به نام خدا

**دانشگاه تهران**

**دانشکده مهندسی برق و کامپیوتر**

# Trustworthy AI

## تمرین شماره یک

| | |
|---|---|
| نام و نام خانوادگی | سیدسروش مجد |
| شماره دانشجویی | دانشجوی مهمان |
| تاریخ ارسال گزارش | ۲۴ فروردین ۱۴۰۲ |

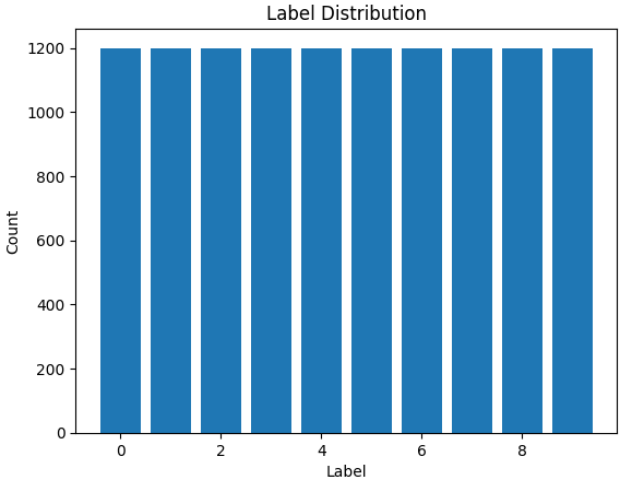
## فهرست گزارش سوالات

## سوال ۱ – Generalization and Robustness

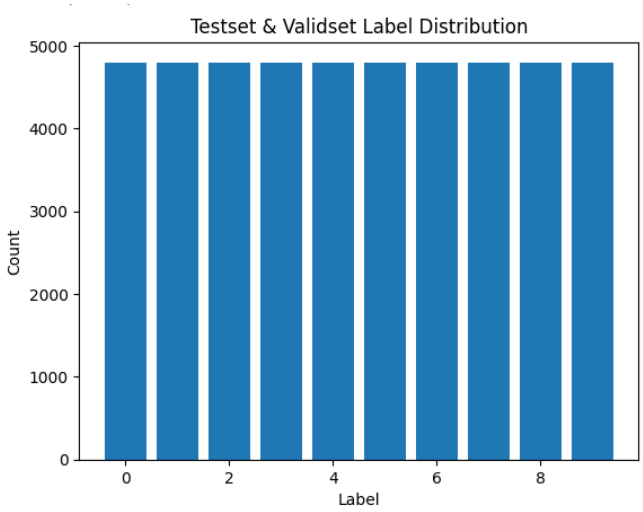
در این تمرین قرار است با داده کم مدلی آموزش دهید که علاوه بر قدرت تعمیم (Generalization) از مقاومت(Robustness) خوبی نیز برخوردار باشد.

۱) داده آموزش CIFAR10 را در نظر بگیرید. در این تمرین تنها ۲۰ درصد این داده را برای آموزش خود جدا می‌کنید و بقیه برای برازش مدل استفاده می‌شود. توجه داشته باشید که تناسب بین کلاس‌ها برقرار باشد.

۲۰ درصد از داده‌های Cifar10 ( ۱۲هزار نمونه) که کلا ۶۰ هزار نمونه است برای آموزش مدل و بقیه ۸۰ درصد (۴۸ هزار نمونه) برای Validation و Test استفاده شدند. تناسب بین کلاس‌ها نیز برقرار است.



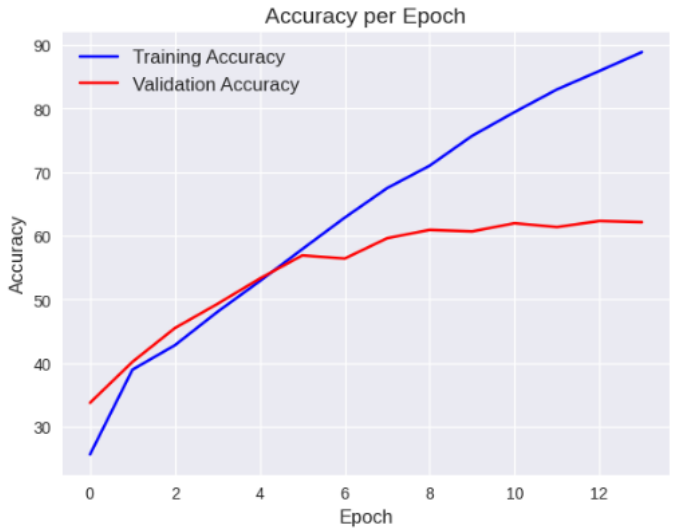

شکل ۱: نمودار توزیع لیبل برای داده **Train**



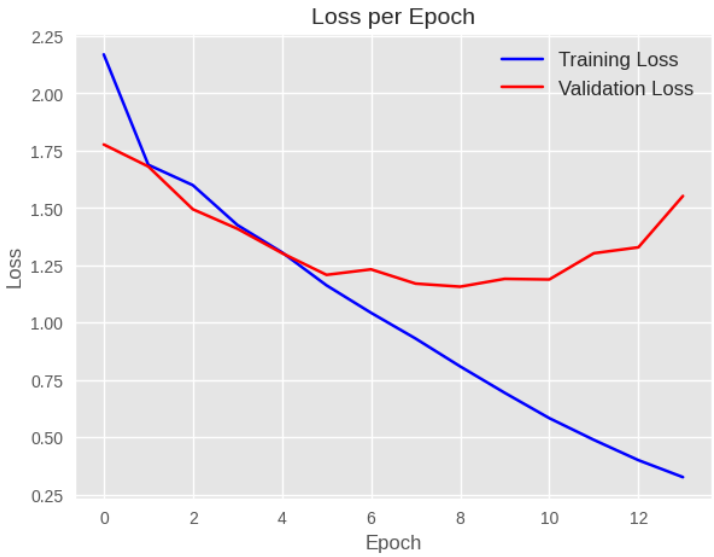

شکل ۲: نمودار توزیع لیبل برای داده **Test** و **Validation**

۲) یک مدل ResNet18 را با استفاده از تابع هزینه CrossEntropy آموزش دهید. دقت آن را در دادگان برازش و تست گزارش کنید. خروجی قسمت کانولوشنال شبکه را کاهش ابعاد داده و آن را برای داده دیده نشده نمایش دهید.

از تابع هزینه Cross-Entropy برای آموزش مدل Resnet18 (با لایه آخر ده تایی) از قبل آموزش داده نشده. از اپتیمایزرAdamW با Learning Rate برابر با ۰.۰۱ و weight decay برابر با ۰.۰۱ و سایز بچ برابر با ۱۲۸ استفاده شد. دقت نهایی بر روی داده تست **۶۲.۶۶** به دست آمد. نمودارهای Accuracy و Loss در شکل ۲ و ۳ نشان داده شده‌اند. این مدل را کمی Overfit کردم تا تاثیر حمله Adversary بر روی آن بهتر مشخص شود.

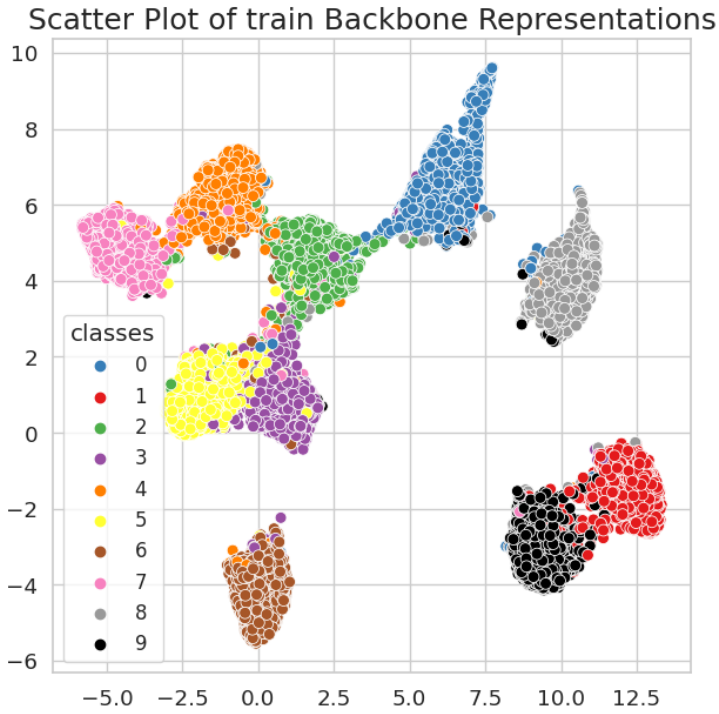



شکل ۳: نمودار **Accuracy** داده‌های **Validation** و **Training** بر حسب **Epoch** حین آموزش

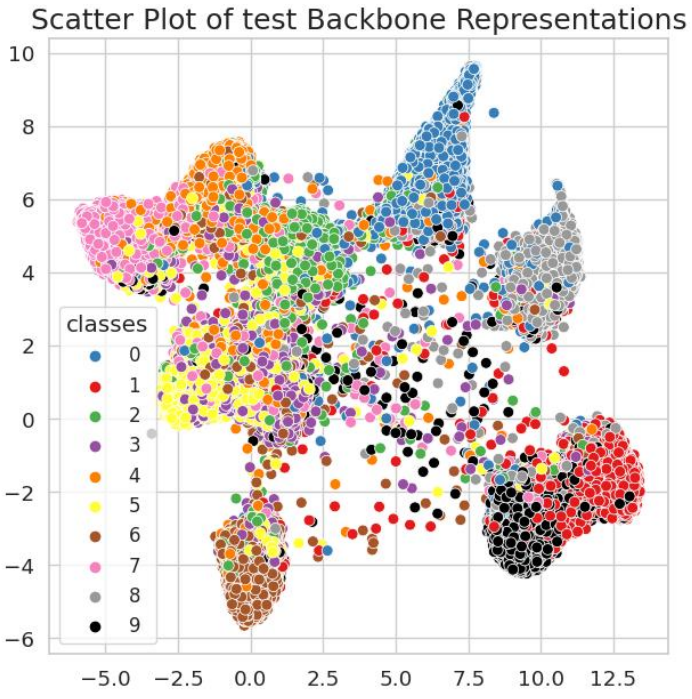



شکل ۴: نمودار **Loss** داده‌های **Validation** و **Training** بر حسب **Epoch** حین آموزش

سپس خروجی بخش Backbone شبکه را با استفاده از Umap به دو بعد برای نمایش کاهش دادیم. ابتدا Umap روی داده آموزش (Unsupervised) fit و سپس برای تست transform شد. شکل ۴ Umap برای امبدینگ داده‌های آموزش و شکل ۵ Umap برای امبدینگ داده‌های تست را نمایش می‌دهد.




شکل ۵: کاهش ابعاد و نمایش **Representation** داده **Train** بخش **Backbone** مدل با استفاده از **Umap**




شکل ۶: کاهش ابعاد و نمایش **Representation** داده **Test** بدون اغتشاش بخش **Backbone** مدل با استفاده از **Umap**

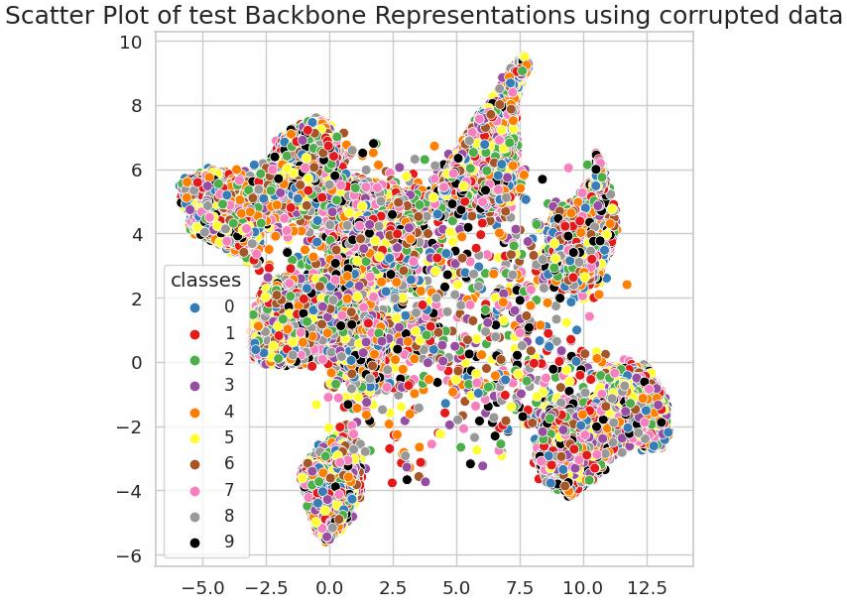
۳) اغتشاشی در داده دیده نشده توسط مدل ایجاد کنید و مجددا دقت و تصویر خروجی backbone را بدست آورید. نتایج را با قسمت قبل مقایسه کنید. اغتشاش شما باید شامل آگمنتیشن هایی مانند نویز، color jitter و ... باشد. همچنین باید یک حمله متخاصمانه مانند Fast Gradient Method به شبکه بزنید.

برای ایجاد اغتشاش از Color jitter و Rotation خیلی کم (۵ درجه) استفاده شد و همچنین ۱۰ تا از پیکسل‌های عکس‌ها که ۳۲ در ۳۲ هستند انتخاب شده و مقدار Intensity یکی از چنل‌های RGB برابر با ۱ قرار داده شد. نمونه‌ای از عکس‌های دارای اغتشاش در شکل زیر نمایش داده شده است. این اغتشاش داده ورودی را از توزیع داده بدون اغتشاش خارج نمی‌کند و در حدی است که انسان برای تشخیصشان به مشکل نمی‌خورد.

شکل ۷: عکس‌های دارای اغتشاش

دقت مدل آموزش دیده با دیتای بدون اغتشاش بر روی داده تست unseen دارای اغتشاش: ٪**۴۵.۲۱**

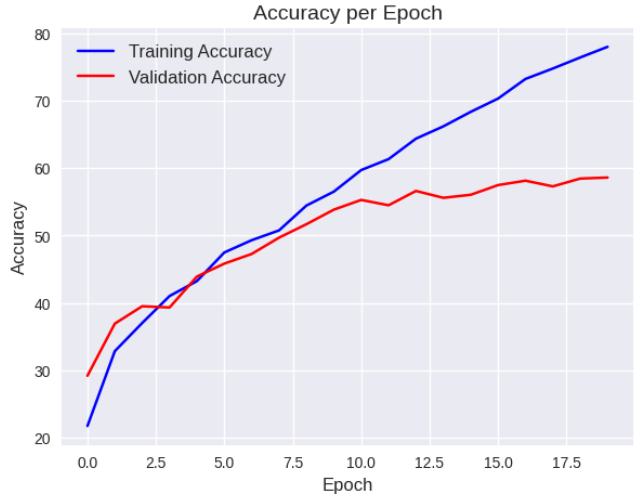
دقت مدل آموزش دیده با دیتای بدون اغتشاش بر روی داده تست unseen دارای اغتشاش و Fast Gradient Method هنگام تست: ٪**۱۷.۵۷**



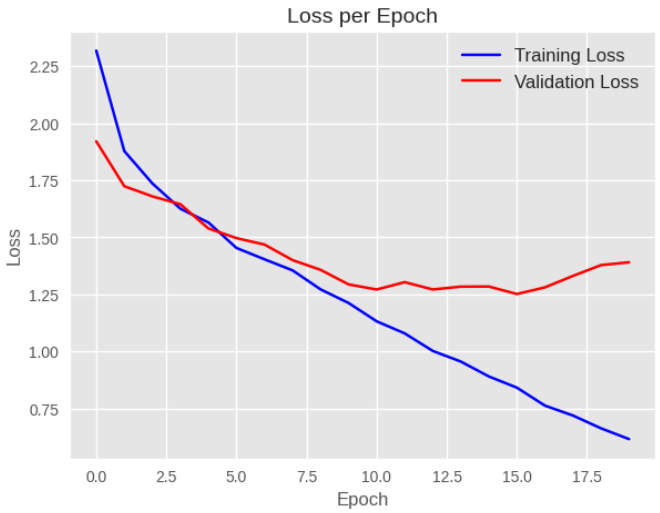

شکل ۸ کاهش ابعاد و نمایش **Representation** داده **Test** دارای اغتشاش بخش **Backbone** مدل با استفاده از **Umap**

۴) قسمت ۲و ۳را با adversarial exampleها مجددا تکرار کنید و نتایج را با حالت بدون adversarial example مقایسه کنید. Adversarial exampleهای شما باید شامل اغتشاشاتی که در قسمت ۳ اعمال کردید باشد.

می‌دانیم که مدل ما تلاش می‌کند از هر فیچری استفاده کند تا دقتش را بالا ببرد. در این حین ممکن است از فیچرهایی بهره برد که دارای Correlation پایینی با لیبل‌ها و Non-Robust هستند. با اضافه کردن اغتشاش به داده Train تلاش می‌کنیم کاری کنیم که مدل برای بالا بردن دقت در زمان Train، کمترین استفاده را از فیچرهای دارای Correlation پایین با لیبل‌ها را ببرد و از فیچرهای Robust استفاده کند. برای آموزش مدل در این بخش با داده‌های اغتشاشی و Adversary Examples در قسمت ۳ استفاده شده است.

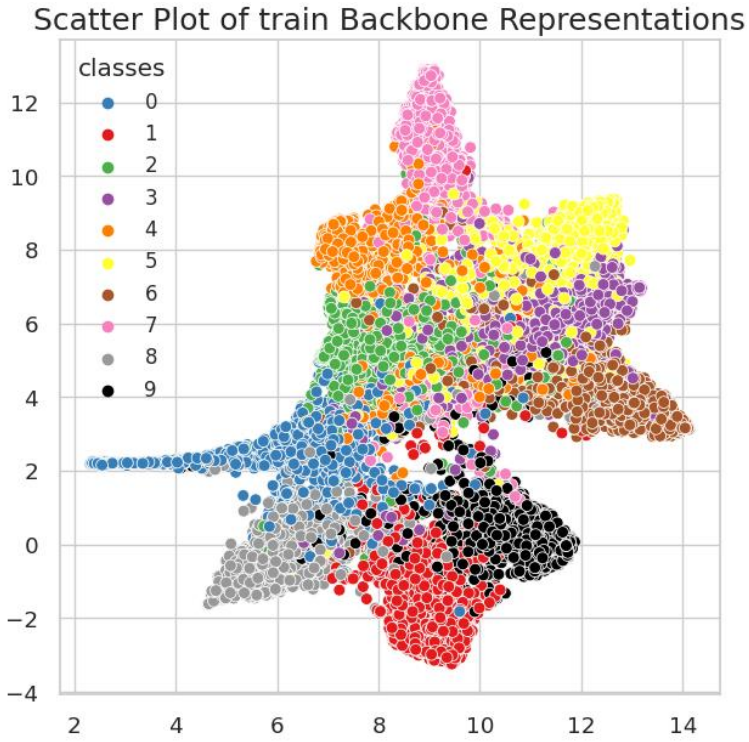



شکل ۹: نمودار **Accuracy** داده‌های **Validation** و **Training** با اغتشاش بر حسب **Epoch** حین آموزش

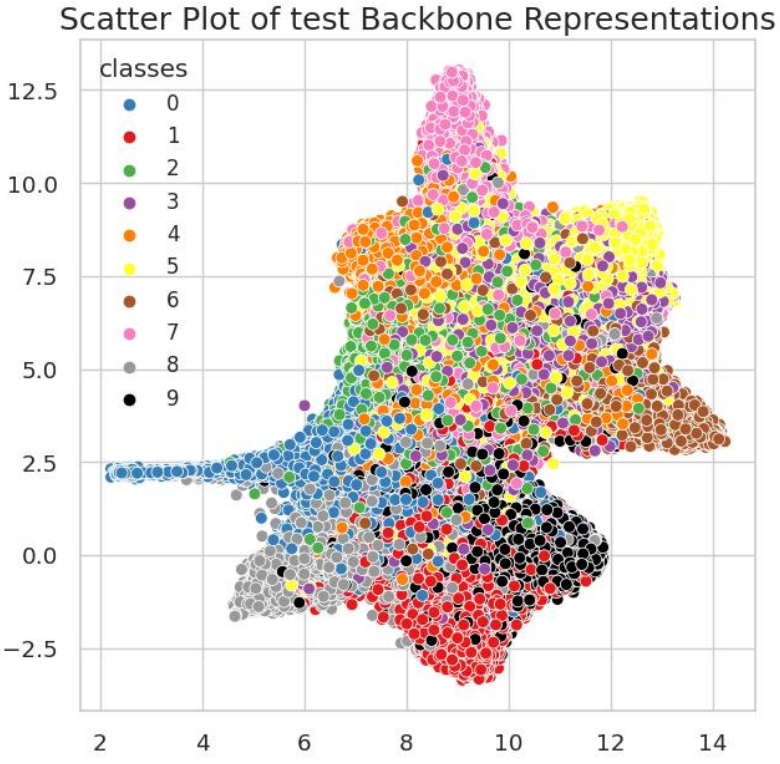



شکل ۱۰: نمودار **Loss** داده‌های **Validation** و **Training** دارای اغتشاش بر حسب **Epoch** حین آموزش

به دلیل اینکه مدل را اورفیت نکردم در نمایش یومپ برای داده‌های تست در شکل ۱۱ مانند قسمت قبل Representationهای کلاس‌های مختلف کاملا از یکدیگر جدا نشده‌اند.




شکل ۱۱: کاهش ابعاد و نمایش **Representation** داده **Train** دارای اغتشاش بخش **Backbone** مدل با استفاده از **Umap**




شکل ۱۲: کاهش ابعاد و نمایش **Representation** داده **Test** دارای اغتشاش بخش **Backbone** مدل با استفاده از **Umap**

دقت مدل آموزش دیده با دیتای دارای اغتشاش بر روی داده تست unseen دارای اغتشاش: ٪**۵۸.۳۵**

دقت مدل آموزش دیده با دیتای دارای اغتشاش با داده تست unseen دارای اغتشاش و با FGM هنگام تست: ٪**۳۵.۳۰**

نسبت به بخش ۳ مدل برای پیشبینی داده‌های تست از فیچرهای Robustتری استفاده کرده است و دقت برای داده‌های تست دارای اغتشاش و با اعمال حمله خصمانه Fast Gradient Method وضعیت بهتری دارد. مدل آموزش دیده با داده اغتشاشی بدون اعمال FGM و داده اغتشاشی با اعمال FGM (منظور اعمال FGM در لوپ آموزش است) از لحاظ دقت و مقاومت خیلی فرقی با یکدیگر نداشتند.

دقت مدل آموزش دیده با دیتای دارای اغتشاش و FGM بر روی داده تست unseen دارای اغتشاش: ٪**۴۹.۲۴**

دقت مدل آموزش دیده با دیتای دارای اغتشاش و FGM با داده تست unseen دارای اغتشاش و با FGM هنگام تست: ٪**۲۸.۷۹**

۵) در مورد تابع هزینه AngularLoss توضیح دهید.

Angular Loss Function دریادگیری عمیق برای کارهایی مانند تشخیص چهره و تخمین وضعیت استفاده می‌شود. برخلاف روش‌های دیگر یادگیری متریک، که بیشتر برای بهینه‌سازی Absolute Distanceها (مانند تابع هزینه Contrastive) یا Relative Distanceها (مانند توابع هزینه Triplet و N-pair) متمرکز هستند، تابع هزینه زاویه‌ای پیشنهاد می‌کند بر روی محدودیت زاویه‌ای در سمپل Negative مثلث‌های سه‌گانه (Anchor, Negative, Positive) تمرکز کنیم. به این صورت که این زاویه از مقدار خاص alpha کمتر باشد. این محدودیت باعث می‌شود فاصله بین Positive و Anchor کم و فاصله Negative از Positive و Anchor زیاد شود. در روش‌های قبلی فاصله بین Anchor و Negative و فاصله Positive و Anchor لحاظ می‌شدند ولی در این روش با محدودیت زاویه‌ای هر سه فاصله (Anchor و Negative) و (Positive و Anchor) و (Positive و Negative) لحاظ می‌شوند و در نتیجه رباستنس بیشتر می‌شود. همچنین در تابع هزینه Triplet انتخاب مرز m مناسب سخت است ولی انتخاب alpha با توجه به تفسیرپذیر بودن هندسی آن آسان‌تر است و زاویه، متریکی ثابت نسبت به چرخش و Scale است که رباستنس بیشتری به تغییرات زیاد در بردار ویژگی فراهم می‌کند.

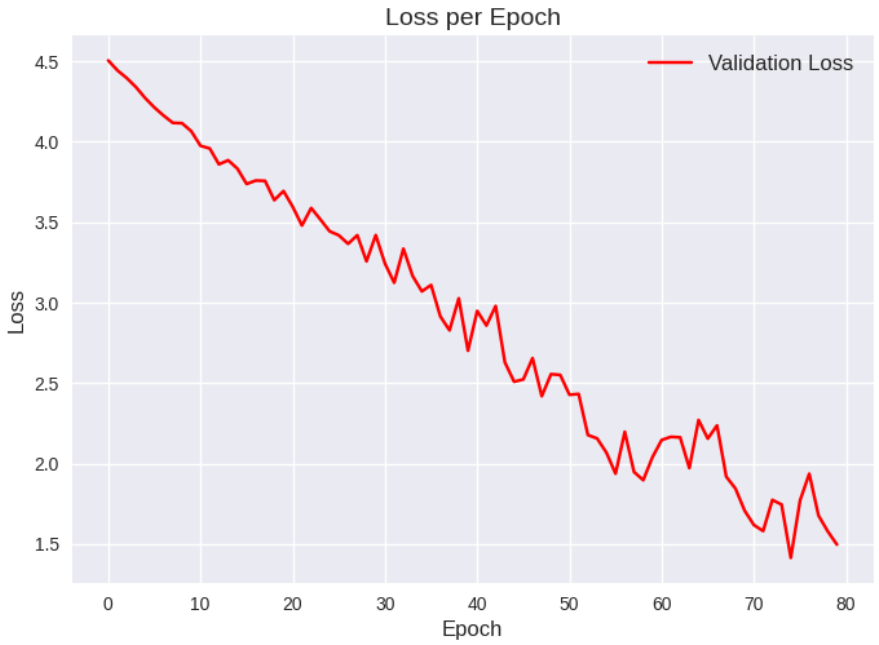
تابع هزینه زاویه‌ای: (در صورتی که زاویه نود Positive بیشتر از ۹۰ باشد شبکه می‌تواند با نزدیک کردن Negative به Anchor زاویه نود Negative را کم کند. برای غلبه بر این مشکل تابع هزینه زیر پیشنهاد شد)

$$l_{ang}(\mathcal{T}) = \left[\|\mathbf{x}_a - \mathbf{x}_p\|^2 - 4\tan^2\alpha\|\mathbf{x}_n - \mathbf{x}_c\|^2\right]_+.$$

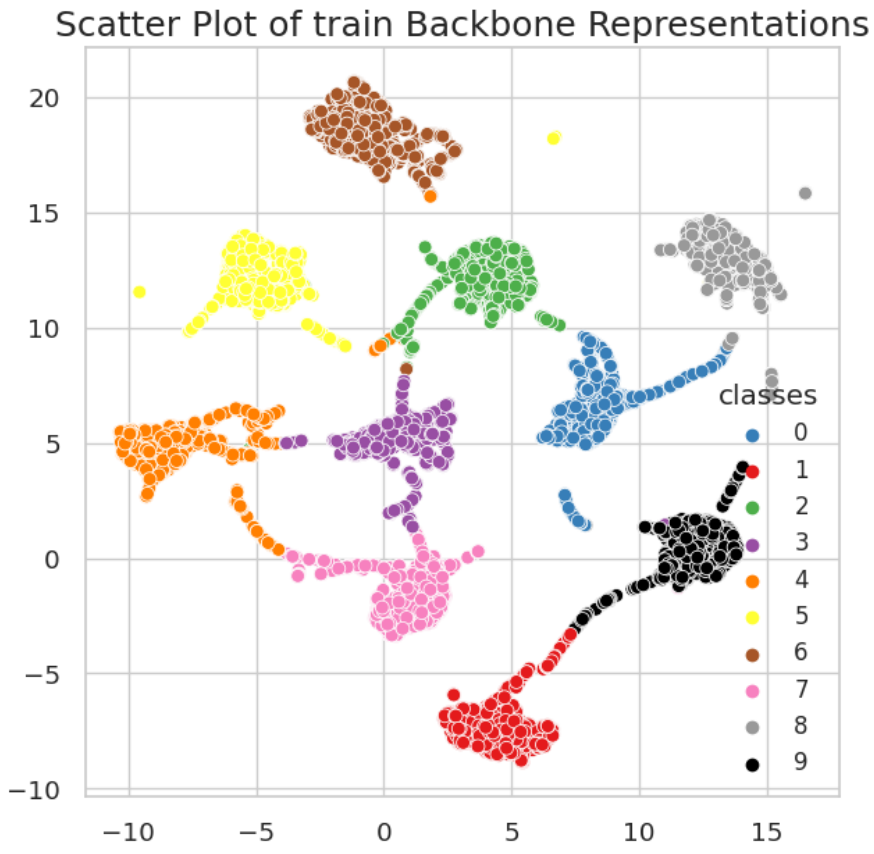
برای مثال از این تابع هزینه بردارهای ویژگی که با Forward Pass کردن تصاویر ورودی از طریق یک شبکه عمیق مانند عصبی کانولوشنی و استخراج خروجی یک لایه خاص به دست می‌آیند را در نظر بگیرید. تابع هزینه زاویه‌ای با ایجاد محدودیت alpha، بردارهای ویژگی نمونه های متعلق به یک کلاس را در فاصله کسینوسی به یکدیگر نزدیک و بردارهای ویژگی کلاس‌های متفاوت را از هم دور می‌کند. نشان داده شده است که تابع هزینه زاویه ای در بهبود دقت مدل‌های یادگیری عمیق در کارهایی مانند تشخیص چهره و تشخیص اشیا، به ویژه در هنگام مواجهه با تغییرات یا اغتشاش‌های بزرگ، موثر است.

۶) بار دیگر قسمت ۲ و ۳را برای تابع هزینه AngularLossانجام دهید و نتایج را با دو حالت قبل مقایسه کنید. (راهنمایی: توابع هزینه pytorch metric learning عمل ساخت تریپلت‌ها را بصورت خودکار انجام می‌دهند. تنها نکته‌ای که باید به آن توجه کنید این است که بچ شما باید شامل تعداد خوبی از کلاس‌ها باشد. برای این کار می‌توانید از BatchSampler استفاده کنید یا یک کلاس Datasetیا DataLoader کاستوم بنویسید.

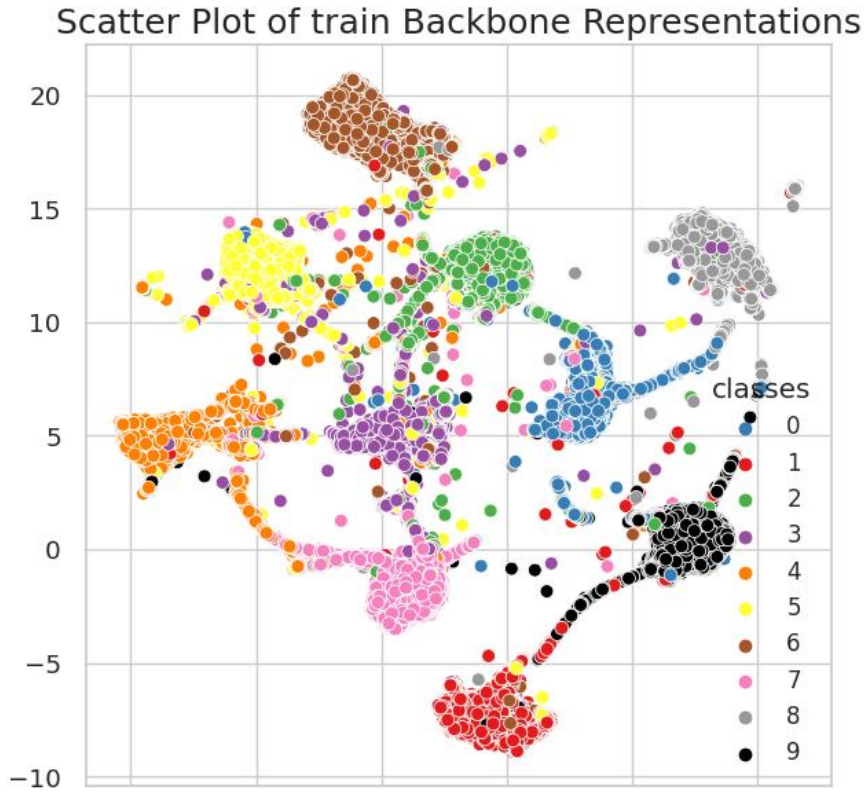
برای آموزش مدل با تابع Angular Loss از مدل resnet18 که Pre-train شده و Adam Optimizer با Learning Rate برابر با ۰.۰۰۱ استفاده شد. لایه آخر به بعد ۱۲۸ مپ و خروجی آن نرمالایز شد. در این بخش مدل را فقط با داده عادی و بدون اغتشاش آموزش می‌دهیم و مقاومت و تعمیم‌پذیری مدل را می‌سنجیم. برای سنجش دقت نیز به دلیل اینکه مدل به ما فقط Embedding می‌دهد از K-Nearest-Neighbour جهت Clustering استفاده شد. همچنین به کمک Batch Sampler، Batch ها با سایز ۱۰۰ دارای ۱۰ سمپل از هر کلاس می‌باشند.



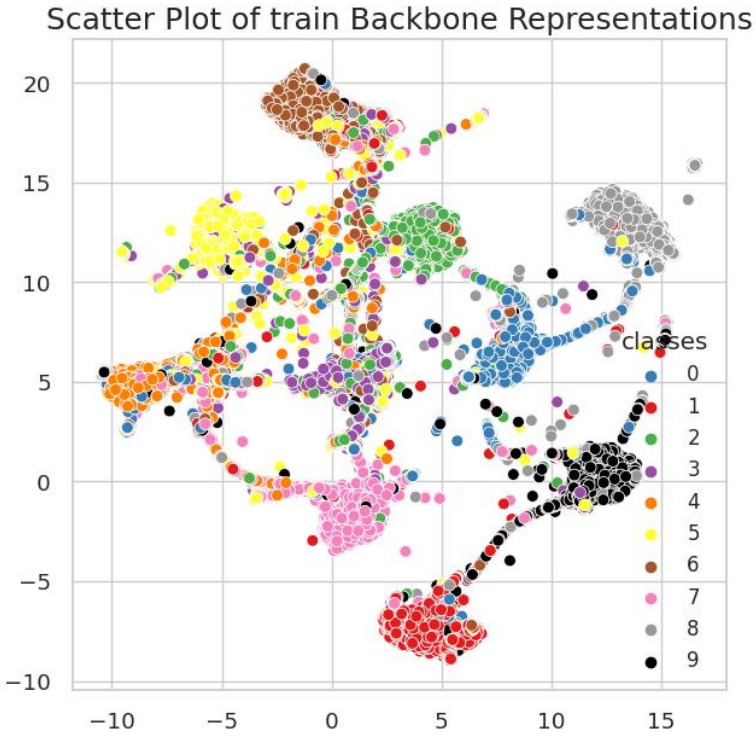

شکل ۱۳: نمودار **Loss** برای داده **Valid** غیر مخدوش در هنگام آموزش مدل




شکل ۱۴: کاهش ابعاد و نمایش **Representation** داده **Train** بدون اغتشاش مدل با استفاده از **Umap**




شکل ۱۵: کاهش ابعاد و نمایش **Representation** داده **Test** بدون اغتشاش مدل با استفاده از **Umap**




شکل ۱۶: کاهش ابعاد و نمایش **Representation** داده **Test** دارای اغتشاش مدل با استفاده از **Umap**

دقت مدل آموزش دیده با دیتای بدون اغتشاش بر روی داده تست unseen بدون اغتشاش: **۷۸.۹۶**٪

دقت مدل آموزش دیده با دیتای بدون اغتشاش با داده تست unseen دارای اغتشاش و FGM: **۵۲.۵۳**٪

مشاهده می‌شود که مدل علاوه بر Generalization بالا (دقت خیلی خوب حدود ۸۰ درصد در داده تست unseen در حالتی که با فقط ۲۰ درصد داده‌ها آموزش دیده است) Robustness بالایی نیز دارد (دقت ۵۲ درصد هنگام تست با داده مخدوش) و مارجین خوبی نیز مهیا می‌کند. در واقع بدون آموزش با Adversarial Examples و داده‌های اغتشاشی هنگام مواجه شدن با داده دارای اغتشاش دقتش خیلی افت نمی‌کند و نسبتا Robust است. دلایل این دقت بالا در قسمت ۵ در توضیح مفهوم Angular-Loss شرح داده شده است.