به نام خدا

دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

Trustworthy AI

تمرین شماره ۱

نام و نام خانوادگی	سارا رستمی
شماره دانشجویی	۸۱۰۱۰۰۳۵۵
تاریخ ارسال گزارش	۱۴۰۲.۰۱.۲۶

فهرست گزارش سوالات

سوال ۱ – Generalization and Robustness

بخش ۱

با استفاده از تابع data_loaders مجموعه‌داده‌ی CIFAR10 را لود کرده و ۲۰ درصد مجموعه‌ی train این دیتاست را به عنوان داده‌ی آموزش و ۸۰ درصد دیگر را به عنوان داده‌ی برازش(validation) جداکردیم.

بخش ۲

از کتابخانه‌ی torchvision مدل Resnet18 را import کردیم. مدل از پیش آموزش نیافته بود. سپس با داده‌های train جدا شده در بخش قبل آن را آموزش و برازش کردیم. و در نهایت با داده‌های تست مدل را ارزیابی کردیم. آموزش را با پارامترهای batch_size = 128، gamma=0.5، step_size=10 ، learning_rate = 1e-3 و epochs = 15 انجام دادیم. نمودارهای accuracy و loss مدل بر حسب هر ایپاک را در شکل ۱ مشاهده می‌کنید. در این شکل، خط نارنجی مربوط به train و خط آبی مربوط به validation می‌باشد. مدل به مرور overfit می‌شود و دقتش به چیزی حدود ۱۰۰ درصد می‌رسد. و روی داده‌های برازش به حدود ۶۱ درصد می‌رسد.

شکل ۱- نمودار **Accuracy** و **loss** مدل **Resnet18** آموزش یافته با داده‌های استاندارد **train** بر حسب ایپاک

دقت مدل آموزش‌یافته روی داده‌های train و داده‌های test را در شکل ۲ مشاهده می‌کنید.

```
Test Accuracy:  0.6121
Train Accuracy 0.9977
```

شکل ۲-دقت مدل **Resnet18** آموزش یافته با داده‌های استاندارد **train** روی داده‌های **test** و **train**

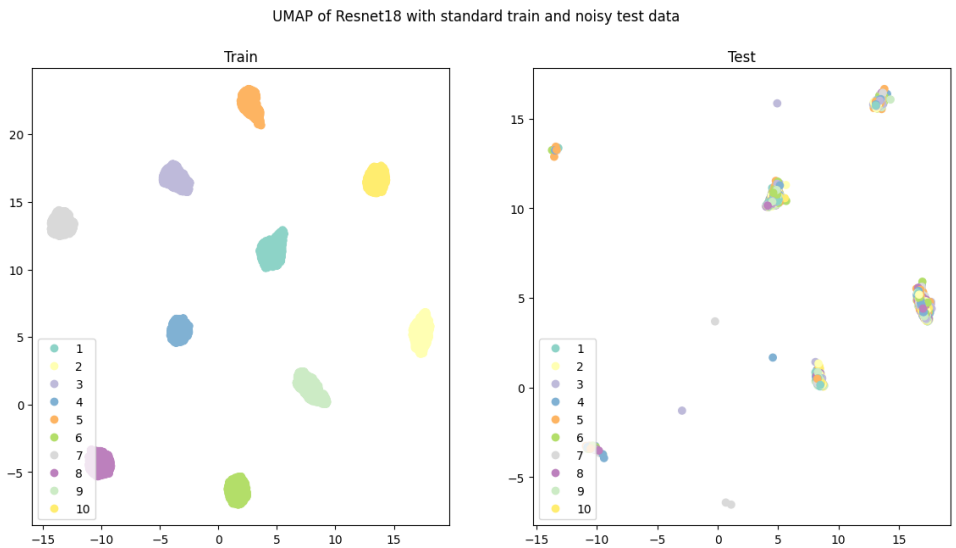
سپس، خروجی backbone این شبکه را توسط UMAP به دو بعد کاهش داده و نمایش می‌دهیم. در شکل ۲ این نمودارها را برای داده‌های test و train مشاهده می‌کنید.

شکل ۳- نمایش **UMAP** بازنمایی حاصل از **Resnet18** آموزش یافته با داده‌های آموزش و تست استاندارد

طبق نمودارهای شکل ۳، دقت داده داده‌های train طبق انتظار تقریبا ۱۰۰ درصد است و دقت داده‌های تست همانطور که در شکل ۲ گزارش شد، ۶۱ درصد بوده که در بازنمایی UMAP آن هم قابل مشاهده‌است.

بخش ۳

در قسمت اول این سوال، اغتشاشی روی داده‌های تست اعمال می‌کنیم. این اغتشاش‌ها شامل colorjitter، نویز گوسی استاندارد و چرخش تصادفی تا حداکثر ۱۰ درجه می‌باشد. مدل Resnet18 آموزش یافته روی داده‌های train استاندارد (از بخش قبل) را روی داده‌های مخدوش شده تست می‌کنیم. نمایش بازنمایی‌های تولید شده توسط این مدل با استفاده از UMAP را در شکل ۴ مشاهده می‌کنید. همانطور که می‌بینید، مدل آموزش یافته یا داده‌های train استاندارد، عملکرد بدی روی داده‌های تست نویزی دارد.



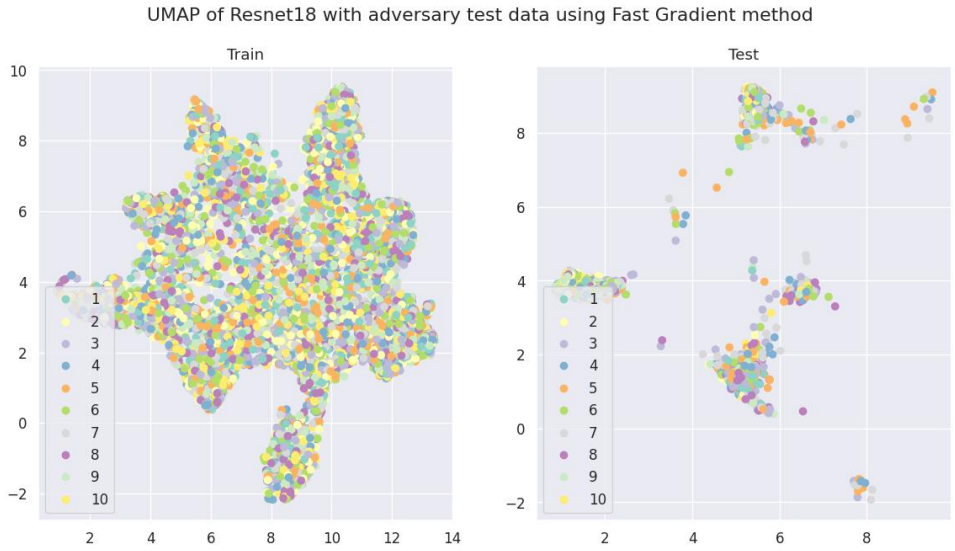
شکل ۴- نمایش **UMAP** بازنمایی حاصل از **Resnet18** آموزش یافته با داده‌های آموزش استاندارد و تست نویزی

در قسمت دوم این سوال، با استفاده از روش Fast Gradient یک adversary attack طراحی کردیم (Epsilon = 0.2) و داده‌های حاصل از این حمله را به مدل دادیم. نتایج این کار را در شکل ۵ مشاهده می‌کنید.

```
Accuracy on test images: 17 %
Accuracy on train images: 1 %
```

شکل ۵- دقت مدل **Resnet18** آموزش یافته با داده‌های آموزش استاندارد حاصل از حمله **adversary**

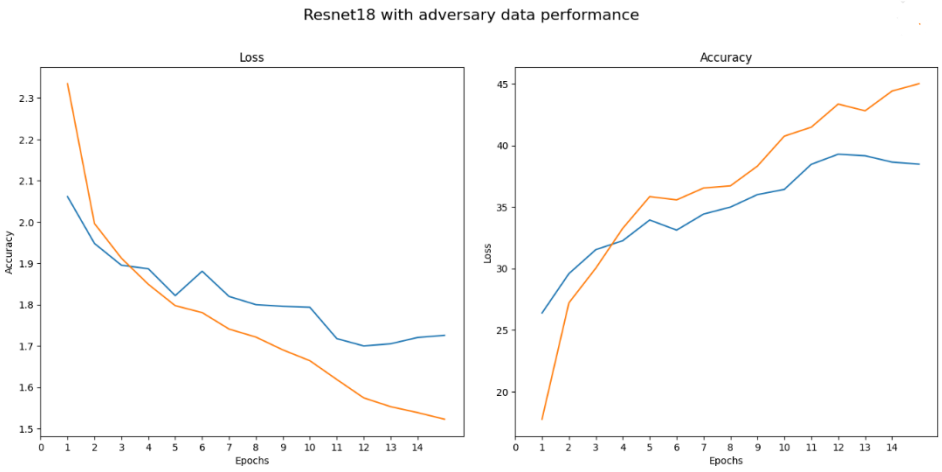
طبق انتظار دقت مدل در اثر این حمله به شدت پایین آمده است. بازنمایی حاصل از داده‌های test و train روی این مدل توسط UMAP را در شکل ۶ مشاهده می‌کنید.



شکل ۶- نمایش **UMAP** بازنمایی حاصل از **Resnet18** آموزش یافته با داده‌های آموزش استاندارد و تست حاصل از حمله **adversary**

بخش ۴

در این بخش برای تولید نمونه‌های adversary، اغتشاشات استفاده شده در بخش ۳ (یعنی colorjitter، نویز گوسی استاندارد و چرخش تصادفی تا حداکثر ۱۰ درجه) را به کار بردیم. عملکرد مدل آموزش‌یافته با داده‌های adversary را در شکل ۶ مشاهده می‌کنید. همانطور که در شکل ۶ می‌بینید، مدل به دقت حدود ۴۵ درصد روی داده‌های train و حدود ۴۰ درصد روی داده‌های برازش می‌رسد.



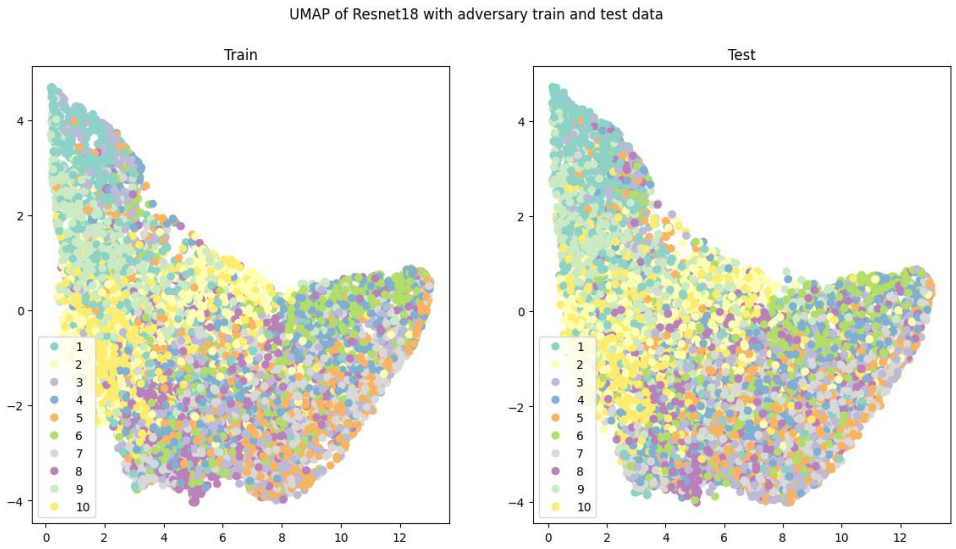
شکل ۷- نمودار **Accuracy** و **loss** مدل **Resnet18** آموزش یافته با داده‌های آموزش **adversary** بر حسب ایپاک

در شکل ۷ دقت این مدل را روی داده‌های آموزش و تست adversary مشاهده می‌کنید.

```
Test Accuracy:  0.387
Train Accuracy 0.4593
```

شکل ۸- دقت مدل **Resnet18** آموزش یافته با داده‌های آموزش **adversary** روی داده‌های آموزش و تست **adversary**

می‌بینیم که این مدل در مقایسه با مدل آموزش یافته با داده‌های استاندارد عملکرد خیلی بهتری روی داده‌های نویزی دارد. به عبارتی مدلی که با داده‌های نویزی آموزش یافته، robustness بیشتری نسبت به نویز پیدا می‌کند. نمایش بازنمایی‌های تولید شده توسط این مدل روی داده‌های آموزش و تست با استفاده از UMAP را در شکل ۸ مشاهده می‌کنید.



شکل ۹- نمایش **UMAP** بازنمایی حاصل از **Resnet18** آموزش یافته با داده‌های آموزش و تست نویزی

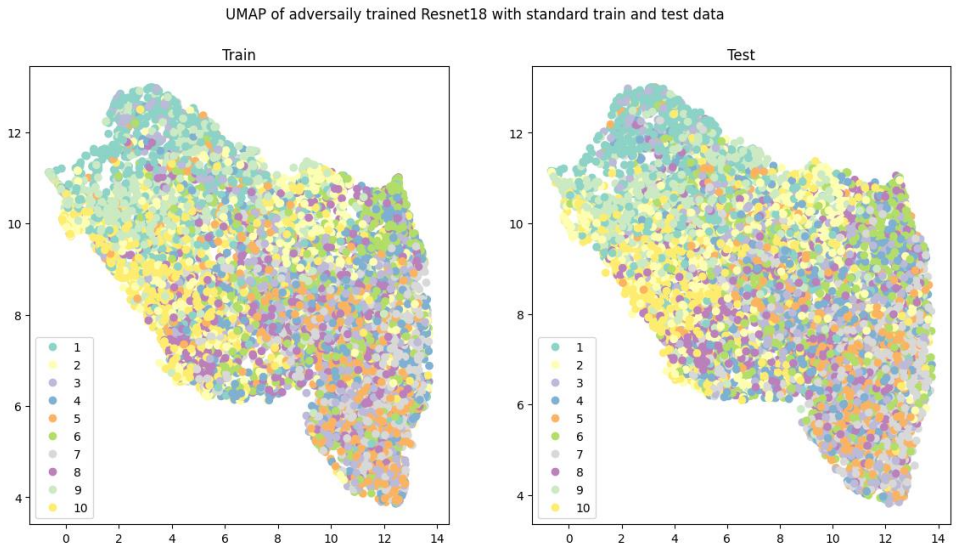
همانطور که در شکل ۸ مشاهده می‌کنید، مدل عملکرد نسبتا مشابهی رو داده‌های آموزش و تست دارد (همانطور که در شکل ۷ گزارش شد). بازنمایی تولید شده توسط این مدل نسبت به مدل آموزش یافته با داده‌های استاندارد، پراغتشاش و غیردقیق می‌باشد که باتوجه به نوع داده‌های آموزش در این دو مدل، قابل انتظار بود. از طرفی نکته قابل توجه در اینجا، نحوه عملکرد دو مدل روی داده‌های تست می‌باشد. همانطور که در شکل ۴ دیدیم، مدل آموزش یافته با داده‌های استاندارد منجر به بازنمایی بدتری از داده‌های تست نویزی می‌شد (بیشتر example ها را روی هم classify کرد).

حال می‌خواهیم عملکرد این مدل (Resnet18 آموزش یافته با داده‌های نویزی) را روی مجموعه train و test استاندارد دیتاست CIFAR10 بسنجیم. نتیجه‌ی آن را در شکل ۹ مشاهده می‌کنید.

```
Test Accuracy:  0.236
Train Accuracy 0.2374
```

شکل ۱۰- دقت مدل **Resnet18** آموزش یافته با داده‌های آموزش **adversary** روی داده‌های آموزش و تست استاندارد

طبق انتظار، دقت مدل adversary روی مجموعه‌داده‌ی استاندارد بدتر از دقت آن روی داده‌های نویزی بوده و به مراتب بدتر از دقت مدل استاندارد روی داده‌های استاندارد می‌باشد. نمایش بازنمایی‌های تولید شده توسط این مدل روی داده‌های آموزش و تست استاندارد با استفاده از UMAP را در شکل ۱۰ مشاهده می‌کنید.



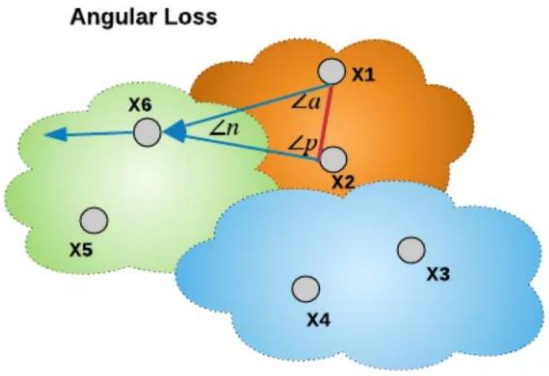
شکل ۱۱- نمایش **UMAP** بازنمایی حاصل از **Resnet18** آموزش یافته به طور **adversary** روی داده‌های آموزش نویزی و تست استاندارد

همانطور که در شکل ۸ مشاهده می‌کنید، مدل عملکرد نسبتا مشابهی رو داده‌های آموزش و تست دارد (همانطور که در شکل ۹ گزارش شد).

بخش ۵

بر خلاف شیوه‌های metric learning دیگر که مبتنی بر بهینه کردن فاصله مطلق (مثل contrastive loss) یا فاصله نسبی (Triplet loss, Lifted Structure Loss, N-Pair loss) هستند، Angular Loss پیشنهاد می‌کند یک رابطه مرتبه سه داخل مثلث triplet بر حسب زاویه در یال منفی رمزگذاری شود.

مشابه N-pair Loss، نسبت به تغییر مقیاس خنثی و مقاوم می‌باشد. این کار را با معرفی تابع هزینه‌ای که زاویه (کسینوس)را در نظر دارد، انجام می‌دهد. این روش سعی می‌کند تا بردارهای ویژگی منفی را از کلاستر مثبت دور کند و هم‌چنین نقاط مثبت را به یکدیگر نزدیک‌تر کند، همانطور که در تصویر زیر نشان داده شده است.



استفاده از این loss مزایایی دارد، از جمله:

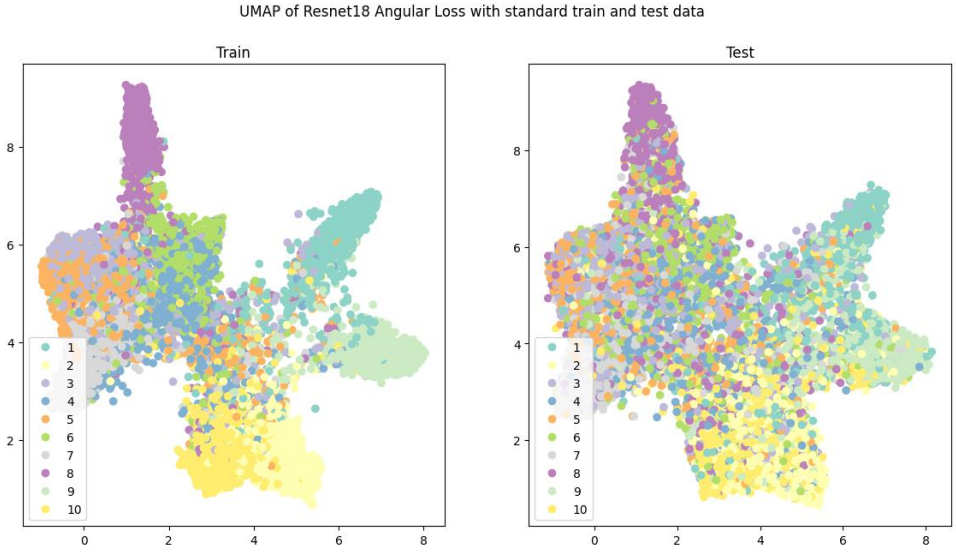
- متفاوت از فاصله اقلیدوسی، فاصله زاویه‌ای (کسینوسی) یک معیار بی‌نظیر برای similarity-transform-invariant است. در حالی که هندسه زاویه‌ای را در نظر می‌گیرد، این روش نه تنها از تغییرات مقیاسی بهره‌برداری می‌کند بلکه مقاومت به تغییرات دورانی را نیز معرفی می‌کند. با وجود اینکه ویژگی‌های تصویر به طور متداول هنگام آموزش تغییر مقیاس می‌یابند، اما برای یک حد ثابت زاویه α، همواره $n \leq \alpha$ صدق می‌کند. به عبارت ساده، دیدگاه هندسه زاویه‌ای در یک loss term، در برابر تغییرات محلی feature map، مقاوم‌تر است.
- قاعده کسینوس توضیح می‌دهد که محاسبه $\angle n$ به همه سه ضلع مثلث نیاز دارد. به عبارت دیگر، در مقابل، ترکیب اولیه تنها دو ضلع را در نظر می‌گیرد. افزودن محدودیت جدید، مقاومت و کارایی بهینه‌سازی را تشویق می‌کند.
- انتخاب loss margin، m، برای ترم خسارت وقتی که اقلیدوسی به عنوان معیار فاصله استفاده می‌شود، کاری آسان نیست. اصلی‌ترین دلیل این امر این است که با افزایش اندازه مجموعه داده، تغییرات داخل کلاسی بین کلاس‌های هدف بسیار متفاوت می‌شوند. بدون وجود یک

مرجع معنادار، تنظیم این پارامتر فوق‌پارامتر بسیار حیاتی است. تنظیم پارامتر α ساده‌تر است به دلیل رفتار scale-invariant آن.

- علاوه بر این، Angular loss به راحتی می‌تواند با توابع هزینه traditional یادگیری metric ترکیب شود تا کارایی کلی را بهبود بخشد.

بخش ۶

در این بخش ابتدا با استفاده از batch sampler توازن کلاس‌ها را در هر batch برقرار می‌کنیم. ابتدا مدل Resnet18 را با داده‌های آموزش استاندارد و تابع هزینه Angular Loss آموزش می‌دهیم و بردار ویژگی با اندازه ۱۲۸ بدست آمده از آن را توسط UMAP نمایش می‌دهیم. همانطور که در شکل ۱۱ مشاهده می‌کنید، بردار ویژگی بدست آمده در فضای دو بعدی نسبتا خوب توانسته بین کلاس‌ها تمایز ایجاد کند.



شکل ۱۲- نمایش **UMAP** بازنمایی حاصل از **Resnet18** آموزش یافته با **Angular Loss** روی داده‌های آموزش و تست استاندارد

حال این بردارهای ویژگی ۱۲۸تایی را به یک طبقه‌بند KNN با $k = 3$ می‌دهیم. دقت این مدل روی داده‌های تست و آموزش استاندارد در شکل ۱۲ گزارش شده‌است.

```
Accuracy on standard train:  0.816
Accuracy on standard test:  0.4065
```

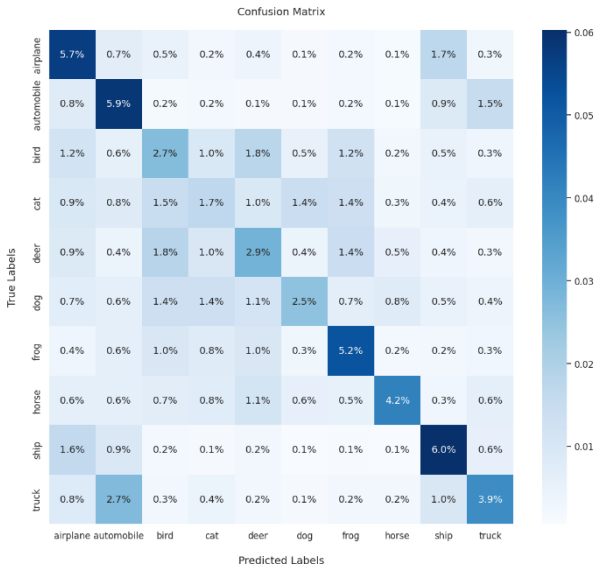
شکل ۱۳- دقت مدل **KNN** با بردار ویژگی بدست آمده از شبکه **Resnet18** آموزش یافته با داده‌های استاندارد **train** و **Angular Loss** روی داده‌های **test** و **train** استاندارد

با توجه به شکل ۱۳، طبقه‌بند KNN نسبتاً خوب توانسته classification انجام دهد. گزارش classification را در شکل ۱۴ و ماتریس آشفتگی آن را در شکل ۱۵ مشاهده می‌کنید. به طور کلی می‌توان گفت این مدل از مدل آموزش یافته در بخش ۲ (با Cross Entropy Loss) بدتر جواب می‌دهد.

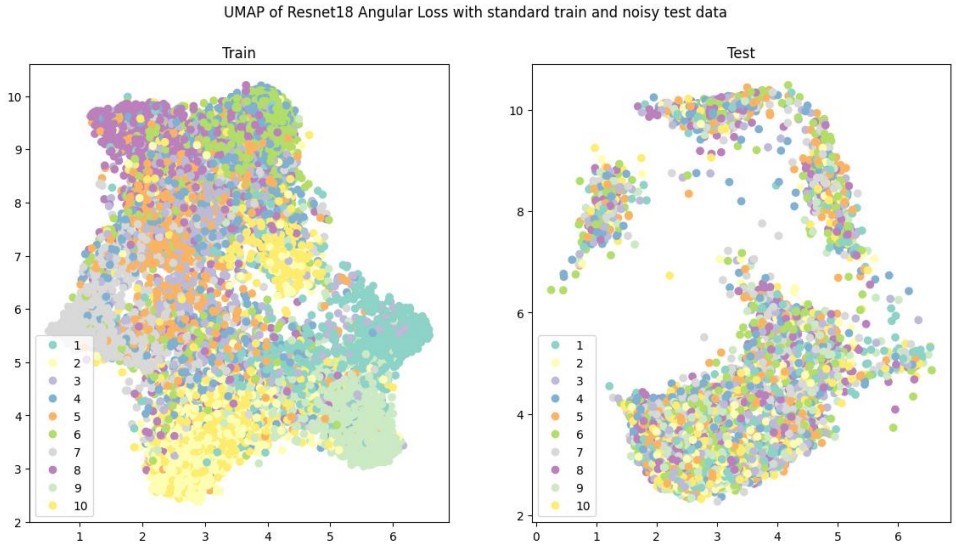
	precision	recall	f1-score	support
0	0.42	0.57	0.48	1000
1	0.42	0.59	0.49	1000
2	0.26	0.27	0.27	1000
3	0.22	0.17	0.19	1000
4	0.29	0.29	0.29	1000
5	0.40	0.25	0.31	1000
6	0.47	0.52	0.50	1000
7	0.61	0.42	0.50	1000
8	0.51	0.60	0.55	1000
9	0.45	0.39	0.42	1000
accuracy			0.41	10000
macro avg	0.41	0.41	0.40	10000
weighted avg	0.41	0.41	0.40	10000

شکل ۱۴ - گزارش **classification** مدل **KNN** روی داده‌های تست و آموزش استاندارد

حال می‌خواهیم عملکرد این مدل آموزش یافته با داده‌های استاندارد را روی داده‌های تست نویزی (adversary) ارزیابی کنیم. برای این‌کار، ابتدا بردارهای ویژگی بدست آمده توسط این مدل به ازای مجموعه داده‌های train استاندارد و مجموعه داده‌های تست adversary را با استفاده از UMAP به فضای دوبعدی می‌بریم که آن را در شکل ۱۶ مشاهده می‌کنید. همانطور که در شکل می‌بینید، مدل همانند بخش قبل داده‌های train را جدا می‌کند در حالیکه بردارهای ویژگی بدست آمده برای داده‌های تست منجر به بازنمایی discriminativeی نشدند.



شکل ۱۵- ماتریس آشفتگی مدل **KNN** روی داده‌های تست و آموزش استاندارد



شکل ۱۶- نمایش **UMAP** بازنمایی حاصل از **Resnet18** آموزش یافته با **Angular Loss** روی داده‌های آموزش استاندارد و تست نویزی

حال این بردارهای ویژگی ۱۲۸تایی را به یک طبقه‌بند KNN با $k = 3$ می‌دهیم. دقت این مدل روی داده‌های تست نویزی و آموزش استاندارد در شکل ۱۷ گزارش شده‌است.

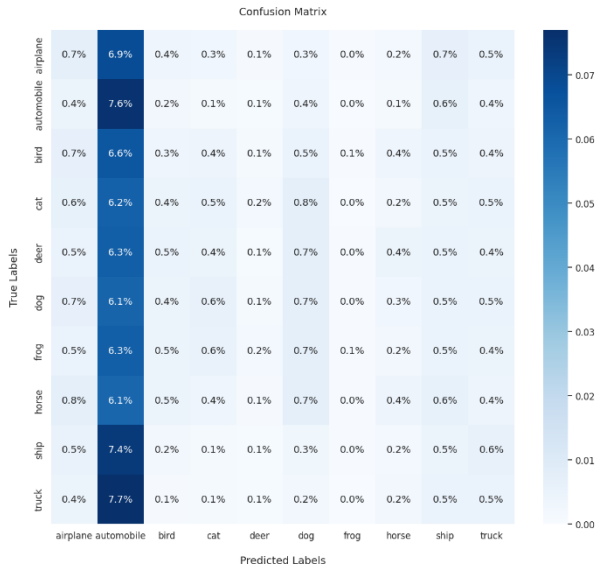
```
Accuracy on standard train:  0.6909
Accuracy on noisy test:  0.1139
```

شکل ۱۷- دقت مدل **KNN** با بردار ویژگی بدست آمده از شبکه **Resnet18** آموزش یافته با داده‌های استاندارد **train** و **Angular Loss** روی داده‌های **test** نویزی و **train** استاندارد

نتایج گزارش شده در شکل ۱۷ با نمودارهای شکل ۱۶ همخوانی دارد. گزارش classification را در شکل ۱۸ و ماتریس آشفتگی آن را در شکل ۱۹ مشاهده می‌کنید.

	precision	recall	f1-score	support
0	0.12	0.07	0.09	1000
1	0.11	0.76	0.20	1000
2	0.09	0.03	0.05	1000
3	0.14	0.05	0.07	1000
4	0.07	0.01	0.01	1000
5	0.13	0.07	0.09	1000
6	0.15	0.01	0.01	1000
7	0.15	0.04	0.06	1000
8	0.09	0.05	0.06	1000
9	0.11	0.05	0.07	1000
accuracy			0.11	10000
macro avg	0.12	0.11	0.07	10000
weighted avg	0.12	0.11	0.07	10000

شکل ۱۸- گزارش **classification** مدل **KNN** روی داده‌های تست نویزی و آموزش استاندارد



شکل ۱۹- ماتریس آشفتگی مدل **KNN** روی داده‌های تست نویزی و آموزش استاندارد

همانطور که در شکل ۱۹ می‌بینید، مدل آموزش یافته با داده‌های استاندارد، روی داده‌های تست نویزی خوب جواب نمی‌دهد.

در این بخش همانند سوال ۳، اغتشاشاتی روی داده‌های تست و train اعمال می‌کنیم. و این بردارهای ویژگی ۱۲۸تایی را به یک طبقه‌بند KNN با $k = 3$ می‌دهیم. دقت این مدل روی داده‌های تست نویزی و آموزش نویزی در شکل ۲۰ گزارش شده‌است.

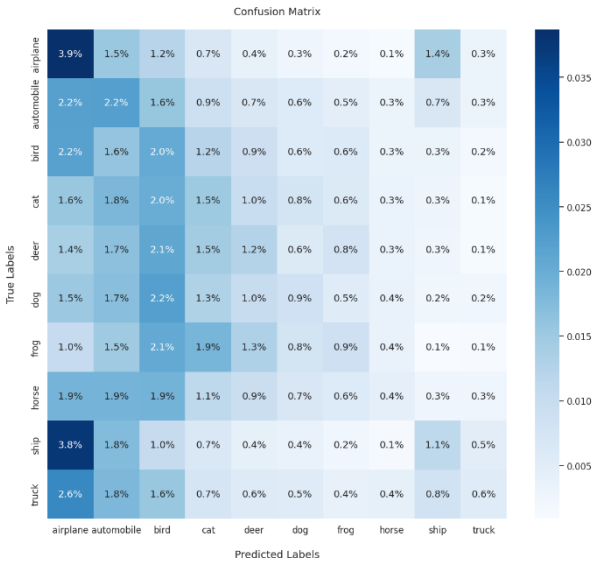
```
Accuracy on noisy train:  0.61
Accuracy on noisy test:  0.19
```

شکل ۲۰- دقت مدل **KNN** با بردار ویژگی بدست آمده از شبکه **Resnet18** آموزش یافته با داده‌های نویزی **train** و **Angular Loss** روی داده‌های **test** و **train** نویزی

گزارش classification را در شکل ۲۱ و ماتریس آشفتگی آن را در شکل ۲۲ مشاهده می‌کنید.

	precision	recall	f1-score	support
0	0.18	0.39	0.24	1000
1	0.13	0.22	0.16	1000
2	0.11	0.20	0.15	1000
3	0.13	0.15	0.14	1000
4	0.15	0.12	0.14	1000
5	0.14	0.09	0.11	1000
6	0.17	0.09	0.12	1000
7	0.14	0.04	0.07	1000
8	0.20	0.11	0.15	1000
9	0.20	0.06	0.09	1000
accuracy			0.15	10000
macro avg	0.16	0.15	0.13	10000
weighted avg	0.16	0.15	0.13	10000

شکل ۲۱- گزارش **classification** مدل **KNN** روی داده‌های تست و آموزش نویزی



شکل ۲۲- ماتریس آشفتگی مدل **KNN** روی داده‌های تست و آموزش نویزی

همانطور که در نتایج این مدل می‌بینید، این مدل آموزش یافته با داده‌های نویزی روی داده‌های unseen نویزی تست بهتر جواب می‌دهد (نسبت به مدل آموزش یافته با داده‌های آموزش استاندارد).

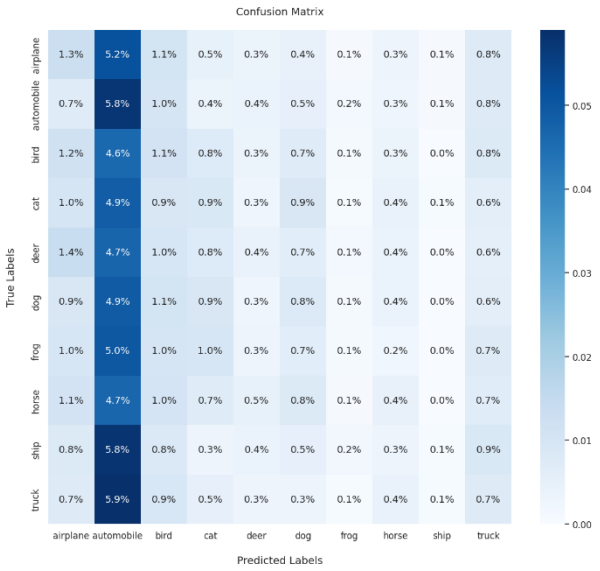
حال با استفاده از روش Fast Gradient یک adversary attack طراحی کردیم (Epsilon = 0.2) و داده‌های حاصل از این حمله را به مدل دادیم. و سپس بازنمایی‌های تولید شده توسط این مدل را به یک طبقه‌بند KNN با k = 3 می‌دهیم. دقت این مدل روی داده‌های تست نویزی و آموزش نویزی و همچنین گزارش classification را در شکل ۲۳ مشاهده می‌کنید.

Accuracy on noisy test: 0.116

	precision	recall	f1-score	support
0	0.13	0.13	0.13	1000
1	0.11	0.58	0.19	1000
2	0.11	0.11	0.11	1000
3	0.13	0.09	0.11	1000
4	0.12	0.04	0.06	1000
5	0.13	0.08	0.10	1000
6	0.10	0.01	0.02	1000
7	0.13	0.04	0.07	1000
8	0.14	0.01	0.01	1000
9	0.10	0.07	0.09	1000
accuracy			0.12	10000
macro avg	0.12	0.12	0.09	10000
weighted avg	0.12	0.12	0.09	10000

شکل ۲۳- گزارش **classification** مدل **KNN** روی داده‌های تست و آموزش **adversary**

و ماتریس آشفتگی آن را در شکل ۲۴ مشاهده می‌کنید.



شکل ۲۴- ماتریس آشفتگی مدل **KNN** روی داده‌های تست و آموزش **adversary**

همانطور که در شکل ۲۴ مشاهده می‌کنید با اعمال حمله adversary به مدل، مدل بیشتر نمونه‌ها را در کلای ماشین طبقه‌بندی می‌کند.

جمع‌بندی نهایی: زمانی که داده‌ها noisy هستند تابع Angular Loss نسبت به Cross Entropy بهتر جواب می‌دهد.