«*In The Name Of GOD*»

[HW-04-Report]

[NEURAL COMPUTING AND DEEP LEARNING]

Hasan Masroor | [403131030] | May 12, 2025

"فهرست مطالب تمرين 04"

Problem 4: convolutional neural networks (CNN)

Question 1:

1.

هدف این تمرین بررسی تأثیر تنظیمات مختلف شبکه‌های کانولوشنی در طبقه‌بندی تصاویر CIFAR-10 است. ابتدا یک مدل پایه با سه لایه کانولوشن می‌سازیم، سپس تأثیر عوامل مختلف مثل عمق شبکه، Dropout، Batch Normalization و Data Augmentation را بررسی می‌کنیم. همچنین از مدل ResNet-18 برای مقایسه استفاده کرده و روش‌های مختلف تنظیم نرخ یادگیری را آزمایش می‌کنیم. در پایان با Grad-CAM تحلیل می‌کنیم که مدل روی چه بخش‌هایی از تصویر تمرکز می‌کند. نتایج هر بخش را با معیارهای دقت، recall و F1-score ارزیابی می‌کنیم.

ابتدا کتابخانه‌های مورد نیاز را import کردیم و سپس دیتاست گفته شده یعنی CIFAR-10 را لود کردیم. دیتاست شامل دو زیر بخش تست و آموزش می‌باشد که به بخش تست دست نمی‌زنیم و بخش آموزش را به دو بخش آموزش و اعتبارسنجی تقسیم می‌کنیم و اندازه این سه مجموعه به شرح زیر است:

```
Training set size: 43750 samples
Validation set size: 6250 samples
Test set size: 10000 samples
```

سپس 10 کلاس این دیتاست را برای درک بهتر در خروجی نمای می‌دهیم:

```
Classes in CIFAR-10 dataset: ['airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck']
```

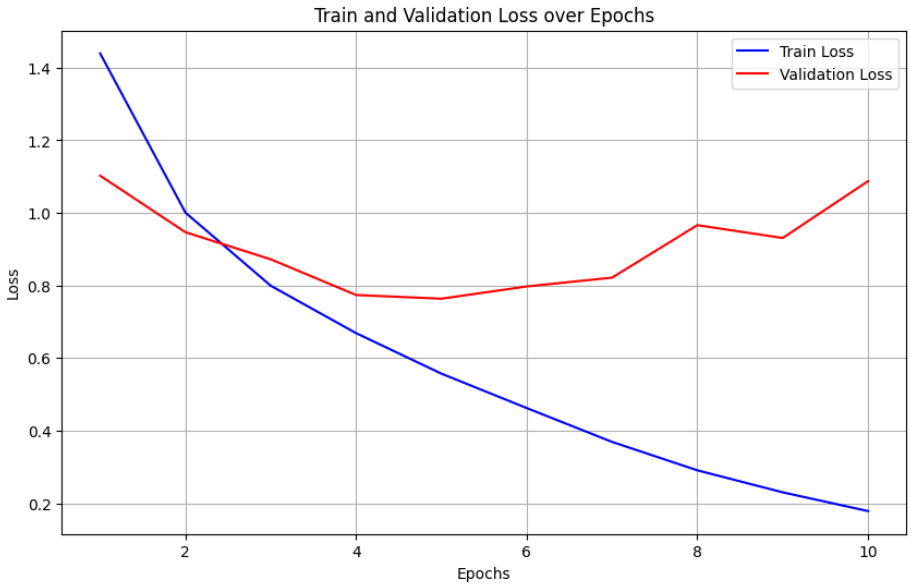
برای سوال اول این تمرین یک مدل شبکه عصبی کانولوشنی ساده برای طبقه‌بندی تصاویر CIFAR-10 طراحی و آموزش دادیم. مدل شامل سه لایه کانولوشن همراه با لایه‌های Pooling و در انتها، دو لایه تمام متصل برای طبقه‌بندی تصاویر در ۱۰ کلاس مختلف می‌باشد. این مدل ساده به عنوان پایه برای مقایسه با تغییرات بعدی در معماری شبکه استفاده خواهد شد.

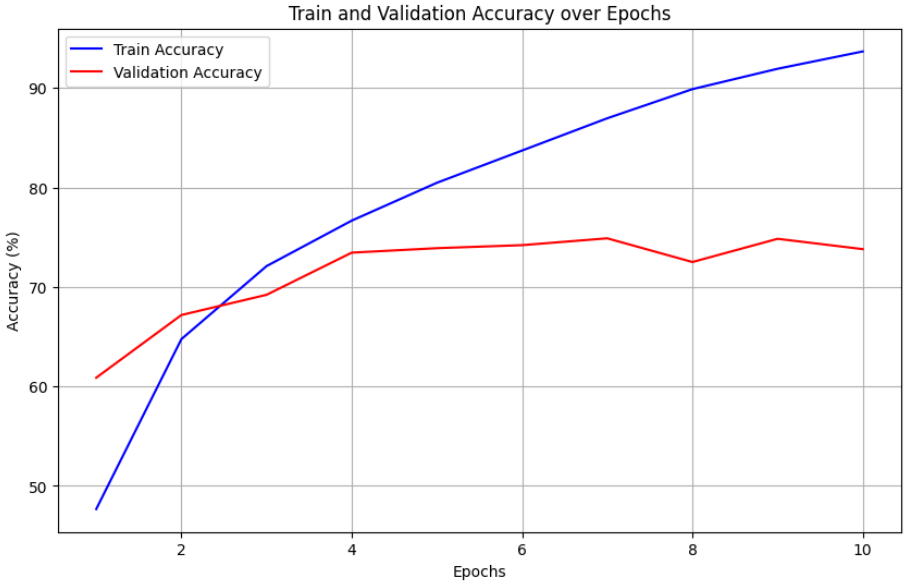
مدل CNN را با استفاده از Adam optimizer و نرخ یادگیری 0.001 آموزش دادیم. در هر epoch، مدل روی داده‌های آموزش به روزرسانی شد و سپس روی داده‌های اعتبارسنجی ارزیابی گردید. این فرآیند برای 10 اپک تکرار شد و در هر مرحله loss و accuracy آموزش و اعتبارسنجی ثبت گردید. در نهایت، مدل روی داده‌های تست ارزیابی شد و معیارهای accuracy، precision، recall و F1-score محاسبه و گزارش شدند و خروجی این قسمت به صورت زیر در آمده است:

```
Epoch [1/10], Train Loss: 1.4393, Validation Loss: 1.1024, Train Accuracy: 47.67%
Epoch [2/10], Train Loss: 1.0006, Validation Loss: 0.9467, Train Accuracy: 64.76%
Epoch [3/10], Train Loss: 0.7992, Validation Loss: 0.8720, Train Accuracy: 72.10%
Epoch [4/10], Train Loss: 0.6686, Validation Loss: 0.7737, Train Accuracy: 76.68%
Epoch [5/10], Train Loss: 0.5570, Validation Loss: 0.7636, Train Accuracy: 80.48%
Epoch [6/10], Train Loss: 0.4622, Validation Loss: 0.7973, Train Accuracy: 83.72%
Epoch [7/10], Train Loss: 0.3690, Validation Loss: 0.8216, Train Accuracy: 86.97%
Epoch [8/10], Train Loss: 0.2907, Validation Loss: 0.9661, Train Accuracy: 89.89%
Epoch [9/10], Train Loss: 0.2301, Validation Loss: 0.9308, Train Accuracy: 91.95%
Epoch [10/10], Train Loss: 0.1784, Validation Loss: 1.0873, Train Accuracy: 93.69%

Final Test Accuracy: 73.79%
Precision: 0.7429
Recall: 0.7379
F1-Score: 0.7389
```

تحلیل نتایج آموزش مدل نشان‌دهنده روند یادگیری مناسب در ابتدای فرآیند است، با کاهش پیوسته loss آموزش و افزایش accuracy که از ٪۴۷ به ٪۹۳ رسیده. با این حال، از epoch هفتم به بعد، افزایش validation loss و فاصله گرفتن آن از train loss نشان‌دهنده شروع overfitting است. دقت نهایی مدل روی داده تست ٪۷۳٫۷۹ با مقادیر precision و recall نزدیک به هم است که نشان‌دهنده عملکرد متعادل مدل در تشخیص تمام کلاس‌هاست. برای بهبود نتایج می‌توان از تکنیک‌های regularization مانند dropout یا augmentation داده استفاده کرد که در بخش‌های بعد این کارها را انجام می‌دهیم.

سپس نمودارهای مربوط به Loss و دقت را برای آموزش و اعتبارسنجی در زیر نمایش دادیم و به خوبی می‌توانیم توضیحات بالا را به صورت بصری ببینیم و مدل از یک جایی به بعد دچار اورفیت شده است :





.2

در این بخش می‌خواهیم تاثیر عمق شبکه را بررسی کنیم و برای این کار ما شبکه ۲ لایه، ٤ لایه و ٥ لایه را انتخاب کردیم. در این بخش یک مدل کانولوشنی با دو لایه پیچشی طراحی کردیم. هر لایه کانولوشن به همراه لایه پولینگ برای کاهش ابعاد فضایی استفاده شده است. در انتها دو لایه تمام متصل برای طبقه‌بندی تصاویر در ۱۰ کلاس مختلف اضافه شده‌اند. این مدل ساده‌تر نسبت به نسخه سه لایه‌ای، به عنوان یکی از معماری‌های مورد بررسی برای مقایسه عملکرد شبکه‌های با عمق مختلف استفاده خواهد شد. در ادامه کانولوشنی عمیق‌تر با چهار لایه پیچشی طراحی کردیم، هر لایه کانولوشن همراه با یک لایه پولینگ برای کاهش تدریجی ابعاد تصویر استفاده شده است. پس از استخراج ویژگی‌ها، دو لایه تمام متصل برای انجام طبقه‌بندی نهایی روی ۱۰ کلاس مجموعه داده CIFAR-10 اضافه شده‌اند. این معماری نسبت به مدل‌های کم‌عمق‌تر قبلی، قابلیت بیشتری برای یادگیری ویژگی‌های پیچیده دارد. مدل سومی یک شبکه کانولوشنی عمیق با پنج لایه پیچشی است و هر لایه کانولوشن به دنبال یک لایه پولینگ قرار گرفته که ابعاد تصویر را به تدریج کاهش می‌دهد. در انتها، دو لایه تمام متصل ویژگی‌های استخراج شده را به ۱۰ کلاس خروجی نگاشت می‌کنند. این معماری نسبت به مدل‌های کم‌عمق‌تر قبلی، ظرفیت یادگیری بیشتری دارد اما ممکن است با چالش‌هایی مانند مشکل vanishing gradients مواجه شود که نیاز به بررسی دارد.

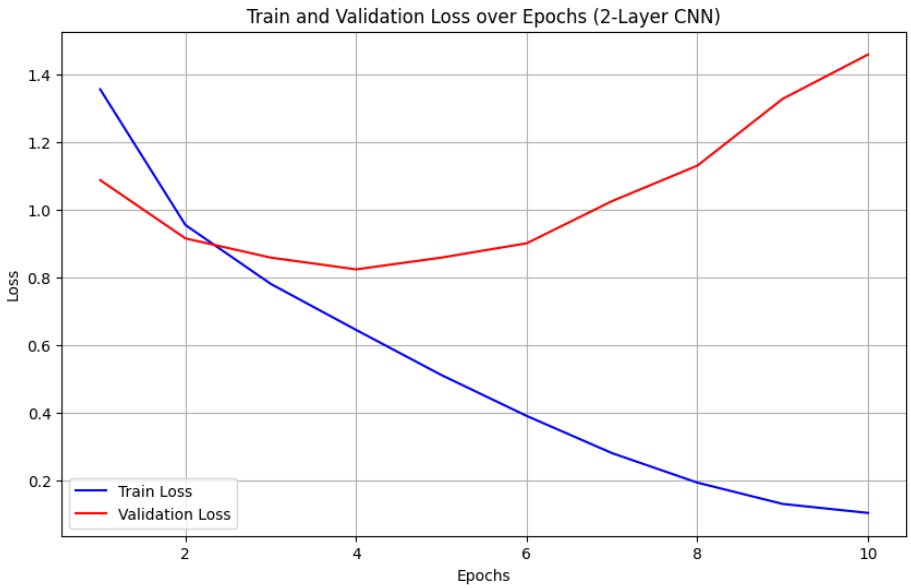
بعد از اینکه کلاس این سه مدل را تعریف کردیم مثل مرحله قبل مدل را آموزش می‌دهیم و معیارهای ارزیابی و نمودارهای مربوطه را برای تجزیه و تحلیل بهتر نمایش دادیم و در ادامه بررسی می‌کنیم:

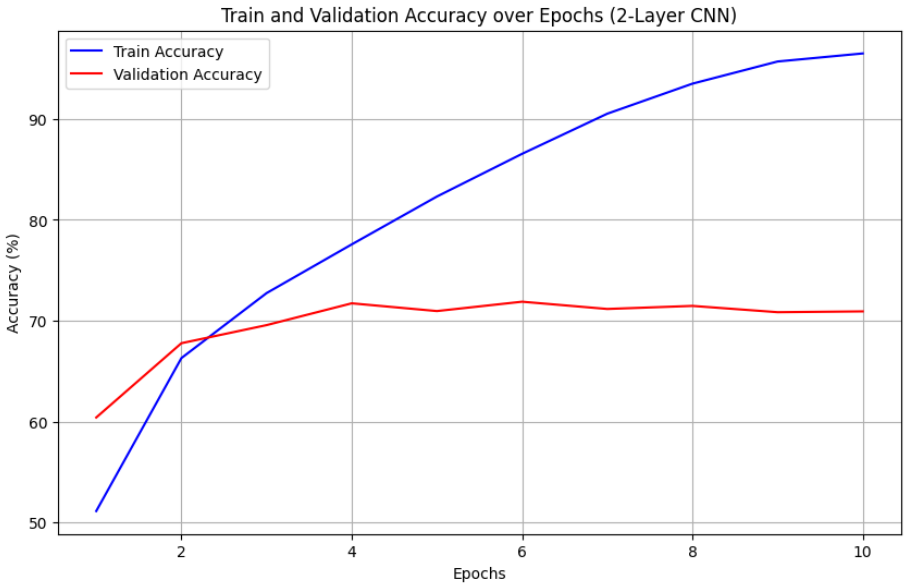
- **2-Layer CNN:**

```
Epoch [1/10], Train Loss: 1.3566, Validation Loss: 1.0878, Train Accuracy: 51.15%, Avg First Layer Gradient: 0.031370
Epoch [2/10], Train Loss: 0.9548, Validation Loss: 0.9152, Train Accuracy: 66.30%, Avg First Layer Gradient: 0.043559
Epoch [3/10], Train Loss: 0.7806, Validation Loss: 0.8584, Train Accuracy: 72.74%, Avg First Layer Gradient: 0.047274
Epoch [4/10], Train Loss: 0.6444, Validation Loss: 0.8236, Train Accuracy: 77.57%, Avg First Layer Gradient: 0.047704
Epoch [5/10], Train Loss: 0.5108, Validation Loss: 0.8584, Train Accuracy: 82.31%, Avg First Layer Gradient: 0.046608
Epoch [6/10], Train Loss: 0.3898, Validation Loss: 0.9009, Train Accuracy: 86.54%, Avg First Layer Gradient: 0.044675
Epoch [7/10], Train Loss: 0.2797, Validation Loss: 1.0256, Train Accuracy: 90.51%, Avg First Layer Gradient: 0.042233
Epoch [8/10], Train Loss: 0.1924, Validation Loss: 1.1307, Train Accuracy: 93.49%, Avg First Layer Gradient: 0.038026
Epoch [9/10], Train Loss: 0.1293, Validation Loss: 1.3283, Train Accuracy: 95.69%, Avg First Layer Gradient: 0.034098
Epoch [10/10], Train Loss: 0.1027, Validation Loss: 1.4595, Train Accuracy: 96.48%, Avg First Layer Gradient: 0.034089

Final Test Accuracy: 70.74%
Precision: 0.7105
Recall: 0.7074
F1-Score: 0.7058
```

نتایج آموزش مدل دو لایه‌ای نشان‌دهنده یادگیری اولیه مناسب است، با افزایش دقت آموزش از ٪۵۱ به ٪۹۶ و میانگین گرادیان لایه اول که در محدوده معقول (۰/۰۳ تا ۰/۰۴) باقی می‌ماند. با این حال، از epoch پنجم به بعد، افزایش تدریجی loss اعتبارسنجی درحالی که loss آموزش کاهش می‌یابد، نشانه واضح overfitting است. دقت نهایی ٪۷۰/۷۴ روی داده تست با مقادیر precision و recall نزدیک به هم عملکرد متعادلی را نشان می‌دهد. مقادیر گرادیان که در طول آموزش نسبتاً پایدار مانده‌اند، حاکی از عدم مواجهه با مشکل vanishing gradient در این معماری کم‌عمق است.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که بعد از چند اپک دچار اورفیت می‌شود و دقت آموزش به طور قابل توجهی زیاد شده است اما اعتبارسنجی بعد از چند اپک حدودا دقت ثابتی دارد و زیاد افزایش نیافته است:



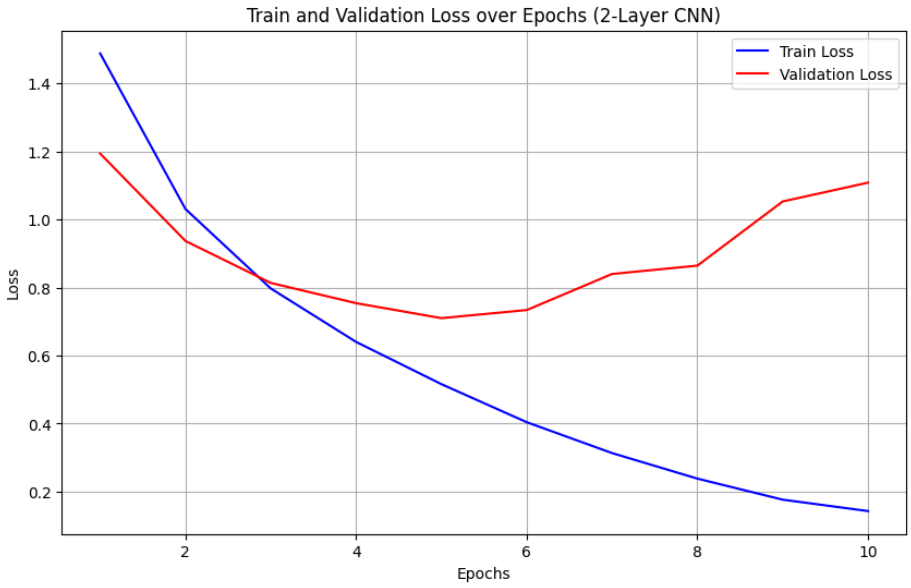


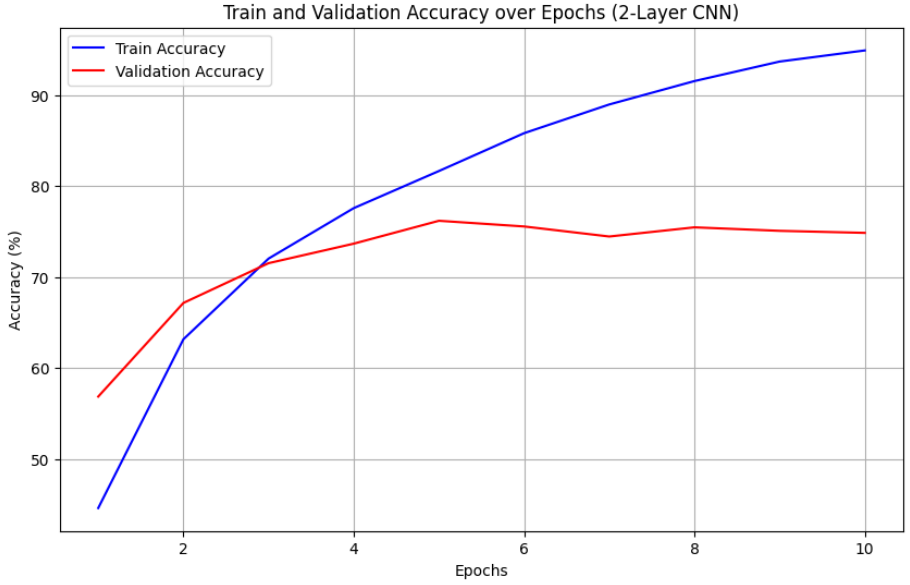
- **4-Layer CNN:**

```
Epoch [1/10], Train Loss: 1.4880, Validation Loss: 1.1937, Train Accuracy: 44.64%, Avg First Layer Gradient: 0.037353
Epoch [2/10], Train Loss: 1.0306, Validation Loss: 0.9368, Train Accuracy: 63.18%, Avg First Layer Gradient: 0.052524
Epoch [3/10], Train Loss: 0.7972, Validation Loss: 0.8135, Train Accuracy: 72.04%, Avg First Layer Gradient: 0.055204
Epoch [4/10], Train Loss: 0.6401, Validation Loss: 0.7540, Train Accuracy: 77.58%, Avg First Layer Gradient: 0.057983
Epoch [5/10], Train Loss: 0.5160, Validation Loss: 0.7100, Train Accuracy: 81.66%, Avg First Layer Gradient: 0.060815
Epoch [6/10], Train Loss: 0.4042, Validation Loss: 0.7339, Train Accuracy: 85.81%, Avg First Layer Gradient: 0.060235
Epoch [7/10], Train Loss: 0.3135, Validation Loss: 0.8398, Train Accuracy: 88.97%, Avg First Layer Gradient: 0.061238
Epoch [8/10], Train Loss: 0.2384, Validation Loss: 0.8644, Train Accuracy: 91.54%, Avg First Layer Gradient: 0.060431
Epoch [9/10], Train Loss: 0.1765, Validation Loss: 1.0530, Train Accuracy: 93.68%, Avg First Layer Gradient: 0.059787
Epoch [10/10], Train Loss: 0.1430, Validation Loss: 1.1082, Train Accuracy: 94.90%, Avg First Layer Gradient: 0.059653

Final Test Accuracy: 74.22%
Precision: 0.7494
Recall: 0.7422
F1-Score: 0.7424
```

نتایج مدل ٤ لایه‌ای نشان می‌دهد با وجود عمق بیشتر شبکه، میانگین گرادیان‌های لایه اول در محدوده سالمی (۰/۰۳۷ تا ۰/۰۶۱) باقی مانده که نشان‌دهنده عدم بروز مشکل vanishing gradient است. مدل توانسته به دقت تست ۷۴/۲۲٪ برسد که نسبت به مدل ۲ لایه‌ای بهبود حدود ۴٪ را نشان می‌دهد. با این حال، از epoch ششم به بعد افزایش loss اعتبارسنجی در حالی که loss آموزش کاهش می‌یابد نشان‌دهنده overfitting است. این معماری عمیق‌تر نسبت به مدل ۲ لایه‌ای عملکرد بهتری نشان داده.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که بعد از چند اپک دچار اورفیت می‌شود و دقت آموزش به طور قابل توجهی زیاد شده است اما اعتبارسنجی بعد از چند اپک حدودا دقت ثابتی دارد و زیاد افزایش نیافته است اما نسبت به مدل ۲ لایه عمکرد بهتری داشته است و هم دقت نهایی بیشتری داشتیم و هم دقت مجموعه اعتبارسنجی بالاتر رفته است:



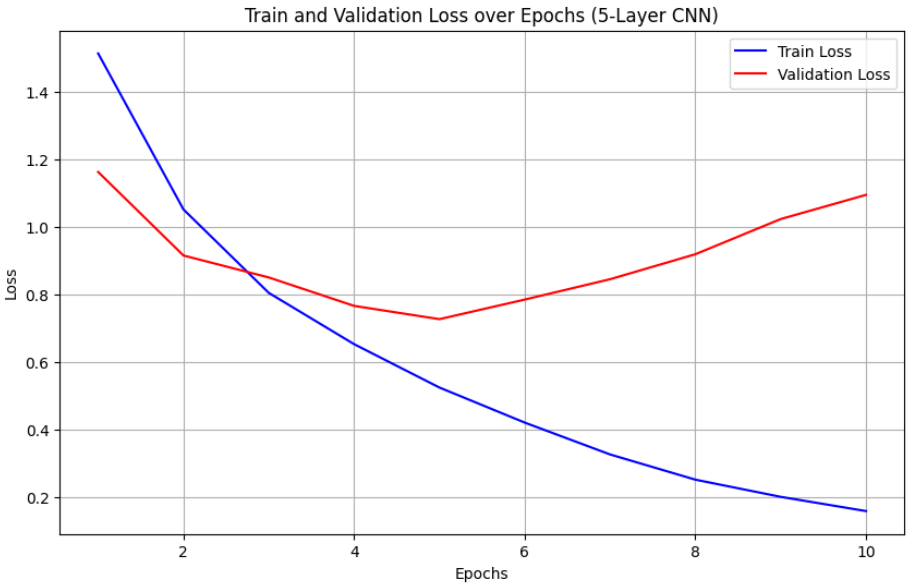


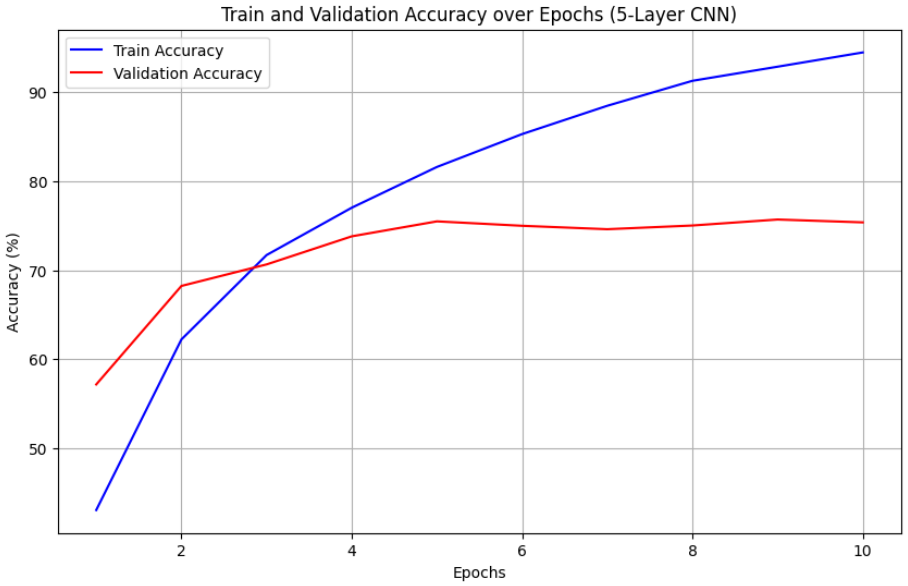
- **5-Layer CNN:**

```
Epoch [1/10], Train Loss: 1.5135, Validation Loss: 1.1627, Train Accuracy: 43.07%, Avg First Layer Gradient: 0.040763
Epoch [2/10], Train Loss: 1.0519, Validation Loss: 0.9156, Train Accuracy: 62.22%, Avg First Layer Gradient: 0.054547
Epoch [3/10], Train Loss: 0.8049, Validation Loss: 0.8506, Train Accuracy: 71.69%, Avg First Layer Gradient: 0.061824
Epoch [4/10], Train Loss: 0.6532, Validation Loss: 0.7665, Train Accuracy: 77.01%, Avg First Layer Gradient: 0.065660
Epoch [5/10], Train Loss: 0.5249, Validation Loss: 0.7272, Train Accuracy: 81.58%, Avg First Layer Gradient: 0.065988
Epoch [6/10], Train Loss: 0.4211, Validation Loss: 0.7852, Train Accuracy: 85.27%, Avg First Layer Gradient: 0.068639
Epoch [7/10], Train Loss: 0.3269, Validation Loss: 0.8454, Train Accuracy: 88.45%, Avg First Layer Gradient: 0.068308
Epoch [8/10], Train Loss: 0.2523, Validation Loss: 0.9193, Train Accuracy: 91.25%, Avg First Layer Gradient: 0.069281
Epoch [9/10], Train Loss: 0.2014, Validation Loss: 1.0234, Train Accuracy: 92.83%, Avg First Layer Gradient: 0.071283
Epoch [10/10], Train Loss: 0.1593, Validation Loss: 1.0950, Train Accuracy: 94.43%, Avg First Layer Gradient: 0.067852

Final Test Accuracy: 75.53%
Precision: 0.7582
Recall: 0.7553
F1-Score: 0.7558
```

نتایج مدل ۵ لایه‌ای نشان‌دهنده بهبود عملکرد نسبت به مدل‌های کم‌عمق‌تر است، با دستیابی به دقت تست ۷۵,۵۳% که بالاترین مقدار در بین مدل‌های بررسی شده می‌باشد. میانگین گرادیان‌های لایه اول در محدوده ۰,۰٤ تا ۰,۰۷ در نوسان بوده که نشان‌دهنده جریان مناسب گرادیان‌ها در شبکه و عدم مواجهه با مشکل vanishing gradientاست. الگوی یادگیری نشان می‌دهد که مدل از epoch پنجم به بعد دچار overfitting شده، به طوری که loss آموزش به کاهش ادامه داده در حالی که loss اعتبارسنجی افزایش یافته است. مقادیر precision (0.7582)، recall (0.7553)و F1-score (0.7558) بسیار نزدیک به هم هستند که نشانگر عملکرد متعادل مدل در تشخیص تمام کلاس‌ها می‌باشد. این نتایج حاکی از آن است که افزایش عمق شبکه تا ۵ لایه در این معماری خاص منجر به بهبود عملکرد شده، هرچند که مسئله overfitting محسوس تر شده است.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که بعد از چند اپک دچار اورفیت می‌شود و دقت آموزش به طور قابل توجهی زیاد شده است اما اعتبارسنجی بعد از چند اپک حدودا دقت ثابتی دارد و زیاد افزایش نیافته است اما نسبت به مدل ۲ لایه عمکرد بهتری داشته است و هم دقت نهایی بیشتری داشتیم و هم دقت مجموعه اعتبارسنجی بالاتر رفته است:





.3

در این بخش باید تأثیر دو تکنیک مهم منظم‌سازی Dropout و Weight Decay را روی مدل پایه بررسی کنیم تا ببینیم چطور می‌توانند از overfitting جلوگیری کرده و فاصله بین دقت آموزش و اعتبارسنجی را کاهش دهند. به دلیل محدودیت‌های سخت‌افزاری و زمانی، به جای انجام گرید سرچ گسترده، با طراحی هوشمندانه ۴ آزمایش

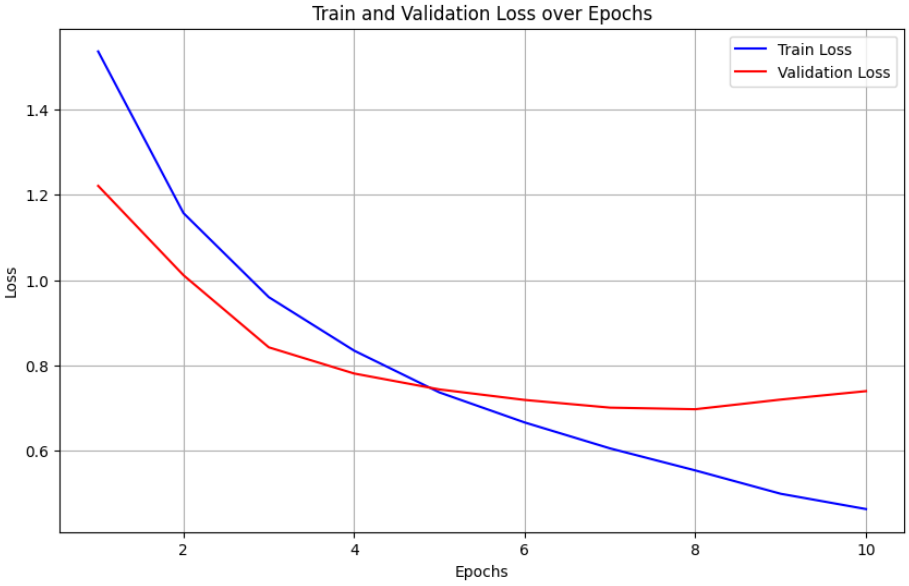
مختلف، تأثیر تغییرات این پارامترها را تحلیل کردیم. در هر آزمایش یکی از پارامترها را تغییر دادیم در حالی که دیگری ثابت بود تا بتوانیم تأثیر مستقل هر کدام را به وضوح مشاهده کنیم. این رویکرد به ما امکان داد تا بفهمیم افزایش یا کاهش هر پارامتر چگونه بر عملکرد مدل تأثیر می‌گذارد و چطور می‌توانیم به تعادل مناسبی بین این دو تکنیک برسیم تا مدل بدون overfitting بهترین عملکرد را داشته باشد. در ادامه مثل قبل مدل‌ها را آموزش دادیم و نتایج خروجی و نمودارها برای هر مقداردهی به صورت زیر حاصل شده است:

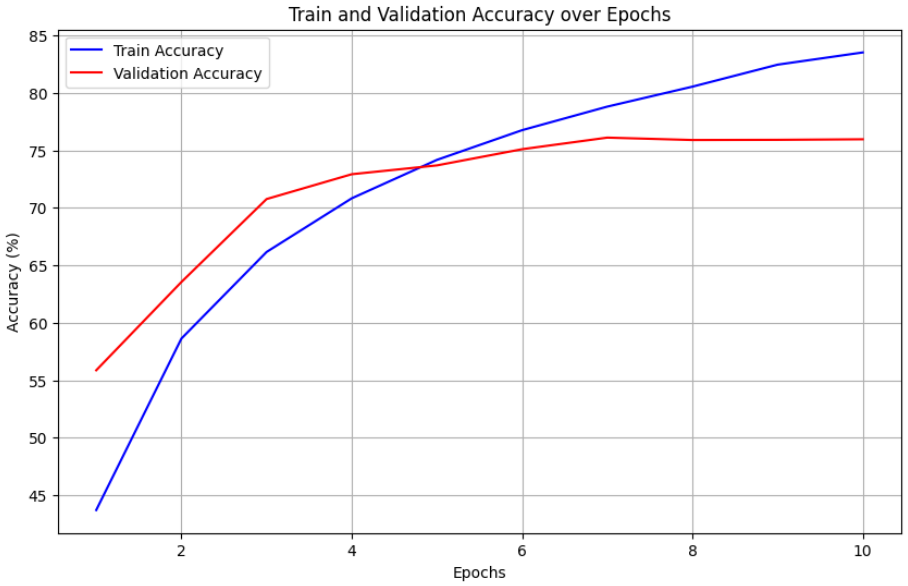
- **Dropout = 0.5 , weight_decay = 0.0001**

```
Epoch [1/10], Train Loss: 1.5350, Validation Loss: 1.2204, Train Accuracy: 43.71%
Epoch [2/10], Train Loss: 1.1566, Validation Loss: 1.0112, Train Accuracy: 58.63%
Epoch [3/10], Train Loss: 0.9599, Validation Loss: 0.8425, Train Accuracy: 66.16%
Epoch [4/10], Train Loss: 0.8348, Validation Loss: 0.7812, Train Accuracy: 70.83%
Epoch [5/10], Train Loss: 0.7370, Validation Loss: 0.7440, Train Accuracy: 74.18%
Epoch [6/10], Train Loss: 0.6666, Validation Loss: 0.7194, Train Accuracy: 76.76%
Epoch [7/10], Train Loss: 0.6060, Validation Loss: 0.7014, Train Accuracy: 78.81%
Epoch [8/10], Train Loss: 0.5544, Validation Loss: 0.6977, Train Accuracy: 80.55%
Epoch [9/10], Train Loss: 0.4998, Validation Loss: 0.7202, Train Accuracy: 82.46%
Epoch [10/10], Train Loss: 0.4640, Validation Loss: 0.7400, Train Accuracy: 83.52%

Final Test Accuracy: 76.67%
Precision: 0.7739
Recall: 0.7667
F1-Score: 0.7684
```

نتایج این آزمایش با Dropout=0.5 و Weight Decay=0.0001 نشان‌دهنده عملکرد متعادل مدل است. در طول آموزش مشاهده می‌شود که loss آموزش و اعتبارسنجی به طور پیوسته و هماهنگ کاهش یافته‌اند، به طوری که در epoch هشتم به کمترین اختلاف (۰/۵۵۴۴ در برابر ۰/۶۹۷۷) رسیده‌اند. این نشان‌دهنده اثر بخشی ترکیب این دو تکنیک در کنترل overfitting است. دقت نهایی تست ٪۷۶/۶۷ که نسبت به مدل پایه (بدون این تنظیمات) حدود ٪۳-۲ بهبود نشان می‌دهد. مقادیر precision (0.7739)، recall (0.7667) و F1-score (0.7684) که بسیار نزدیک به هم هستند، حاکی از عملکرد متعادل مدل در تشخیص تمام کلاس‌هاست. نکته جالب توجه این است که با وجود Dropout نسبتاً بالا (۰/۵)، مدل توانسته به دقت آموزش مناسبی (٪۸۳/۵۲) دست یابد که نشان‌دهنده این است که Weight Decay به خوبی از overfitting جلوگیری کرده است. این ترکیب پارامترها به نظر می‌رسد تعادل مناسبی بین یادگیری و منظم‌سازی ایجاد کرده باشد.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که مدل از اپک ۵ به بعد مجدد دچار اورفیت شده است اما دقت آنها در آموزش و اعتبارسنجی کمتر نست به حالات قبلی شده است:


Train and Validation Loss over Epochs
Train Loss
Validation Loss
Loss
Epochs
1.4
1.2
1.0
0.8
0.6
2
4
6
8
10

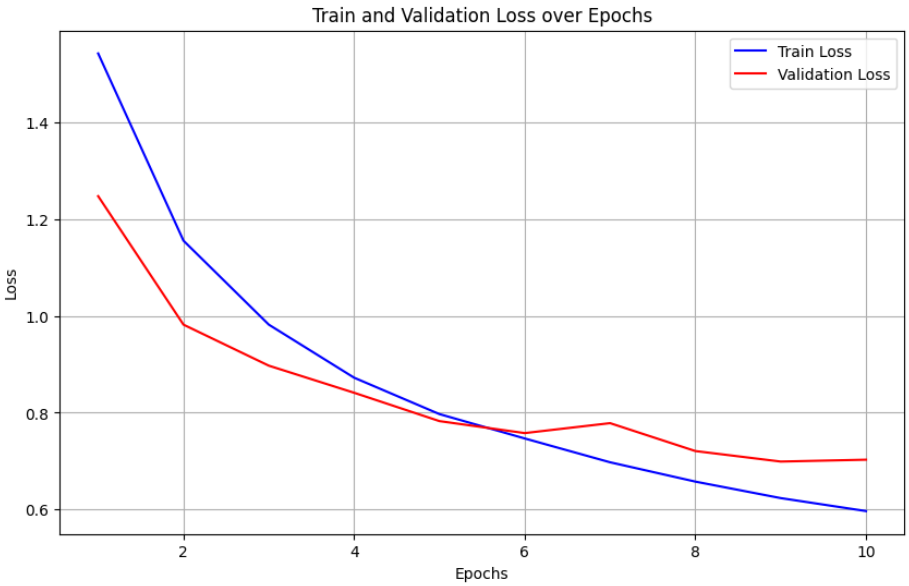

Train and Validation Accuracy over Epochs
Train Accuracy
Validation Accuracy
Accuracy (%)
Epochs
85
80
75
70
65
60
55
50
45
2
4
6
8
10

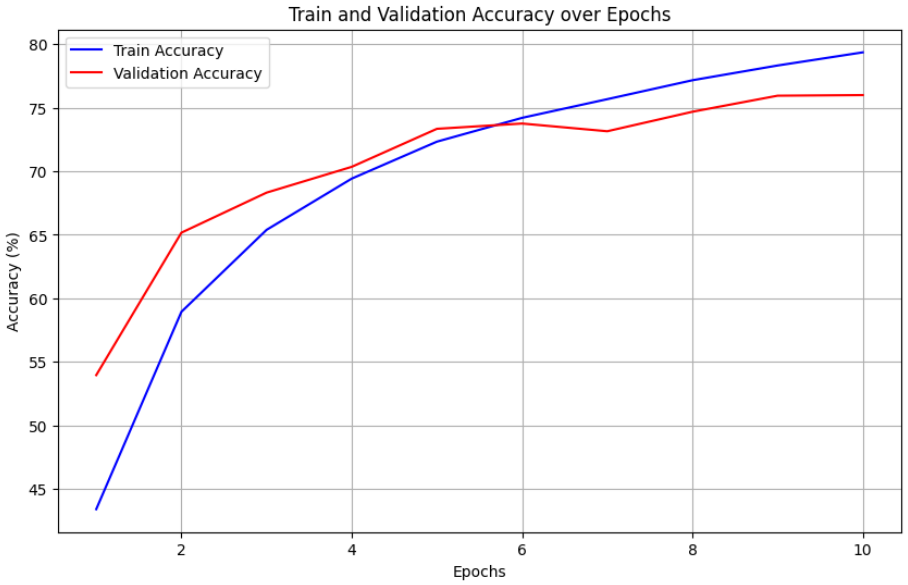
- **Dropout = 0.4 , weight_decay = 0.001**

```
Epoch [1/10], Train Loss: 1.5418, Validation Loss: 1.2471, Train Accuracy: 43.40%
Epoch [2/10], Train Loss: 1.1551, Validation Loss: 0.9817, Train Accuracy: 58.94%
Epoch [3/10], Train Loss: 0.9818, Validation Loss: 0.8970, Train Accuracy: 65.39%
Epoch [4/10], Train Loss: 0.8722, Validation Loss: 0.8411, Train Accuracy: 69.42%
Epoch [5/10], Train Loss: 0.7970, Validation Loss: 0.7825, Train Accuracy: 72.33%
Epoch [6/10], Train Loss: 0.7467, Validation Loss: 0.7578, Train Accuracy: 74.21%
Epoch [7/10], Train Loss: 0.6975, Validation Loss: 0.7783, Train Accuracy: 75.68%
Epoch [8/10], Train Loss: 0.6576, Validation Loss: 0.7207, Train Accuracy: 77.17%
Epoch [9/10], Train Loss: 0.6235, Validation Loss: 0.6990, Train Accuracy: 78.33%
Epoch [10/10], Train Loss: 0.5968, Validation Loss: 0.7028, Train Accuracy: 79.36%

Final Test Accuracy: 76.61%
Precision: 0.7657
Recall: 0.7661
F1-Score: 0.7627
```

نتایج مدل با Dropout=0.4 و Weight Decay=0.001 نشان‌دهنده عملکرد متعادل و کنترل overfitting است. در طول آموزش، lossآموزش و اعتبارسنجی روند کاهشی هماهنگی داشته‌اند و اختلاف معقولی بین آنها حفظ شده است. دقت نهایی تست ۷۶,۶۱% حاکی از عملکرد مناسب مدل است که نسبت به حالت پایه حدود ۲-۳% بهبود نشان می‌دهد. مقادیر(0.7657) precision ، (0.7661) recallو (0.7627) F1-score که بسیار نزدیک به هم هستند، نشانگر عملکرد متوازن مدل در تشخیص تمام کلاس‌ها می‌باشد. نکته قابل توجه ثبات نسبی loss اعتبارسنجی در epoch های پایانی (نوسان حول ۰,۷) است که نشان می‌دهد مدل به خوبی تعمیم یافته و از overfitting جلوگیری شده است. این ترکیب پارامترها اگرچه دقت آموزش را کمی پایین‌تر (۷۹,۳۶%) نگه داشته، اما به مدل کمک کرده تا تعادل بهتری بین یادگیری و تعمیم‌پذیری ایجاد کند. Weight Decay نسبتاً بالا (۰,۰۰۱) همراه با Dropout متوسط (۰,٤) ظاهراً اثر تنظیم‌کنندگی خوبی داشته‌اند. این تنظیمات می‌تواند برای مواردی که جلوگیری از overfitting اولویت دارد، گزینه مناسبی باشد.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که اورفیت بهتر شده است و از اپک ٦ به بعد دچار آن شدیم و تفسیرهای مشابهی مثل قبل نیز می‌توانیم برای آنها ارائه دهیم:


Train and Validation Loss over Epochs
Train Loss
Validation Loss
Loss
Epochs
1.4
1.2
1.0
0.8
0.6
2
4
6
8
10

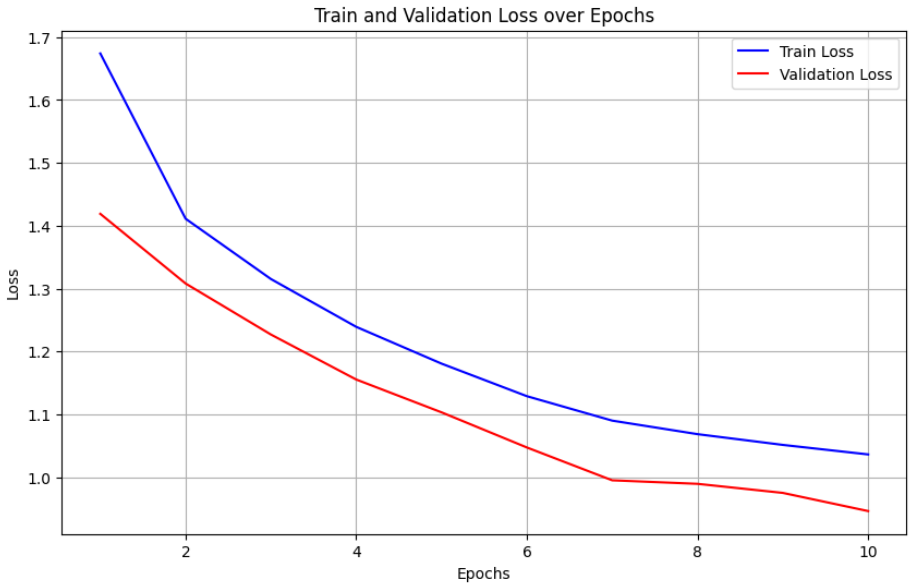

Train and Validation Accuracy over Epochs
Train Accuracy
Validation Accuracy
Accuracy (%)
Epochs
80
75
70
65
60
55
50
45
2
4
6
8
10

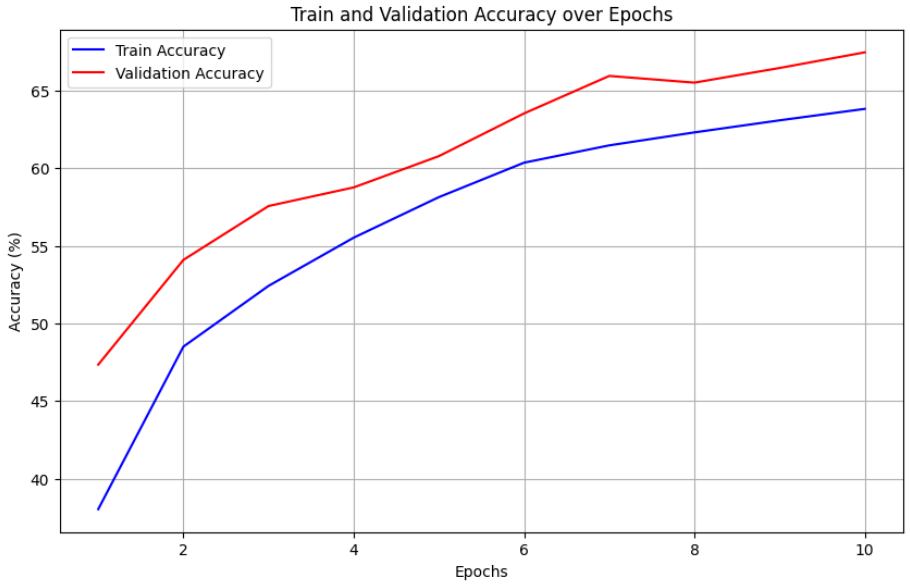
- **Dropout = 0.4 , weight_decay = 0.01**

```
Epoch [1/10], Train Loss: 1.6741, Validation Loss: 1.4190, Train Accuracy: 38.03%
Epoch [2/10], Train Loss: 1.4112, Validation Loss: 1.3082, Train Accuracy: 48.51%
Epoch [3/10], Train Loss: 1.3153, Validation Loss: 1.2271, Train Accuracy: 52.42%
Epoch [4/10], Train Loss: 1.2393, Validation Loss: 1.1555, Train Accuracy: 55.53%
Epoch [5/10], Train Loss: 1.1808, Validation Loss: 1.1033, Train Accuracy: 58.14%
Epoch [6/10], Train Loss: 1.1290, Validation Loss: 1.0475, Train Accuracy: 60.36%
Epoch [7/10], Train Loss: 1.0902, Validation Loss: 0.9952, Train Accuracy: 61.48%
Epoch [8/10], Train Loss: 1.0686, Validation Loss: 0.9897, Train Accuracy: 62.31%
Epoch [9/10], Train Loss: 1.0515, Validation Loss: 0.9752, Train Accuracy: 63.09%
Epoch [10/10], Train Loss: 1.0364, Validation Loss: 0.9463, Train Accuracy: 63.83%

Final Test Accuracy: 67.45%
Precision: 0.6776
Recall: 0.6745
F1-Score: 0.6743
```

نتایج این آزمایش با Dropout=0.4 و Weight Decay=0.01 نشان‌دهنده تأثیر شدید regularization بر عملکرد مدل است. با وجود اینکه مدل توانسته از overfitting جلوگیری کند اما به نظر می‌رسد مقدار بالای Weight Decay (0.01)باعث محدودیت بیش از حد در یادگیری شده است. دقت نهایی تست ٦٧,٤٥% که نسبت به حالت‌های قبلی کاهش محسوسی دارد، نشان می‌دهد این سطح از regularization ممکن است بیش از حد قوی بوده باشد. مقادیر(0.6776) precision ، (0.6745) recallو (0.6743) F1-score همچنان متعادل هستند، اما در سطح پایین‌تری قرار دارند. روند آموزش نشان می‌دهد که مدل به کندی یاد می‌گیرد (دقت آموزش در انتها تنها ٦٣,٨٣%) و به نظر می‌رسد نتوانسته ویژگی‌های پیچیده داده را به خوبی یاد بگیرد. این حالت معمولاً زمانی رخ می‌دهد که مقدار Weight Decay بیش از حد بالا باشد و به طور جدی بروزرسانی وزن‌ها را محدود کند. این نتایج نشان می‌دهد که اگرچه این تنظیمات از overfitting جلوگیری کرده‌اند، اما ممکن است برای این معماری و مجموعه داده خاص، مقدار Weight Decay=0.01 بیش از حد باشد و تعادل مناسبی بین یادگیری و regularization برقرار نکرده باشد. برای بهبود نتایج، می‌توان مقدار Weight Decay را کاهش داد یا از ترکیب متعادل‌تری از Dropout و Weight Decay استفاده کرد.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که دیگر مشکل اورفیت را نداریم و منحنی مربوط به اعتبارسنجی پس از مدتی به مقدار ثابت یا افزایشی تبدیل نشده است و از طرفی برای دقت هم منحنی این دو مجموعه خیلی بهم هم نزدیک هستند و پس در خروجی داریم:


Train and Validation Loss over Epochs
Train Loss
Validation Loss
Loss
Epochs
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1.0
2
4
6
8
10

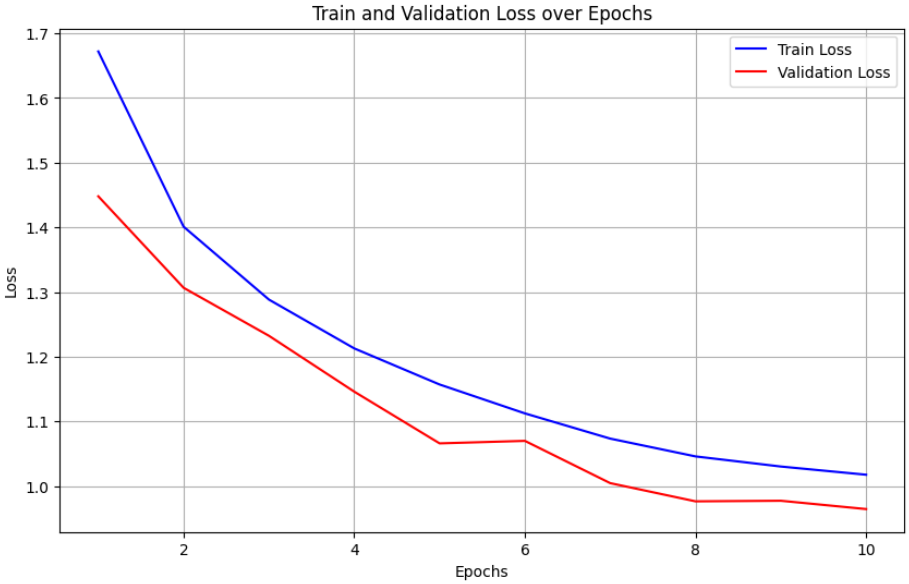

Train and Validation Accuracy over Epochs
Train Accuracy
Validation Accuracy
Accuracy (%)
Epochs
65
60
55
50
45
40
2
4
6
8
10

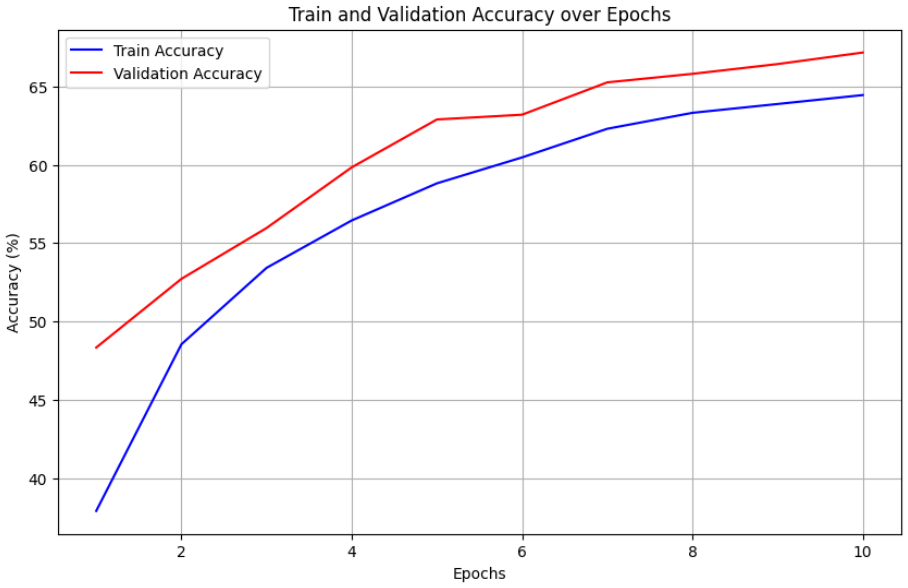
- **Dropout = 0.3 , weight_decay = 0.001**

```
Epoch [1/10], Train Loss: 1.6720, Validation Loss: 1.4480, Train Accuracy: 37.91%
Epoch [2/10], Train Loss: 1.4011, Validation Loss: 1.3066, Train Accuracy: 48.54%
Epoch [3/10], Train Loss: 1.2884, Validation Loss: 1.2324, Train Accuracy: 53.42%
Epoch [4/10], Train Loss: 1.2129, Validation Loss: 1.1461, Train Accuracy: 56.45%
Epoch [5/10], Train Loss: 1.1572, Validation Loss: 1.0661, Train Accuracy: 58.82%
Epoch [6/10], Train Loss: 1.1125, Validation Loss: 1.0700, Train Accuracy: 60.47%
Epoch [7/10], Train Loss: 1.0735, Validation Loss: 1.0047, Train Accuracy: 62.30%
Epoch [8/10], Train Loss: 1.0460, Validation Loss: 0.9765, Train Accuracy: 63.32%
Epoch [9/10], Train Loss: 1.0303, Validation Loss: 0.9774, Train Accuracy: 63.88%
Epoch [10/10], Train Loss: 1.0177, Validation Loss: 0.9646, Train Accuracy: 64.45%

Final Test Accuracy: 67.71%
Precision: 0.6724
Recall: 0.6771
F1-Score: 0.6697
```

نتایج این مدل با Dropout=0.3 و Weight Decay=0.001 نشان‌دهنده محدودیت‌های ناشی از تنظیمات بیش‌ازحد محافظه‌کارانه است. با وجود ثبات نسبی در اختلاف loss آموزش و اعتبارسنجی نتوانسته به دقت مطلوب دست یابد. دقت نهایی تست ٦٧,٧١% و دقت آموزش ٦٤,٤٥% نشان می‌دهد مدل تحت‌آموزش (underfitting)قرار دارد. مقادیر precision (0.6724)، recall (0.6771)و F1-score (0.6697)نه‌تنها پایین‌تر از حالت‌های قبلی هستند، بلکه عدم تعادل جزئی بین آنها مشاهده می‌شود که می‌تواند نشان‌دهنده مشکل در یادگیری ویژگی‌های برخی کلاس‌ها باشد. روند کند بهبود دقت آموزش (از ٣٧,٩١% به ٦٤,٤٥%) حاکی از آن است که مدل با سرعت بسیار کمی یاد می‌گیرد.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که دیگر مشکل اورفیت را نداریم و منحنی مربوط به اعتبارسنجی پس از مدتی به مقدار ثابت یا افزایشی تبدیل نشده است و از طرفی برای دقت هم منحنی این دو مجموعه خیلی بهم هم نزدیک هستند و پس در خروجی داریم:


Train and Validation Loss over Epochs
Train Loss
Validation Loss
Loss
Epochs
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1.0
2
4
6
8
10


Train and Validation Accuracy over Epochs
Train Accuracy
Validation Accuracy
Accuracy (%)
Epochs
65
60
55
50
45
40
2
4
6
8
10

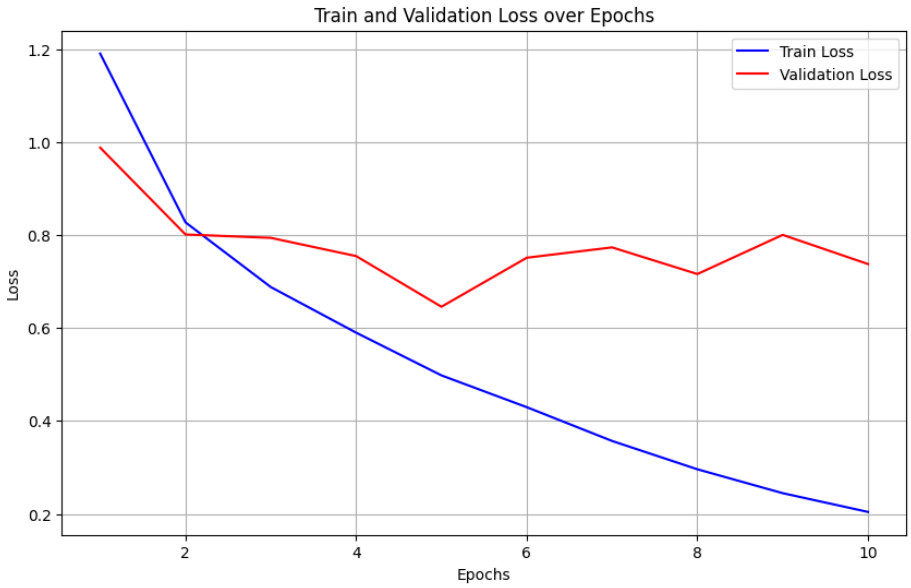
.4

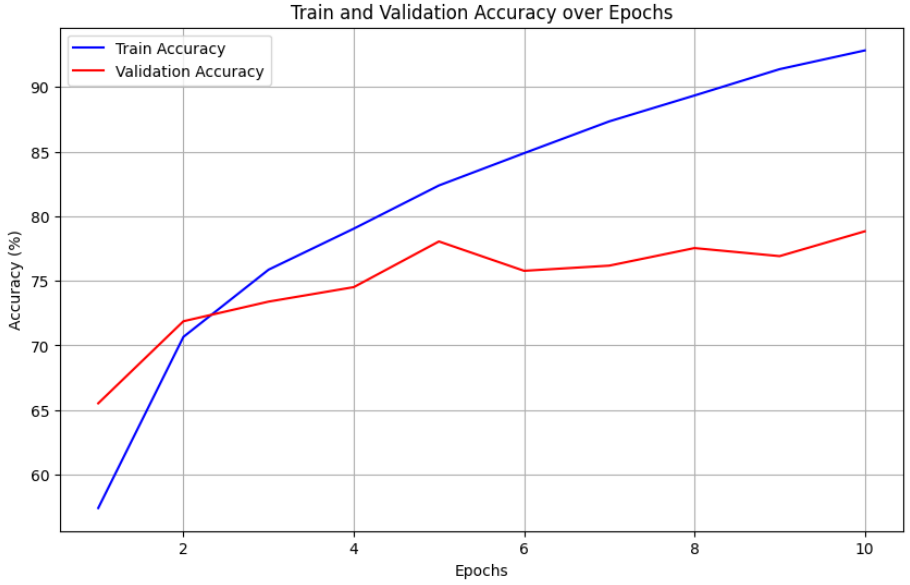
در این بخش، تأثیر افزودن لایه‌های BatchNorm به مدل پایه را بررسی می‌کنیم تا ببینیم آیا این تکنیک می‌تواند سرعت همگرایی و دقت مدل را بهبود بخشد یا خیر. مدل جدید شامل همان معماری ۳ لایه کانولوشن است، اما بعد از هر لایه کانولوشن یک لایه BatchNorm2d اضافه شده است. این لایه‌ها با نرمال‌سازی فعال‌سازی‌های هر لایه، به پایداری و سرعت آموزش کمک می‌کنند. درforward pass ، ابتدا عمل کانولوشن انجام می‌شود، سپس خروجی نرمال‌سازی شده و بعد تابع فعال‌سازی ReLU اعمال می‌گردد. در انتها مانند قبل، لایه‌های تمام‌متصل برای طبقه‌بندی استفاده شده‌اند. مقایسه عملکرد این مدل با مدل پایه نشان می‌دهد که این تکنیک چطور می‌تواند بر یادگیری شبکه تأثیر بگذارد.

```
Epoch [1/10], Train Loss: 1.1901, Validation Loss: 0.9877, Train Accuracy: 57.41%
Epoch [2/10], Train Loss: 0.8271, Validation Loss: 0.8009, Train Accuracy: 70.66%
Epoch [3/10], Train Loss: 0.6879, Validation Loss: 0.7937, Train Accuracy: 75.86%
Epoch [4/10], Train Loss: 0.5898, Validation Loss: 0.7545, Train Accuracy: 79.04%
Epoch [5/10], Train Loss: 0.4979, Validation Loss: 0.6455, Train Accuracy: 82.38%
Epoch [6/10], Train Loss: 0.4296, Validation Loss: 0.7509, Train Accuracy: 84.88%
Epoch [7/10], Train Loss: 0.3569, Validation Loss: 0.7732, Train Accuracy: 87.34%
Epoch [8/10], Train Loss: 0.2960, Validation Loss: 0.7159, Train Accuracy: 89.34%
Epoch [9/10], Train Loss: 0.2446, Validation Loss: 0.8001, Train Accuracy: 91.38%
Epoch [10/10], Train Loss: 0.2043, Validation Loss: 0.7374, Train Accuracy: 92.83%

Final Test Accuracy: 77.71%
Precision: 0.7844
Recall: 0.7771
F1-Score: 0.7792
```

نتایج مدل با Batch Normalization بهبود قابل توجهی را در عملکرد نشان می‌دهد که به وضوح اثرات مثبت این تکنیک را تأیید می‌کند. دقت نهایی تست به ۷۷/۷۱ درصد رسیده که حدود ۳ تا ۴ درصد بهتر از مدل پایه بدون این قابلیت است. از همان ابتدای آموزش شاهد یادگیری سریع‌تر مدل هستیم که با دقت آموزش ۵۷/۴۱ درصد در اولین دوره شروع شده و این نشان‌دهنده اثر تنظیم‌کنندگی مؤثر BatchNorm است. در طول آموزش، اختلاف معقول و کنترل‌شده‌ای بین خطای آموزش و اعتبارسنجی مشاهده می‌شود که معمولاً حول ۰/۵ در دوره‌های پایانی در نوسان است و این نشان می‌دهد مدل کمتر دچار بیش‌برازش شده است. مقادیر precision حدود ۰/۷۸۴۴، recall حدود ۰/۷۷۷۱ و F1-score حدود ۰/۷۷۹۲ که بسیار نزدیک به هم هستند، نشانگر عملکرد متعادل مدل در تشخیص تمامی کلاس‌ها می‌باشد. تغییرات خطای اعتبارسنجی در دوره‌های پایانی بین ۰/۷۱ تا ۰/۸ در نوسان است که نسبت به مدل پایه از پایداری بیشتری برخوردار است. این نتایج به خوبی نشان می‌دهد که Batch Normalization نه تنها سرعت یادگیری را افزایش داده، بلکه از بیش‌برازش نیز جلوگیری کرده و در نهایت به عملکرد کلی بهتری منجر شده است. چنین بهبودی در عملکرد مدل، استفاده از این تکنیک را در معماری‌های عمیق‌تر نیز به عنوان یک انتخاب مناسب پیشنهاد می‌کند، چرا که هم به پایداری آموزش کمک می‌کند و هم نتایج نهایی را بهبود می‌بخشد.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که نمودار اعتبارسنجی نوسانات زیادی دارد و گاهی خطا کاهش پیدا می‌کند و گاهی زیاد می‌شود و برای دقت هم از یک جایی به بعد سرعت افزایش دقت در آموزش نسبت به اعتبارسنجی زیاد می‌شود و این خروجی را در زیر نمیاش دادیم:





5.

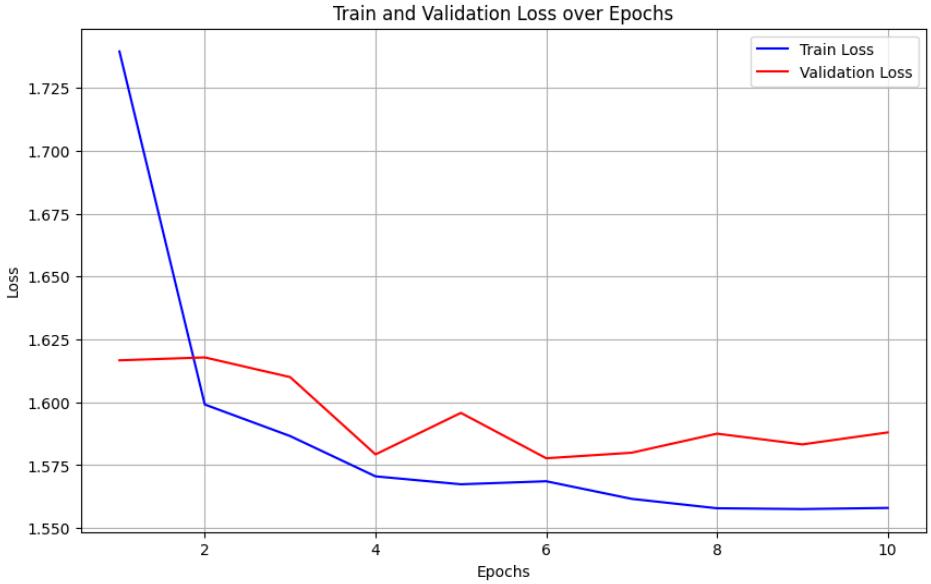
در این بخش، ResNet18 ازپیش‌آموزش‌دیده را روی CIFAR-10 تنظیم دقیق می‌کنیم. ابتدا فقط لایه آخر را آموزش داده، سپس کل مدل را تنظیم می‌کنیم تا عملکرد آن را با مدل‌های قبلی مقایسه کنیم و در ابتدا اول قبل فاین تیون را انجام دادیم. مدل ResNet18 از پیش آموزش‌دیده را بارگذاری کرده‌ایم و ابتدا تمام لایه‌ها به جز لایه آخر را فریز می‌کنیم تا وزن‌شان در طول آموزش به‌روز نشود. سپس لایه خروجی را برای تطابق با ۱۰ کلاس CIFAR-10 جایگزین کرده و فقط این لایه جدید را به مدت ۱۰ دوره آموزش می‌دهیم. این روش به مدل اجازه می‌دهد بدون تغییر در ویژگی‌های یادگرفته‌شده قبلی، فقط یک طبقه‌بند جدید برای داده‌های ما یاد بگیرد. در طول آموزش، loss و accuracy مدل در هر دوره روی داده‌های آموزش و اعتبارسنجی محاسبه و ذخیره می‌شود. در نهایت، مدل روی داده‌های تست ارزیابی شده و معیارهای دقت، precision، recall و F1-score گزارش می‌شوند. این روش معمولاً نتایج بهتری نسبت به آموزش از صفر ارائه می‌دهد، چرا که از دانش قبلی مدل در تشخیص ویژگی‌های عمومی تصاویر استفاده می‌کند.

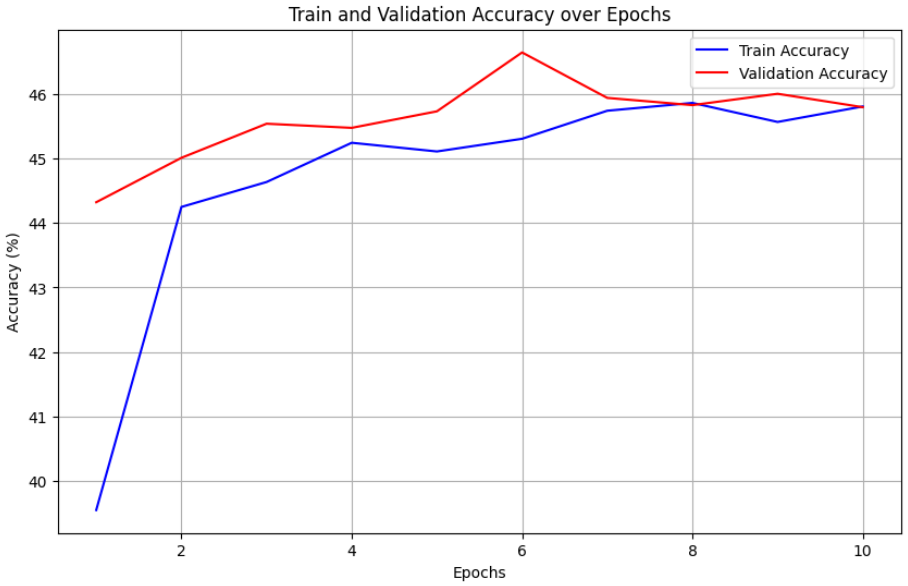
- **Before Fine-Tune ResNet18**

```
Epoch [1/10], Train Loss: 1.7395, Validation Loss: 1.6167, Train Accuracy: 39.55%
Epoch [2/10], Train Loss: 1.5991, Validation Loss: 1.6178, Train Accuracy: 44.25%
Epoch [3/10], Train Loss: 1.5866, Validation Loss: 1.6100, Train Accuracy: 44.63%
Epoch [4/10], Train Loss: 1.5705, Validation Loss: 1.5793, Train Accuracy: 45.24%
Epoch [5/10], Train Loss: 1.5674, Validation Loss: 1.5958, Train Accuracy: 45.11%
Epoch [6/10], Train Loss: 1.5686, Validation Loss: 1.5777, Train Accuracy: 45.30%
Epoch [7/10], Train Loss: 1.5616, Validation Loss: 1.5799, Train Accuracy: 45.74%
Epoch [8/10], Train Loss: 1.5579, Validation Loss: 1.5875, Train Accuracy: 45.86%
Epoch [9/10], Train Loss: 1.5575, Validation Loss: 1.5832, Train Accuracy: 45.56%
Epoch [10/10], Train Loss: 1.5580, Validation Loss: 1.5880, Train Accuracy: 45.80%

Final Test Accuracy: 45.55%
Precision: 0.4637
Recall: 0.4555
F1-Score: 0.4482
```

نتایج نشان می‌دهد مدل به درستی آموزش نمی‌بیند، با دقت تست پایین ٪۴۵/۵۵ و تغییرات ناچیز در طول ۱۰ دوره. خطای آموزش و اعتبارسنجی حول ۱/۵ ثابت مانده که نشان‌دهنده عدم یادگیری موثر است. مقادیر precision، recall و F1-score نزدیک به ۰/۴۵ عملکرد ضعیف و یکنواخت مدل در تمام کلاس‌ها را نشان می‌دهد. این وضعیت معمولاً زمانی رخ می‌دهد که یا نرخ یادگیری نامناسب باشد یا مدل ظرفیت کافی برای یادگیری نداشته باشد.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که برای هر دو نمودار نتایج جالبی برای مدل قبل فاین تیون نداریم و برای هر دو Loss بالاتر از ۱/۵۰۰ است و دقتشان نیز زیر ۵۰ درصد است و خروجی را در ادامه می‌توانیم مشاهده کنیم:


Train and Validation Loss over Epochs
Train Loss
Validation Loss
Loss
Epochs
1.725
1.700
1.675
1.650
1.625
1.600
1.575
1.550
2
4
6
8
10


Train and Validation Accuracy over Epochs
Train Accuracy
Validation Accuracy
Accuracy (%)
Epochs
46
45
44
43
42
41
40
2
4
6
8
10

در این مرحله، استراتژی فاین‌تیونینگ هوشمندانه‌تری را برای مدل ResNet18 پیاده‌سازی کردیم. برخلاف روش قبلی که فقط لایه آخر را آموزش می‌داد، این بار لایه‌های ابتدایی را فریز کرده و به سایر لایه‌ها اجازه دادیم تا در طول آموزش به‌روزرسانی شوند. این رویکرد به مدل امکان می‌دهد تا ضمن حفظ دانش پایه از پیش‌آموخته‌شده، ویژگی‌های خاص مجموعه داده CIFAR-10 را نیز یاد بگیرد. مدل را با نرخ یادگیری ۰,۰۰۱ و به مدت ۱۰ دوره آموزش دادیم. در طول آموزش، lossو accuracy مدل روی داده‌های آموزش و اعتبارسنجی در هر دوره محاسبه و ذخیره شد. سپس مدل روی داده‌های تست ارزیابی شده و معیارهای مختلفی شامل دقت کلی (accuracy)، precision، recallو F1-score محاسبه گردید. این روش معمولاً نتایج بهتری نسبت به حالت قبلی ارائه می‌دهد، چرا که تعادل بهتری بین حفظ دانش قبلی و یادگیری ویژگی‌های جدید برقرار می‌کند. نتایج این روش معمولاً به مراتب بهتر از حالتی است که فقط لایه آخر را آموزش می‌دهیم، چرا که به مدل اجازه می‌دهیم تا ویژگی‌های سطح میانی و بالایی را نیز با داده‌های جدید تطبیق دهد، در حالی که ویژگی‌های پایه‌ای و عمومی‌تر که در لایه‌های اولیه وجود دارند، حفظ می‌شوند.

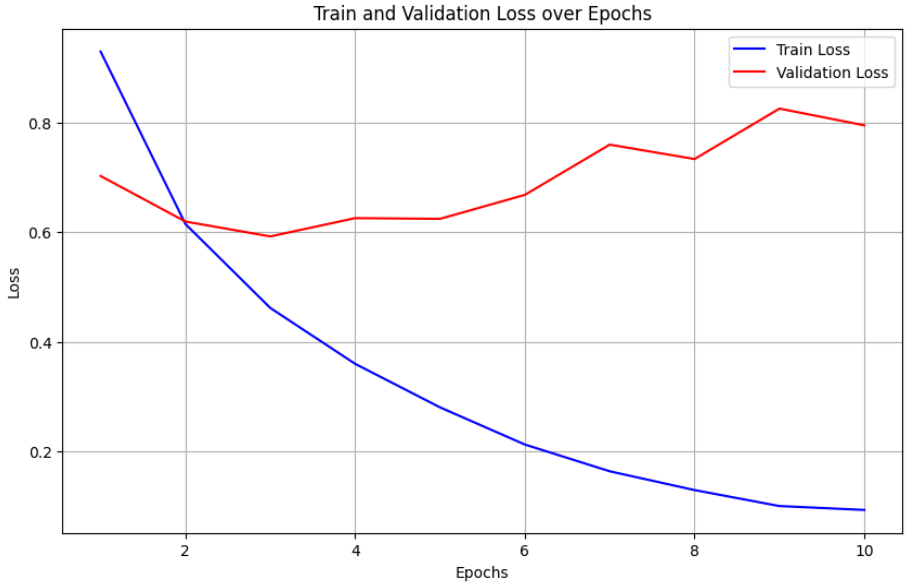
- **After Fine-Tune ResNet18**

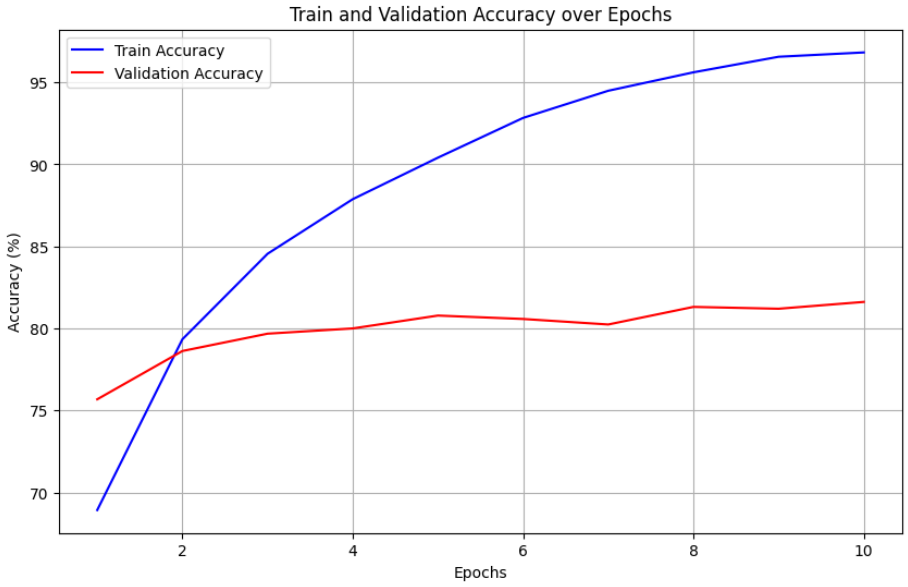
```
Epoch [1/10], Train Loss: 0.9290, Validation Loss: 0.7022, Train Accuracy: 68.93%
Epoch [2/10], Train Loss: 0.6146, Validation Loss: 0.6192, Train Accuracy: 79.34%
Epoch [3/10], Train Loss: 0.4617, Validation Loss: 0.5921, Train Accuracy: 84.55%
Epoch [4/10], Train Loss: 0.3598, Validation Loss: 0.6254, Train Accuracy: 87.88%
Epoch [5/10], Train Loss: 0.2805, Validation Loss: 0.6241, Train Accuracy: 90.41%
Epoch [6/10], Train Loss: 0.2128, Validation Loss: 0.6678, Train Accuracy: 92.83%
Epoch [7/10], Train Loss: 0.1640, Validation Loss: 0.7593, Train Accuracy: 94.49%
Epoch [8/10], Train Loss: 0.1297, Validation Loss: 0.7331, Train Accuracy: 95.61%
Epoch [9/10], Train Loss: 0.1006, Validation Loss: 0.8250, Train Accuracy: 96.56%
Epoch [10/10], Train Loss: 0.0936, Validation Loss: 0.7945, Train Accuracy: 96.82%

Final Test Accuracy: 81.00%
Precision: 0.8093
Recall: 0.8100
F1-Score: 0.8086
```

نتایج این مرحله از فاین‌تیونینگ بهبود چشمگیری نسبت به حالت قبلی نشان می‌دهد. مدل به سرعت یادگیری را آغاز کرده و در اولین دوره به دقت آموزش ٪۶۸/۹۳ و خطای اعتبارسنجی ۰/۷۰۲۲ رسیده است. این نشان‌دهنده تأثیر مثبت آزادسازی برخی لایه‌ها برای آموزش است. در طول ۱۰ دوره آموزش، دقت آموزش به طور پیوسته افزایش یافته و به ٪۹۶/۸۲ رسیده است. با این حال، از دوره چهارم به بعد شاهد افزایش تدریجی خطای اعتبارسنجی هستیم که نشان‌دهنده شروع overfitting است، اگرچه این افزایش نسبتاً کنترل‌شده بوده است. دقت نهایی تست ٪۸۱ و مقادیر precision ۰/۸۰۹۳، recall ۰/۸۱۰۰و ... که بسیار نزدیک به هم هستند، نشان‌دهنده عملکرد متعادل و قوی مدل در تشخیص تمام کلاس‌ها می‌باشد. این نتایج به وضوح نشان می‌دهد که استراتژی فاین‌تیونینگ موفق‌تر از روش قبلی بوده است.

همچنین از بهترین مدل های قبلی ما هم عملکرد بهتری داشت و ۳-٤ درصد دقت را افزایش داده است و توانسته است به دقت و عملکرد بهتری دست پیدا کند.

مثل قبل برای درک بهتر نمودارهای دقت و Loss را نمایش دادیم و می‌بینیم که برخلاف افزایش دقتی که نسبت به مدل‌های قبلی داشته است اما خیلی زود دچار اورفیت و از یک جتیی به بعد دیگر دقت مجموعه اعتبارسنجی تغییر آنچنانی نمی‌کند و تا اپک ٤ نسبتا عملکرد متعادی داشته است اما بعد آن با پدیده اورفیت درگیر خواهیم شد و خروجی این قسمت از کد را در پایین مشاهده می‌کنیم:





6.

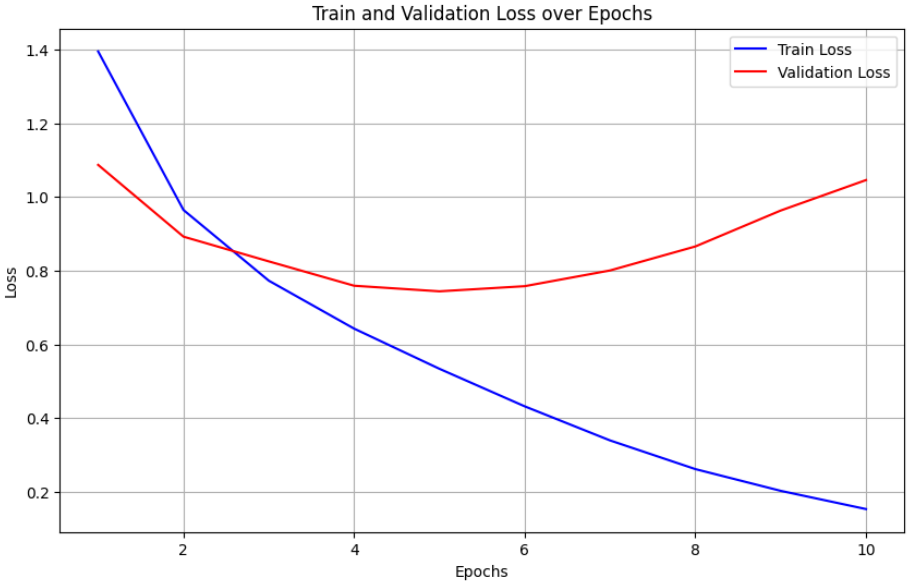
در این بخش باید از افزایش داده (Data Augmentationبا سه روش مختلف (برش تصادفی، چرخش افقی و تغییر رنگ) استفاده کنیم تا تنوع داده‌های آموزشی را افزایش دهیم. سپس تأثیر این تغییرات را روی عملکرد مدل پایه بررسی و تحلیل کنیم تا ببینیم آیا دقت مدل بهبود پیدا می‌کند یا خیر.

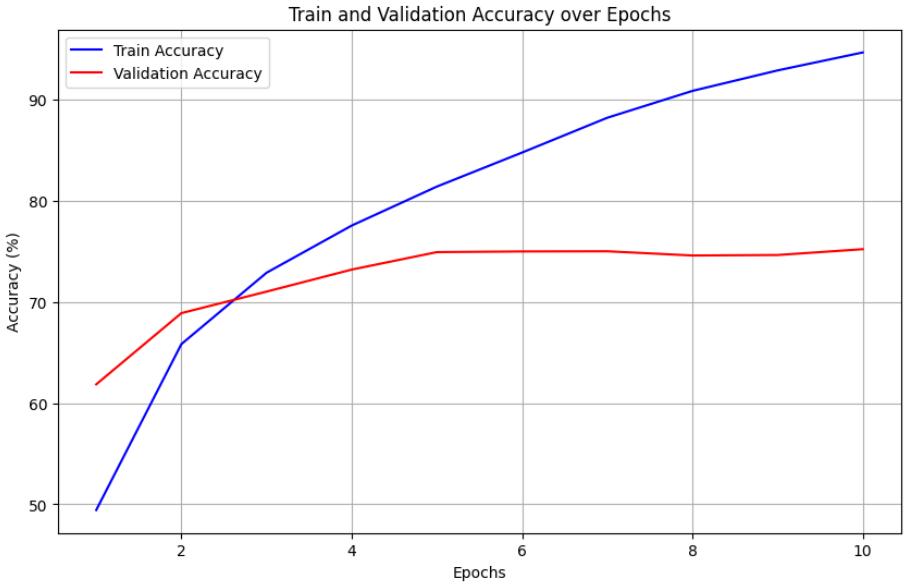
در این مرحله ابتدا با استفاده از کتابخانه transforms.torchvision، سه روش مختلف افزایش داده شامل RandomCrop برای برش تصادفی تصاویر با حاشیه ٤ پیکسل، RandomHorizontalFlip برای چرخش افقی تصادفی تصاویر و ColorJitter برای تغییر تصادفی پارامترهای روشنایی، کنتراست و saturation را پیاده‌سازی کردیم. این تبدیل‌ها فقط روی داده‌های آموزشی اعمال شدند در حالی که برای داده‌های اعتبارسنجی و تست فقط از تبدیل‌های پایه شامل تبدیل به تنسور و نرمال‌سازی استفاده کردیم. در ادامه از همان مدل پایه سوال ۱ استفاده کردیم و مدل را آموزش دادیم که خروجی آن به صورت زیر در آمد:

```
Epoch [1/10], Train Loss: 1.3944, Validation Loss: 1.0867, Train Accuracy: 49.43%
Epoch [2/10], Train Loss: 0.9643, Validation Loss: 0.8920, Train Accuracy: 65.84%
Epoch [3/10], Train Loss: 0.7726, Validation Loss: 0.8249, Train Accuracy: 72.89%
Epoch [4/10], Train Loss: 0.6425, Validation Loss: 0.7589, Train Accuracy: 77.55%
Epoch [5/10], Train Loss: 0.5336, Validation Loss: 0.7437, Train Accuracy: 81.41%
Epoch [6/10], Train Loss: 0.4317, Validation Loss: 0.7577, Train Accuracy: 84.77%
Epoch [7/10], Train Loss: 0.3392, Validation Loss: 0.8003, Train Accuracy: 88.21%
Epoch [8/10], Train Loss: 0.2615, Validation Loss: 0.8654, Train Accuracy: 90.86%
Epoch [9/10], Train Loss: 0.2025, Validation Loss: 0.9628, Train Accuracy: 92.89%
Epoch [10/10], Train Loss: 0.1531, Validation Loss: 1.0455, Train Accuracy: 94.66%

Final Test Accuracy: 75.38%
Precision: 0.7560
Recall: 0.7538
F1-Score: 0.7537
```

نتایج نشان می‌دهد استفاده از augmentation داده‌ها تأثیر مثبتی بر عملکرد مدل داشته است. در طول آموزش، مدل به خوبی یادگیری را آغاز کرده و دقت آموزش از ٪۴۹٫۴۳ در دوره اول به ٪۹۴٫۶۶ در دوره دهم رسیده است. با این حال، از دوره پنجم به بعد افزایش تدریجی loss اعتبارسنجی در حالی که loss آموزش کاهش می‌یابد، نشان‌دهنده شروع overfitting است، اگرچه این افزایش نسبتاً کنترل‌شده بوده است. دقت نهایی تست ٪۷۵٫۳۸ شده است. این نتایج نسبت به مدل پایه بدون augmentation بهبود حدود ۳-۲ درصدی در دقت تست نشان می‌دهد که مؤثر بودن روش‌های افزایش داده را تأیید می‌کند. برای بهبود بیشتر می‌توان از ترکیب این روش‌ها با تکنیک‌های دیگر مانند dropout یا batch normalization استفاده کرد.

سپس نمودارهای مربوط به Loss و دقت را برای آموزش و اعتبارسنجی در زیر نمایش دادیم و به خوبی می‌توانیم توضیحات بالا را به صورت بصری ببینیم و مدل از یک جایی به بعد دچار اورفیت شده است و تفاوت دقت اعتبارسنجی و آموزش نیز تفاضل بیشتری با هم پیدا کرده است :


Train and Validation Loss over Epochs
Train Loss
Validation Loss
Loss
Epochs
1.4
1.2
1.0
0.8
0.6
0.4
0.2
2
4
6
8
10


Train and Validation Accuracy over Epochs
Train Accuracy
Validation Accuracy
Accuracy (%)
Epochs
90
80
70
60
50
2
4
6
8
10

.7

در این بخش سه روش مختلف تنظیم نرخ یادگیری شامل استفاده از نرخ ثابت در طول آموزش، کاهش تدریجی نرخ یادگیری در مراحل مشخص و روش کاهش کسینوسی نرخ یادگیری را پیاده‌سازی و مقایسه می‌کنیم تا تأثیر هر کدام را بر روند یادگیری مدل و نتایج نهایی بررسی کنیم.

- **Fixed Learning Rate:**

```
Epoch [1/10], Train Loss: 1.4393, Validation Loss: 1.1024, Train Accuracy: 47.67%
Epoch [2/10], Train Loss: 1.0006, Validation Loss: 0.9467, Train Accuracy: 64.76%
Epoch [3/10], Train Loss: 0.7992, Validation Loss: 0.8720, Train Accuracy: 72.10%
Epoch [4/10], Train Loss: 0.6686, Validation Loss: 0.7737, Train Accuracy: 76.68%
Epoch [5/10], Train Loss: 0.5570, Validation Loss: 0.7636, Train Accuracy: 80.48%
Epoch [6/10], Train Loss: 0.4622, Validation Loss: 0.7973, Train Accuracy: 83.72%
Epoch [7/10], Train Loss: 0.3690, Validation Loss: 0.8216, Train Accuracy: 86.97%
Epoch [8/10], Train Loss: 0.2907, Validation Loss: 0.9661, Train Accuracy: 89.89%
Epoch [9/10], Train Loss: 0.2301, Validation Loss: 0.9308, Train Accuracy: 91.95%
Epoch [10/10], Train Loss: 0.1784, Validation Loss: 1.0873, Train Accuracy: 93.69%

Final Test Accuracy: 73.79%
Precision: 0.7429
Recall: 0.7379
F1-Score: 0.7389
```

برای این قسمت به عنوان مثال همان مدل پایه سوال اول که نرخ یادگیری ثابتی داشت را آوردیم.

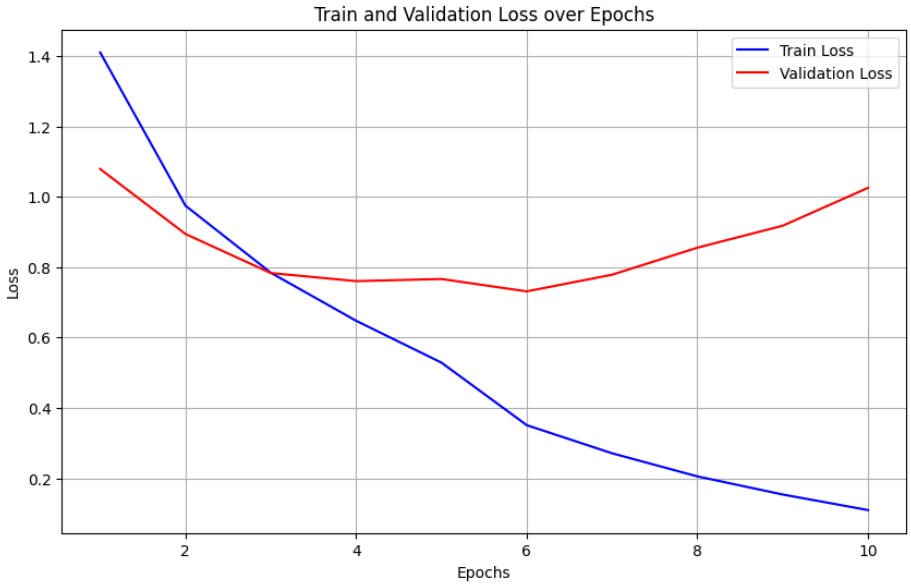
- **Step Learning Rate:**

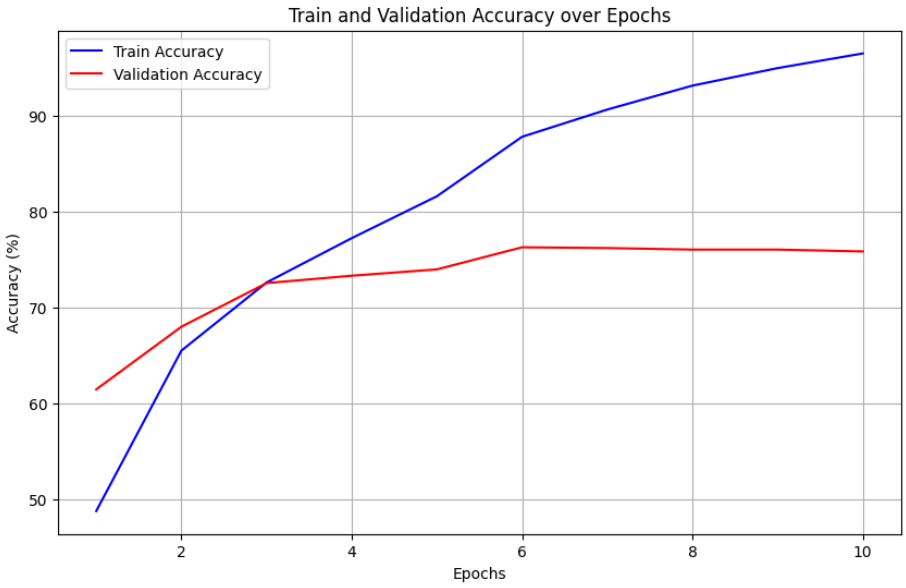
```
Epoch [1/10], Train Loss: 1.4098, Validation Loss: 1.0786, Train Accuracy: 48.82%
Epoch [2/10], Train Loss: 0.9745, Validation Loss: 0.8940, Train Accuracy: 65.53%
Epoch [3/10], Train Loss: 0.7839, Validation Loss: 0.7831, Train Accuracy: 72.62%
Epoch [4/10], Train Loss: 0.6474, Validation Loss: 0.7601, Train Accuracy: 77.24%
Epoch [5/10], Train Loss: 0.5285, Validation Loss: 0.7661, Train Accuracy: 81.60%
Epoch [6/10], Train Loss: 0.3508, Validation Loss: 0.7309, Train Accuracy: 87.79%
Epoch [7/10], Train Loss: 0.2709, Validation Loss: 0.7783, Train Accuracy: 90.63%
Epoch [8/10], Train Loss: 0.2053, Validation Loss: 0.8551, Train Accuracy: 93.12%
Epoch [9/10], Train Loss: 0.1538, Validation Loss: 0.9175, Train Accuracy: 94.94%
Epoch [10/10], Train Loss: 0.1096, Validation Loss: 1.0251, Train Accuracy: 96.47%

Final Test Accuracy: 75.39%
Precision: 0.7561
Recall: 0.7539
F1-Score: 0.7543
```

در این کد از روش کاهش پلکانی نرخ یادگیری (StepLR) استفاده شده که در آن نرخ یادگیری هر ۵ دوره به نصف کاهش می‌یابد ابتدا مدل CNN با نرخ یادگیری اولیه ۰/۰۰۱ آموزش داده می‌شود و پس از هر ۵ دوره، scheduler نرخ یادگیری را کاهش می‌دهد. این روش به مدل کمک می‌کند در مراحل اولیه با نرخ یادگیری بالاتر به سرعت همگرا شود و در مراحل بعدی با کاهش نرخ یادگیری، بهینه‌سازی دقیق‌تری انجام دهد. در طول آموزش، loss و accuracy برای داده‌های آموزش و اعتبارسنجی در هر دوره محاسبه و ذخیره می‌شود و در نهایت عملکرد مدل روی داده تست با معیارهای دقت، precision، recall و F1-score ارزیابی می‌گردد.

نتایج نشان می‌دهد روش StepLR عملکرد مطلوبی داشته است. دقت نهایی ٪۷۵/۳۹ روی داده تست حاصل شده که نشان‌دهنده یادگیری مؤثر مدل است. در طول آموزش مشاهده می‌شود که پس از کاهش نرخ یادگیری در دوره پنجم، مدل بهبود محسوسی در دقت آموزش (از ٪۸۱/۶ به ٪۸۷/۷۹) نشان داده است. با این حال، از دوره هفتم به بعد افزایش تدریجی loss اعتبارسنجی درحالی که loss آموزش کاهش می‌یابد، نشان‌دهنده شروع overfitting است. این نتایج حاکی از آن است که کاهش پلکانی نرخ یادگیری می‌تواند به مدل کمک کند تا بدون گیر کردن در بهینه‌های محلی، همگرایی بهتری داشته باشد.

سپس نمودارهای مربوط به Loss و دقت را برای آموزش و اعتبارسنجی در زیر نمایش دادیم و می‌توانیم مثل بخش‌های قبلی همانطور تجزیه و تحلیل کنیم:





- **Cosine Annealing:**

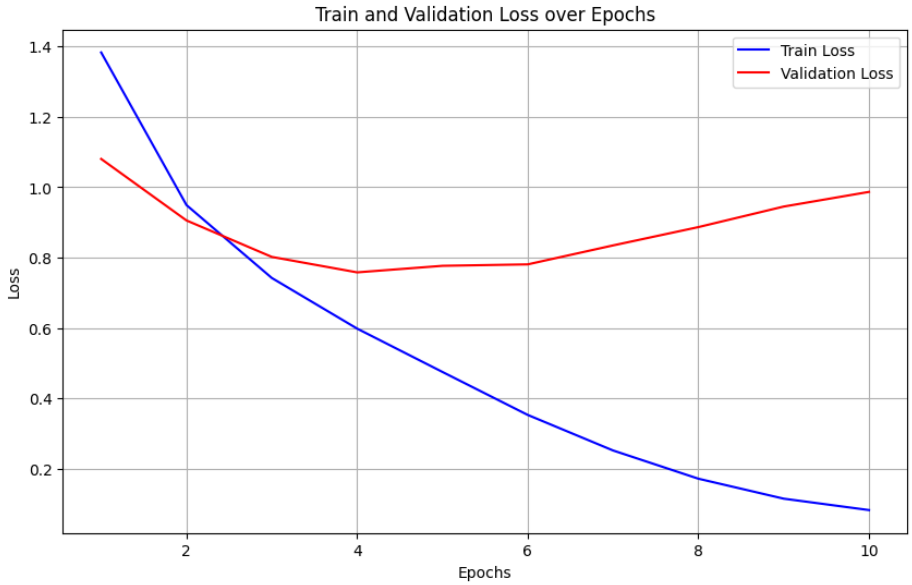
```
Epoch [1/10], Train Loss: 1.3828, Validation Loss: 1.0804, Train Accuracy: 49.88%
Epoch [2/10], Train Loss: 0.9493, Validation Loss: 0.9057, Train Accuracy: 66.43%
Epoch [3/10], Train Loss: 0.7423, Validation Loss: 0.8021, Train Accuracy: 73.95%
Epoch [4/10], Train Loss: 0.5985, Validation Loss: 0.7582, Train Accuracy: 78.98%
Epoch [5/10], Train Loss: 0.4754, Validation Loss: 0.7768, Train Accuracy: 83.38%
Epoch [6/10], Train Loss: 0.3533, Validation Loss: 0.7809, Train Accuracy: 87.66%
Epoch [7/10], Train Loss: 0.2524, Validation Loss: 0.8348, Train Accuracy: 91.59%
Epoch [8/10], Train Loss: 0.1722, Validation Loss: 0.8868, Train Accuracy: 94.65%
Epoch [9/10], Train Loss: 0.1155, Validation Loss: 0.9451, Train Accuracy: 96.90%
Epoch [10/10], Train Loss: 0.0831, Validation Loss: 0.9868, Train Accuracy: 98.08%

Final Test Accuracy: 77.01%
Precision: 0.7687
Recall: 0.7701
F1-Score: 0.7688
```

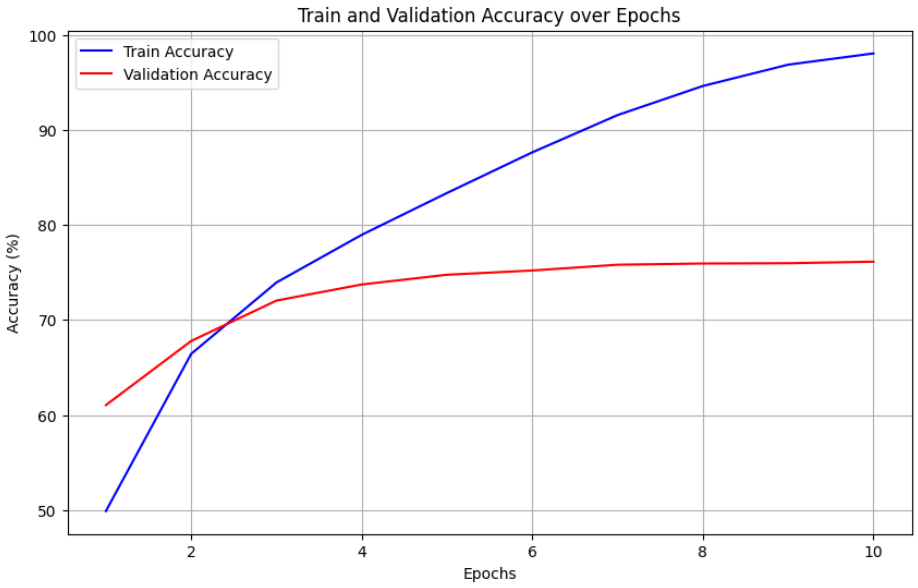
در این کد از روش Cosine Annealing برای تنظیم نرخ یادگیری استفاده شده که طی آن نرخ یادگیری به صورت کسینوسی از مقدار اولیه (۰/۰۰۱) تا حداقل مشخص شده (۰/۰۰۰۱) در طول ۱۰ دوره کاهش می‌یابد. این رویکرد به مدل اجازه می‌دهد در مراحل مختلف آموزش با نرخ‌های یادگیری بهینه‌شده کار کند، به طوری که در ابتدا با نرخ بالاتر تغییرات بزرگتر و در انتها با نرخ پایین‌تر تنظیمات دقیق‌تری انجام دهد. در هر دوره، پس از مرحله آموزش، نرخ یادگیری با فراخوانی scheduler_cosine.step() به‌روزرسانی می‌شود. عملکرد مدل در طول آموزش با محاسبه loss و accuracy برای داده‌های آموزش و اعتبارسنجی پیگیری شده و در نهایت دقت نهایی و

سایر معیارهای ارزیابی روی داده تست محاسبه می‌گردد. این روش معمولاً به همگرایی بهتر و نتایج مطلوب‌تری نسبت به روش‌های ثابت یا پلکانی منجر می‌شود.

نتایج نشان می‌دهد روش Cosine Annealing به خوبی عمل کرده و بهترین دقت تست (۷۷/۰۱٪) را در مقایسه با روش‌های قبلی ارائه داده است. روند آموزش نشان می‌دهد مدل به صورت پیوسته و پایدار یادگیری داشته، به طوری که دقت آموزش از ۴۹/۸۸٪ به ۹۸/۰۸٪ رسیده است. اگرچه از دوره پنجم به بعد افزایش آرام loss اعتبارسنجی مشاهده می‌شود (از ۰/۷۷۶۸ به ۰/۹۸۶۸)، اما این افزایش نسبت به روش‌های دیگر کنترل‌شده‌تر بوده است. این نتایج حاکی از آن است که کاهش کسینوسی نرخ یادگیری به مدل اجازه داده تا با سرعت مناسب یاد بگیرد و در عین حال از overfitting جلوگیری کند. بهبود حدود ۱/۵ تا ۲ درصدی دقت تست نسبت به روش StepLR، مؤثر بودن این روش را تأیید می‌کند.

سپس نمودارهای مربوط به Loss و دقت را برای آموزش و اعتبارسنجی در زیر نمایش دادیم و می‌توانیم مثل بخش‌های قبلی همانطور تجزیه و تحلیل کنیم:





مقایسه نتایج سه روش تنظیم نرخ یادگیری نشان می‌دهد روش کسینوسی با کسب دقت تست ۷۷/۰۱ درصد بهترین عملکرد را داشته که حدود ۳/۲۲ درصد بهتر از روش نرخ ثابت (۷۳/۷۹ درصد) و ۱/۶۲ درصد بهتر از روش کاهش پلکانی (۷۵/۳۹ درصد) است. روش کسینوسی نه تنها به بالاترین دقت دست یافته، بلکه با حفظ اختلاف معقول بین دقت آموزش و تست (حدود ۲۱ درصد) نشان داده بهتر از سایر روش‌ها از overfitting جلوگیری کرده است. در حالی که روش نرخ ثابت با اختلاف ۲۰ درصدی و کاهش پلکانی با اختلاف ۲۳ درصدی بین دقت آموزش و تست، بیش‌برازش بیشتری نشان دادند. از نظر معیارهای دیگر مانند precision، recall و F1-score نیز روش کسینوسی با حدود ۰/۷۷ عملکرد متعادل‌تری نسبت به روش کاهش پلکانی (۰/۷۵) و نرخ ثابت (۰/۷۴) ارائه داده است. این برتری احتمالاً به دلیل کاهش تدریجی و غیرخطی نرخ یادگیری در روش کسینوسی است که به مدل اجازه می‌دهد در مراحل مختلف آموزش با نرخ‌های بهینه یاد بگیرد. در مقابل، روش کاهش پلکانی اگرچه نسبت به نرخ ثابت بهتر عمل کرد، اما تغییرات ناگهانی نرخ یادگیری در دوره پنجم باعث ایجاد نوساناتی در روند یادگیری شد. در نتیجه می‌توان گفت روش کسینوسی با توجه به عملکرد پایدارتر و نتایج بهتر، مناسب‌ترین انتخاب برای این مدل و مجموعه داده بوده است.

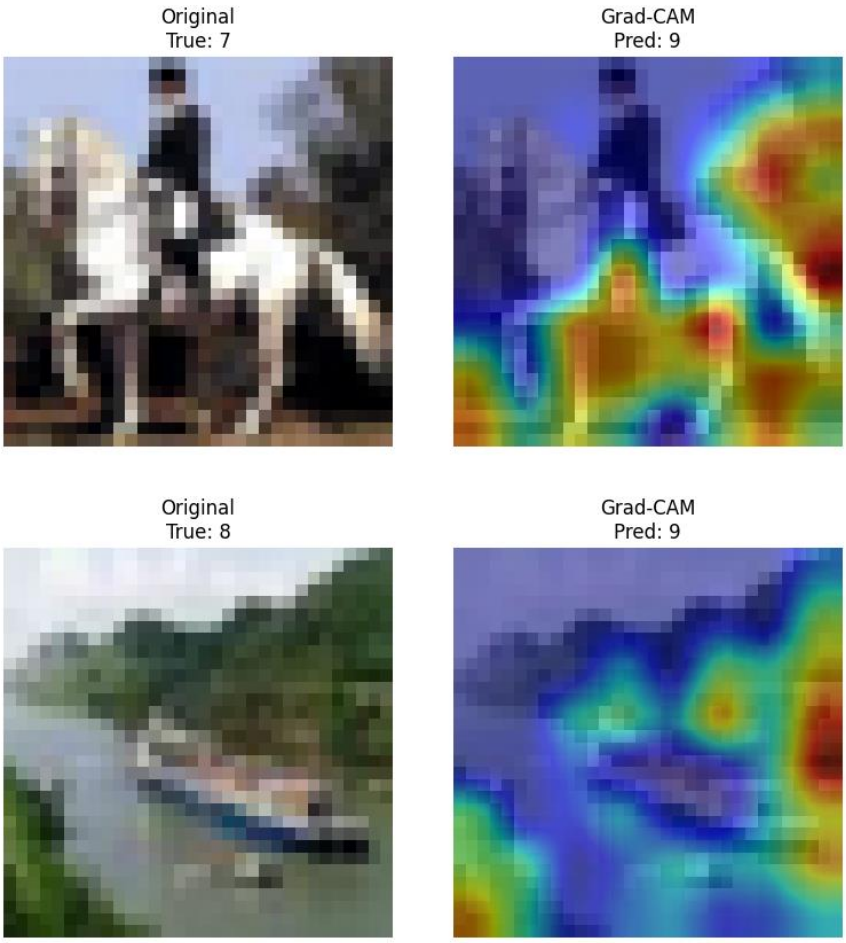
8.

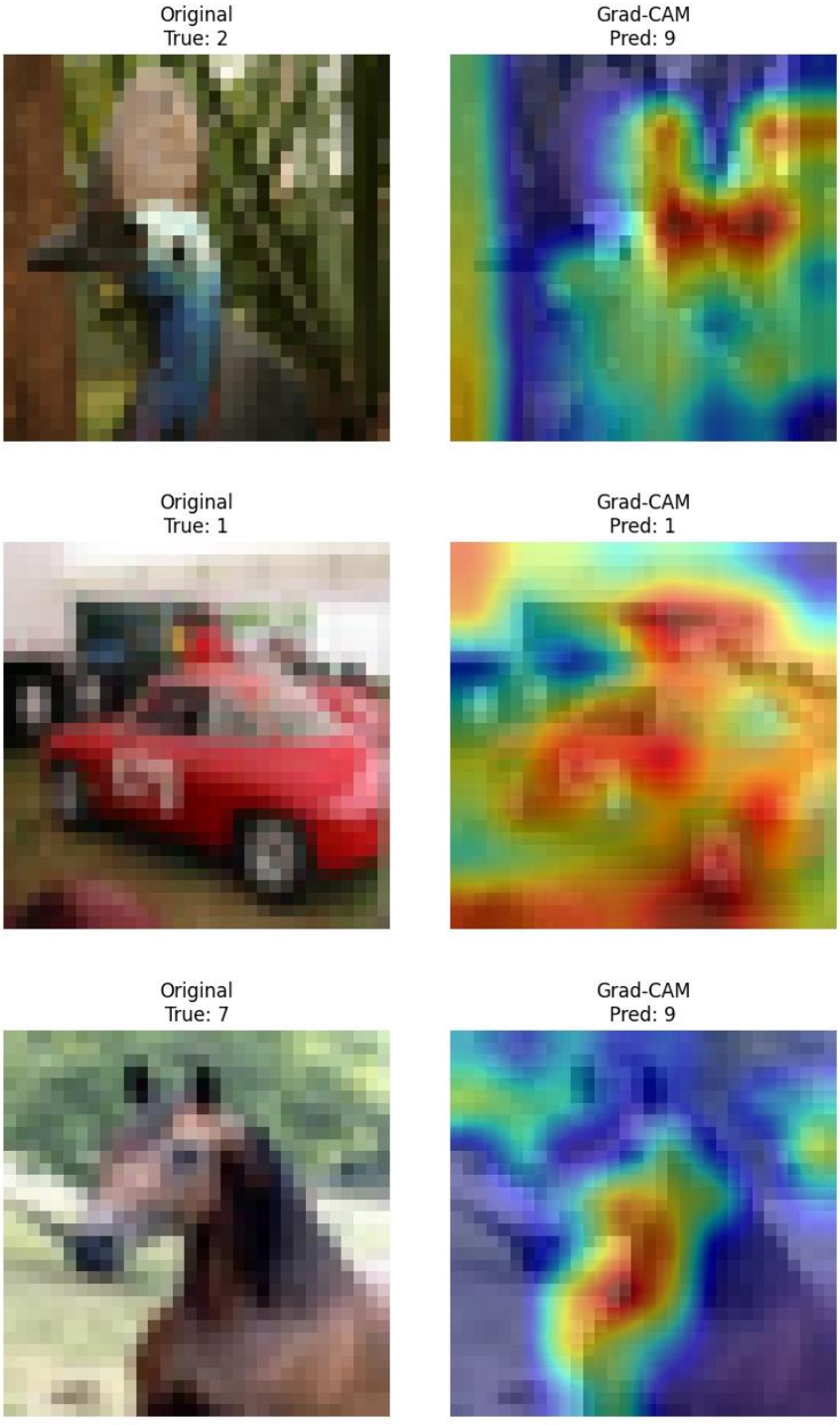
در این بخش باید با استفاده از Grad-CAM، نقشه‌های حرارتی برای پنج تصویر تصادفی از مجموعه تست ایجاد کنیم تا ببینیم مدل بر چه بخش‌هایی از تصویر تمرکز کرده است. سپس تحلیل می‌کنیم که آیا این تمرکز بر روی ویژگی‌های مرتبط با کلاس تصویر است یا خیر، که به درک ما از تصمیم‌گیری‌های مدل کمک می‌کند.

در این کد از مدل CNN_BatchNorm که قبلاً تعریف شده برای تولید نقشه‌های فعال‌سازی Grad-CAM استفاده می‌کنیم. ابتدا مدل را در حالت evaluation قرار داده و یک extractor برای Grad-CAM روی لایه

کانولوشن سوم تنظیم می‌کنیم. سپس با تابع show_gradcam_pair تصاویر اصلی را همراه با نقشه‌های حرارتی Grad-CAM نمایش می‌دهیم که مناطق مهم برای تصمیم‌گیری مدل را برجسته می‌کند. در نهایت برای ۵ نمونه تصادفی از مجموعه تست، این نقشه‌ها را تولید و نمایش می‌دهیم تا ببینیم مدل روی کدام بخش‌های تصویر تمرکز کرده است. این تحلیل به ما کمک می‌کند درک بهتری از رفتار مدل و منطق تصمیم‌گیری آن داشته باشیم.

در نهایت نیز به طور تصادفی ۵ نمونه را از مجموعه تست برای این بخش انتخاب کردیم و خروجی‌های تصویر اصلی و Grad-CAM مربوط به هرکدام را نمایش دادیم (به عنوان مثال در تصویر اول، مدل اشتباه پیش‌بینی کرده است (کلاس ۹ به جای کلاس ۷). نقشه حرارتی نشان می‌دهد مدل روی بخش‌های نامربوط تصویر تمرکز کرده که منجر به این خطا شده است، این می‌تواند نشان‌دهنده نیاز به بهبود معماری مدل یا افزایش داده‌های آموزشی باشد و یا در تصویر چهارم در صفحه بعد نشان می‌دهد که پیش‌بینی صحیح انجام شده (کلاس ۱) و همچنین مناطق پررنگ در نقشه Grad-CAM نشان می‌دهد مدل به درستی روی ویژگی‌های کلیدی شیء در تصویر تمرکز کرده است. این تطابق بین مناطق فعال و بخش‌های معنادار تصویر، نشان‌دهنده یادگیری صحیح مدل برای این کلاس است):





«. . . اردیبهشت‌ماه ۱۴۰۴ . . .»