی وجود گره در مپ مورد نظر در هنگام آموزش است.

سوال دو

مورد A

در این مورد نیز هدف پیاده سازی شبکه MLP برای پیش بینی تابع سینوس است.

همانند تمرین قبل یک مدل از کلاس sequential تولید میکنیم و سپس با متد Add لایه های مورد نظر را به آن اضافه میکنیم. برای این مورد ما دو لایه میانی برای این مدل قرار میدهیم و لایه آخر را تابع فعالساز قرار میدهیم.

سپس بعد از تشکیل مدل توابع fit و predict را روی آن فراخوانی میکنیم و نتیجه را روی نمودار با استفاده از کتابخانه pyplot رسم میکنیم.

مورد B

هدف در این سوال پیاده سازی شبکه RBF **برای پیش بینی تابع سیسنوس است. برای این منظور ابتدا** training data **را در تابع** rbf **تعریف میکنیم که در واقع یک تابع گوسی است.**

در مرحله بعد ما نیاز داریم که دوایر مورد نیاز برای این شبکه را یافته و دیتا را دسته بندی کنیم. برای این کار از الگوریتم k-means clustering **استفاده میکنیم تا مرکز دوایر را بیابیم.**

در این تابع ابتدا دوایر به صورت رندوم مقدار دهی اولیه میشوند . سپس تا زمانی که همگرایی اتفاق نیفتد درآرایه distances **فاصله مرکز هر دایره با هر نقطه محاسبه میشود . کوچک ترین فاصله هر نقطه تا مرکز دایره اطرافش در آرایه** closetcluster **ذخیره میشود. سپس نقاط قرار گرفته درون فضای دایره موردنظر در آن قرار میگیرند.**

پس ارز آن اگر دایره ها حرکت نکرده باشند همگرایی انجام میشود و یک کپی از دایره ها گرفته میشود. سپس دوایر با یک یا صفر نقطه در clusterwithnopoint **نگه داشته میشوند. اگر چنین دایره ایی وجود داشته باشد یعنی باید به گروه های دیگر اضافه شود که این کار در ادامه انجام می پذیرد.**

در سازنده کلاس RBF **هم مقادیر پارامتر های مختلف مثل ضریب یادگیری و** epoch **و متغیر** k **مقدار دهی میشوند.**

و وزن ها و بایاس هم به صورت رندوم مقدار دهی میشوند.

در قسمت مین برنامه توابع fit **و** predict **روی نمونه ایی که از کلاس** RBF **ساخته شده صدا شده میشود و سپس دیتای پیش بینی شده و اصلی روی نمودار رسم میشوند.**

در تابع fit **در خطوط ابتدایی انتخاب میکنیم که از** deviations **هایی که از الگوریتم بالا بدست میاید استفاده کنیم یا همه** base **ها را مجبور به استفاده** deviation **استاندارد که از فرمول بدست میاید بکنیم.**

در ادامه این تابع نیز مانند تمرین قبل ما داده نمونه را از طریق forward pass **و وزن ها و بایاس را از طریق** backward pass **آپدیت می کنیم.**

مورد C

در این قسمت نیز نتایج دو مدل را با هم مقایسه میکنیم.

در کل طبق مشاهدات مدل RBF **بسیار سریع تر از** MLP **است. مزیت مدل** RBF **این است که با دیتای ورودی کمتر میتواند پاسخ پیش بینی شده و جوال مسئله را یافت.**

در مدل RBF **تابع فعالسازی توابع گوسی بر حسب فاصلع نقاط ورودی از مرکز دوایر است اما در** MPL **این تابع می تواند هر تابع غیز خطی باشد.**

در MLP **نیز مانند اسمش امکان وجود جندین لایه میانی موجود است اما در** RBF **لایه میانی واحد است.**

در RBF **لایه آخر تابع فعالسازی ندارد اما در** MLP **امکان دارد این تابع موجود باشد اما میتواند موجود هم نباشدکه در این سوال به این گونه است.**

نمودار مقایسه این دو شبکه در نوت بوک موجود است.