فاز اول :

1. ساختار مدل CNN

* لایه‌های کانولوشنال:
  + 3 لایه کانولوشن با کرنل x3 و فیلترهای به ترتیب 32، 64 و 128
  + از تابع فعال‌سازی ReLU استفاده شده است
* لایه‌های Pooling
  + لایه‌های MaxPooling با سایز x2 بعد از هر لایه کانولوشن
  + کاهش ابعاد داده از x32 به x8 در خروجی آخرین لایه Pooling
* لایه‌های Fully Connected
  + لایه Flatten برای تبدیل داده‌های 3بعدی به 1بعدی
  + لایه Dense با 128 نورون و تابع فعال‌سازی ReLU
  + لایه Dropout با نرخ 0.5 برای کاهش بیش‌برازش
  + لایه خروجی با 10 نورون و تابع فعال‌سازی softmax برای طبقه‌بندی 10 کلاس

2. پارامترهای کلیدی مدل

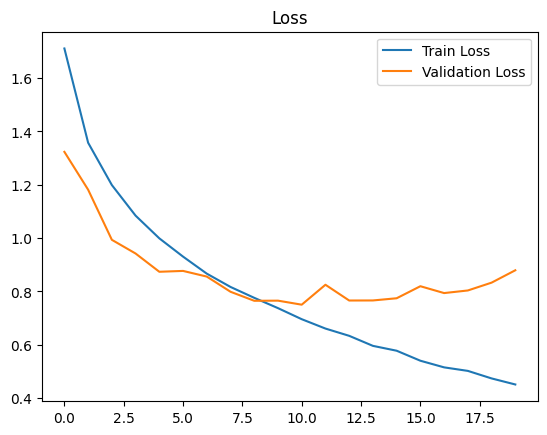
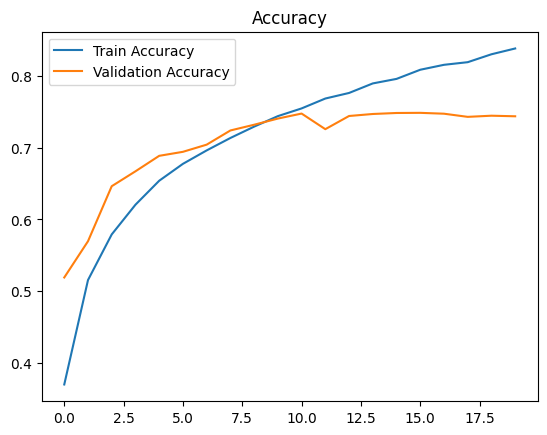
* تعداد پارامترها356,810:
* بهینه‌ساز: Adam با تنظیمات پیش‌فرض
* تابع زیان Categorical Crossentropy
* معیار ارزیابی: دقت (Accuracy)

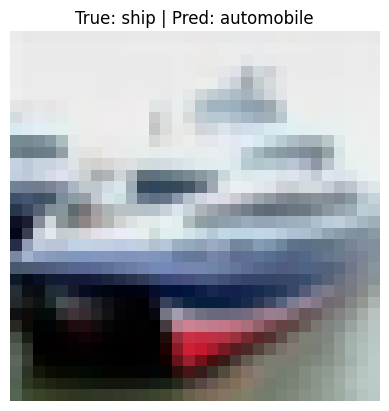
3. پیش‌پردازش داده‌ها

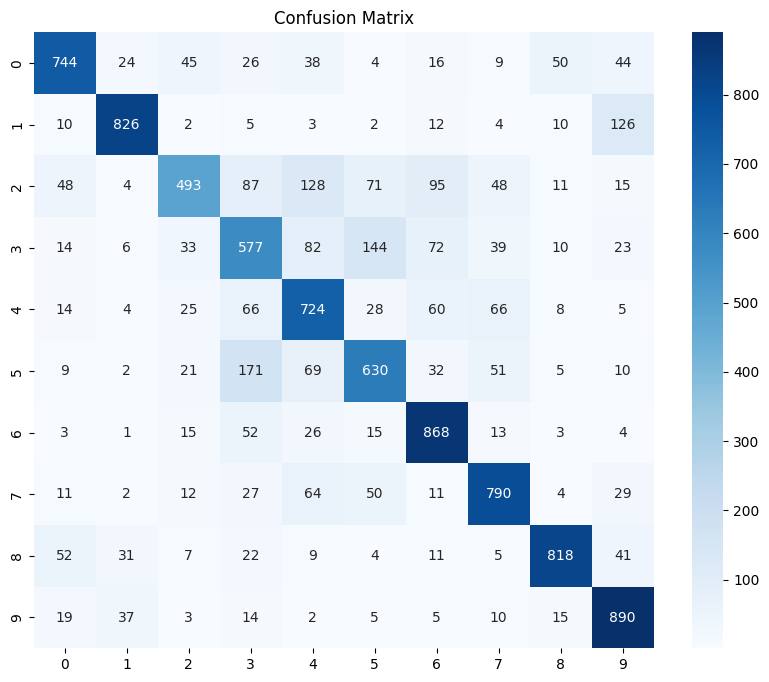
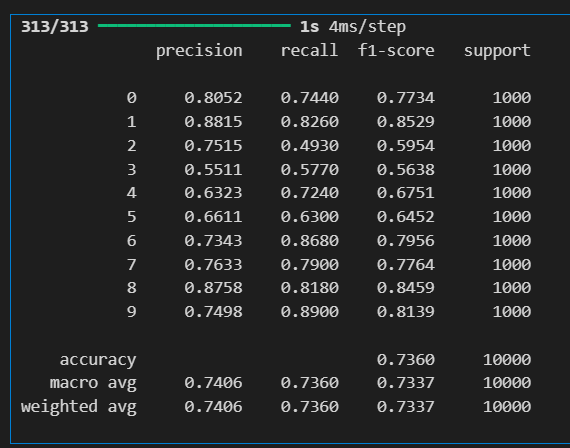
* نرمال‌سازی: مقادیر پیکسل‌ها به بازه [0,1] تبدیل شده‌اند
* تقسیم داده‌ها:
  + 45000 نمونه آموزشی
  + 5000 نمونه اعتبارسنجی
  + 10000 نمونه تست

4. فرآیند آموزش

* دوره‌های آموزش: 20 دوره (Epoch)
* اندازه دسته: 64 نمونه در هر به‌روزرسانی

**نتیجه** :

در این مرحله می بینیم که با افزایش Epoch عکس ها به خوبی train نمی شوند و فقط دقت داده train بالا می رود ولی validation تغییری نمی کند (پس از مدتی عملا یادگیری ندارد ) و overfit می شود.

['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']

در عکس های نزدیک به هم (حیوانات \_ تراکتور وماشین) می بینیم که ممکن است اشتباه بگیرید

راه های کلی افزایش دقت در این فاز: بهبود لایه ها و معماری و یا پیدا کردن کانولوشن که داده های بهتری برای train به ما بدهند

فاز 1-2 :

1. افزودن لایه‌های جدید و تغییر معماری شبکه

* **افزودن Batch Normalization**
  + لایه‌های BatchNorm بعد از هر لایه کانولوشن اضافه شده‌اند
  + کمک به پایدارسازی آموزش و افزایش سرعت همگرایی
* **معماری عمیق‌تر**:
  + هر بلوک کانولوشن اکنون دارای دو لایه Conv2D به جای یک لایه است
  + بلوک اول: دو لایه 3x3 با 32 فیلتر
  + بلوک دوم: دو لایه 3x3 با 64 فیلتر
* **تغییر نرخ Dropout**

2. افزودن Data Augmentation

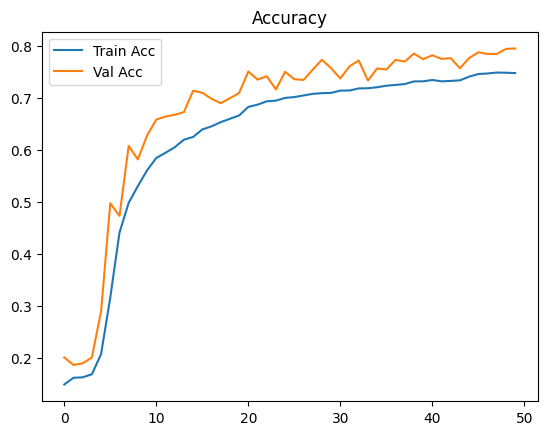
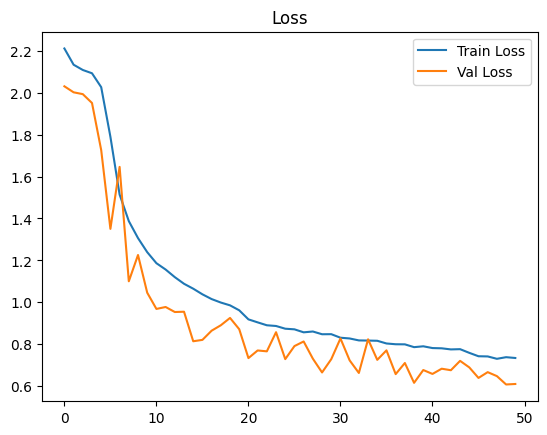
* **ایجاد ImageDataGenerator**
  + چرخش تصاویر تا 15 درجه
  + جابجایی افقی و عمودی (10%)
  + چرخش افقی
  + زوم و برش (10%)
  + پر کردن نقاط خالی با روش 'nearest'
* **استفاده در فرآیند آموزش**:
  + جایگزینی x\_train با جریان داده‌های افزایش یافته

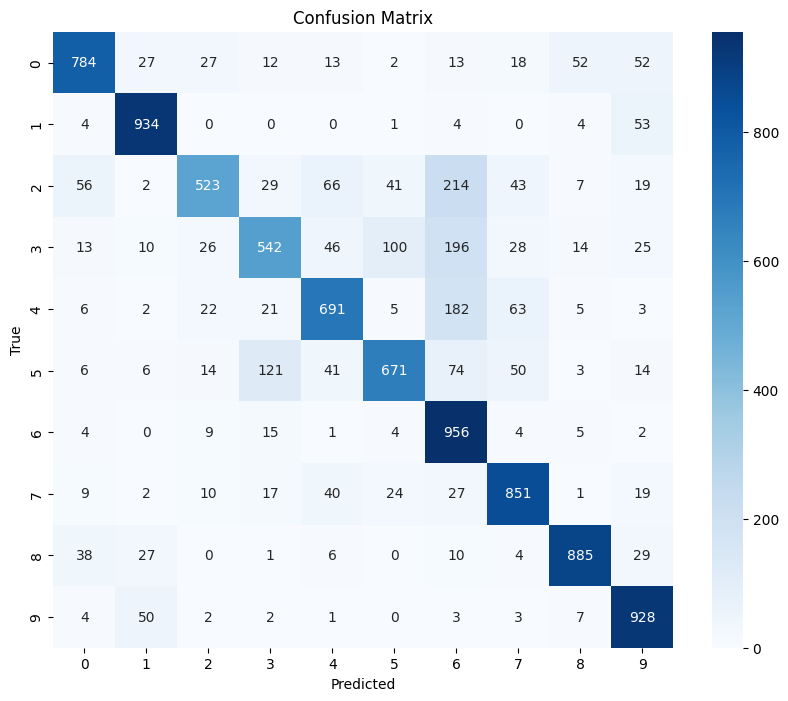
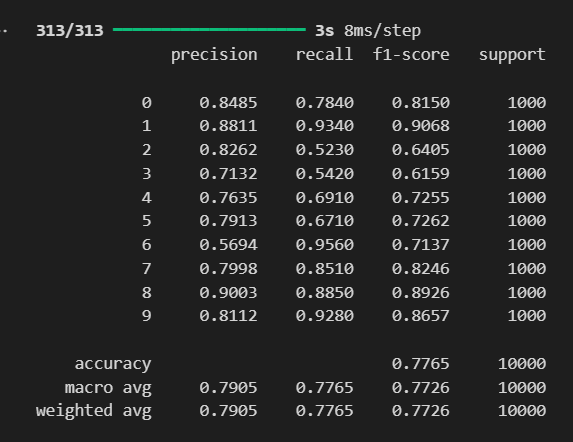
3. افزودن Callback‌های پیشرفته

* **Early Stopping**
  + توقف آموزش وقتی validation loss برای 10 دوره بهبود نیابد
  + بازگردانی بهترین وزن‌های مدل
* **Model Checkpoint**
  + ذخیره خودکار بهترین مدل در فایل 'best\_model.keras'
* **Reduce Learning Rate on Plateau**
  + کاهش نرخ یادگیری به نصف وقتی validation loss برای 5 دوره بهبود نیابد
  + حداقل نرخ یادگیری: 1e-6

4. تغییرات در آموزش مدل

* **افزایش حداکثر دوره‌های آموزش**:
  + از 20 دوره به 50 دوره افزایش یافته
* **تنظیم صریح نرخ یادگیری**:
  + Adam(learning\_rate=0.001) به جای مقدار پیش‌فرض

در اینجا می بینیم که با در بیشترموارد دقت train کمی کمتر از validation می شود و با افزایش دقت train ، دقت validation نیز زیاد می شود و overfit نداریم.



['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','truck']

تداخل بین ماشین الات در اینجا کاهش یافت و به حد مناسبی رسید اما همچنان در حیوانات اشتباه گرفتن با یکدیگر مشاهده می شود

می بینیم که خیلی از حیوانات را قورباغه تشخیص می دهد

بهبود یافته اما به یک سمت بایاس شده

راه های افزایش دقت در این فاز: افزایش دور اموزش (چون هنوز شیب افزایش دقت در val مثبت است)،بهبود لایه ها و معماری و کانولوشن ، می توان از Augmentation بیشتر (چه مقداری و چه اضافه کردن parameter) استفاده کرد و استانداردسازی کرد، **Hyperparameter Tuning**

فاز 2-2 )نسخه نهایی):

1. **معماری پیشرفته‌تر شبکه (Improved Architecture)**

* **اضافه شدن بلوک کانولوشن سوم**:
  + بلوک جدید با 128 فیلتر و دو لایه کانولوشن
  + افزایش عمق شبکه برای استخراج ویژگی‌های پیچیده‌تر
* **جایگزینی Flatten با GlobalAveragePooling2D**
  + کاهش قابل توجه پارامترها (از 128,000 به 128)
  + کاهش بیش‌برازش و بهبود تعمیم‌پذیری
* **اضافه شدن لایه‌های متراکم (Dense) بیشتر**:
  + دو لایه Dense با 256 و 128 نورون
  + افزایش ظرفیت مدل برای یادگیری روابط پیچیده‌تر

2. **منظم‌سازی پیشرفته (Enhanced Regularization)**

* **افزودن Regularizer به لایه‌ها**:
  + L2 regularization در لایه‌های کانولوشن
  + ترکیب L1 و L2 (l1\_l2) در لایه‌های Dense

3. **پیش‌پردازش داده‌های پیشرفته (Advanced Preprocessing)**

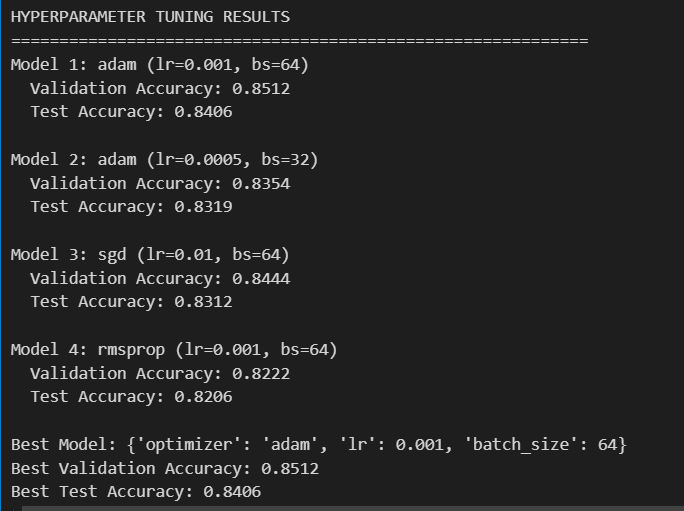
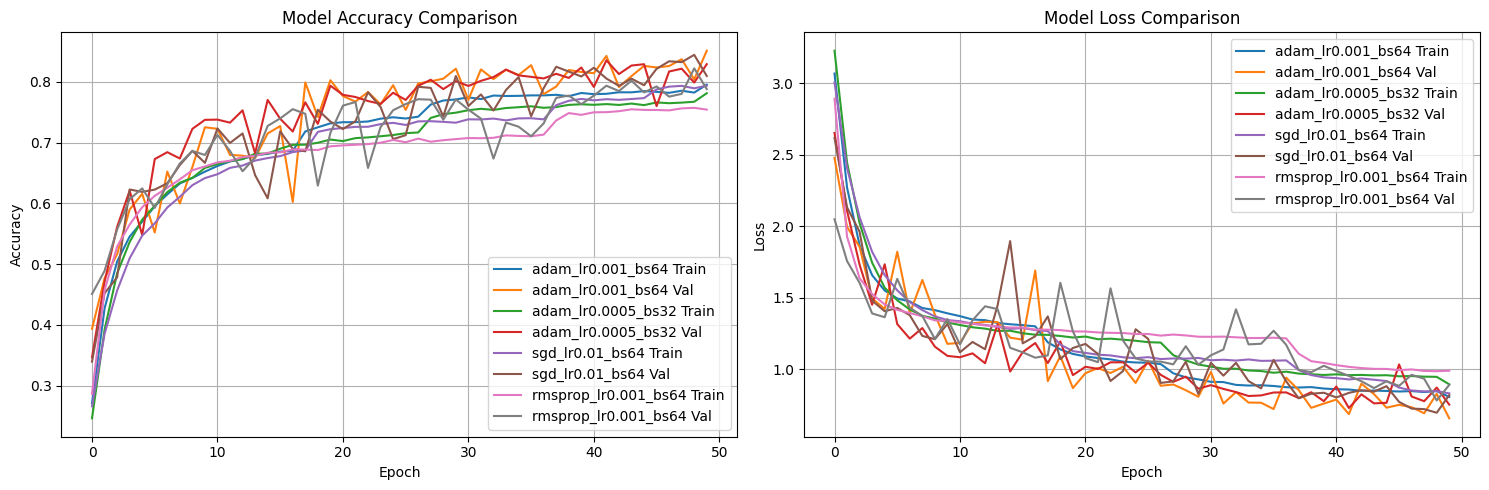
* **استانداردسازی داده‌ها**:
  + محاسبه mean و std از داده‌های آموزشی
  + اعمال استانداردسازی به همه داده‌ها
* **افزایش داده‌های پیشرفته‌تر**:
  + افزودن تنظیم روشنایی (brightness\_range)
  + افزودن تغییر کانال‌رنگی (channel\_shift\_range)
  + افزایش محدوده تغییرات (20 درجه چرخش، 20% جابجایی)

4. **بهینه‌سازی فراگیر (Hyperparameter Tuning)**

* **آزمایش چندین بهینه‌ساز**:
  + Adam، SGD و RMSprop با نرخ‌های یادگیری مختلف
* **تست ترکیب‌های مختلف هایپرپارامترها**:
  + 4 ترکیب مختلف از بهینه‌ساز، نرخ یادگیری و اندازه دسته

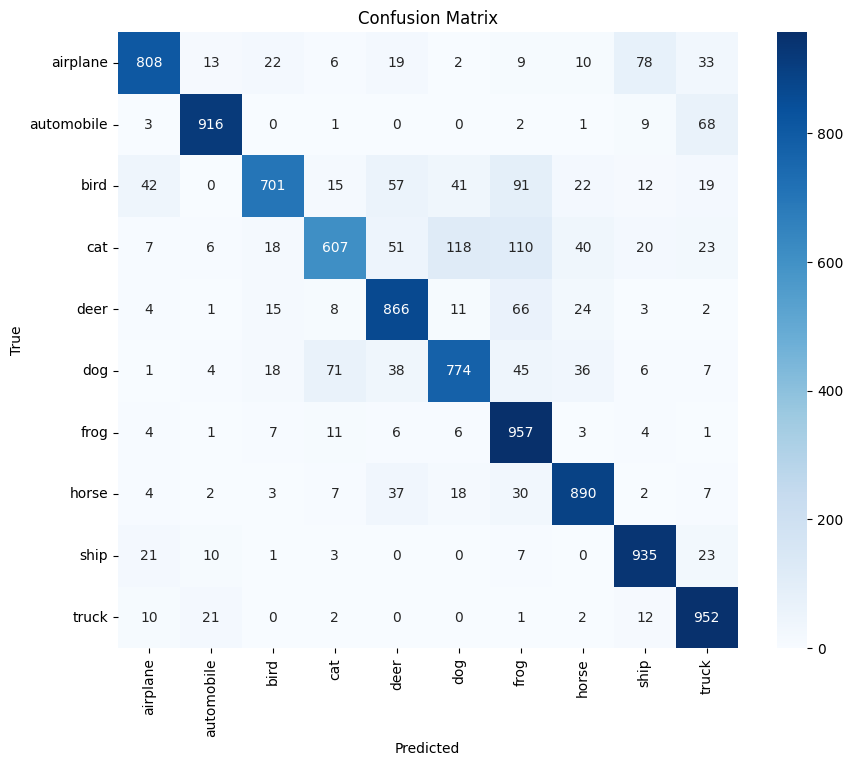
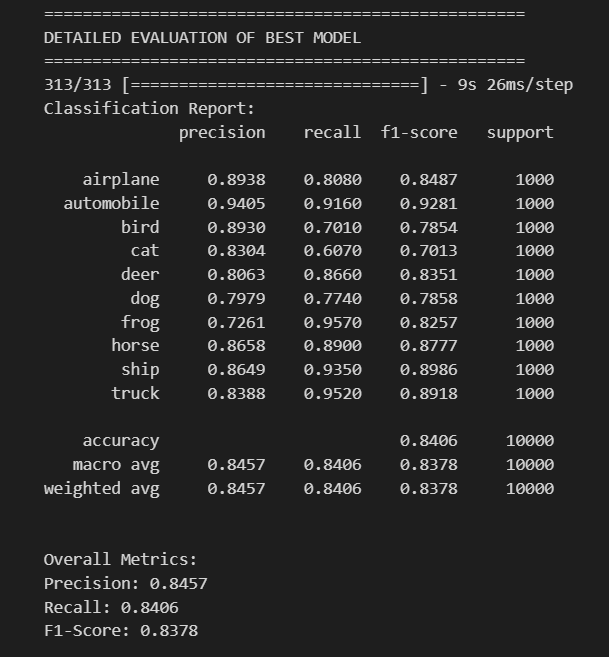
5.**بهینه‌سازی آموزش (Training Optimization)**

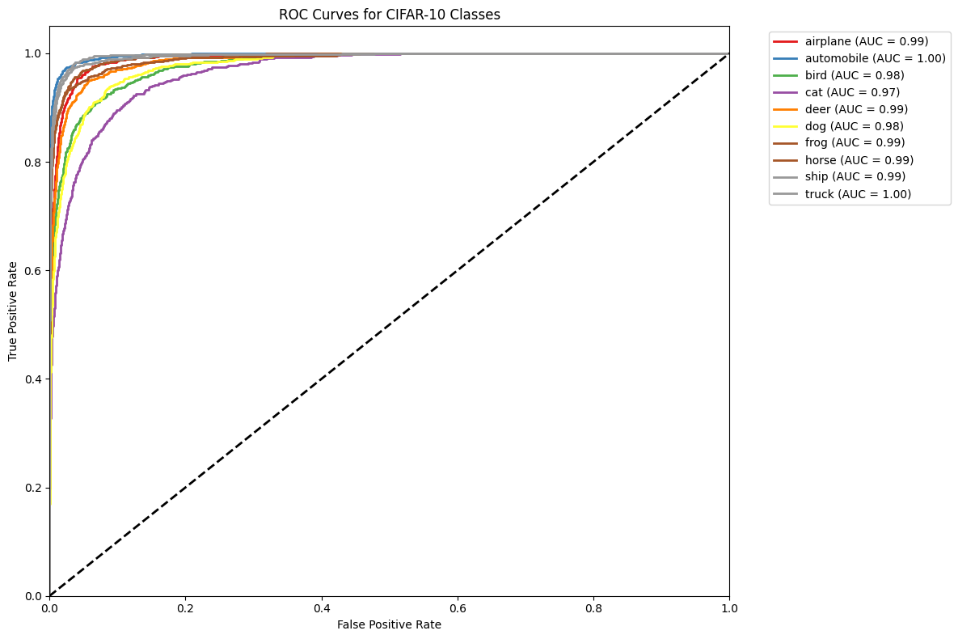
* **تنظیم پویای نرخ یادگیری**
* **توقف زودهنگام هوشمند**
* **ذخیره بهترین مدل(دقت در val)**

نتایج:

می توان دید که در adam-ir0.001 و adam-ir0.0005 به صورت کلی دقت val از train بیشتر است (overfit نشده)

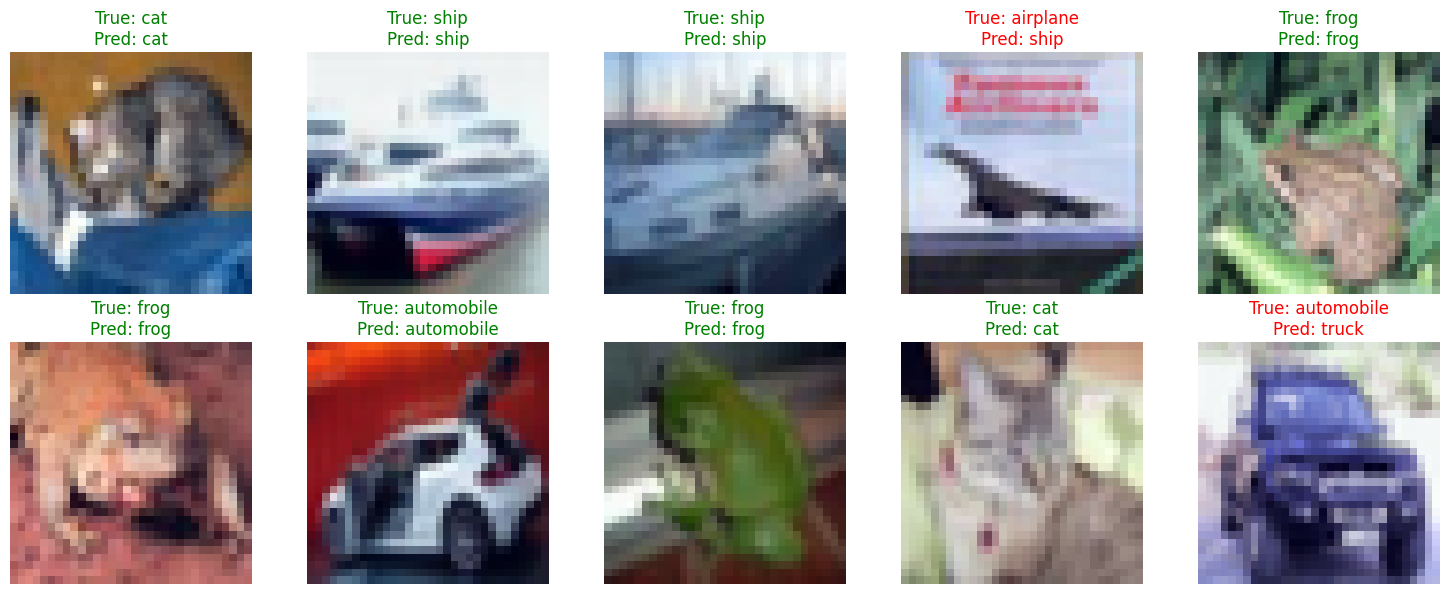
تمام مدل ها به خوبی train شده اند (شاید در بعضی نقاط دقت val کم شود اما به صورت کلی خوب هستند)

****



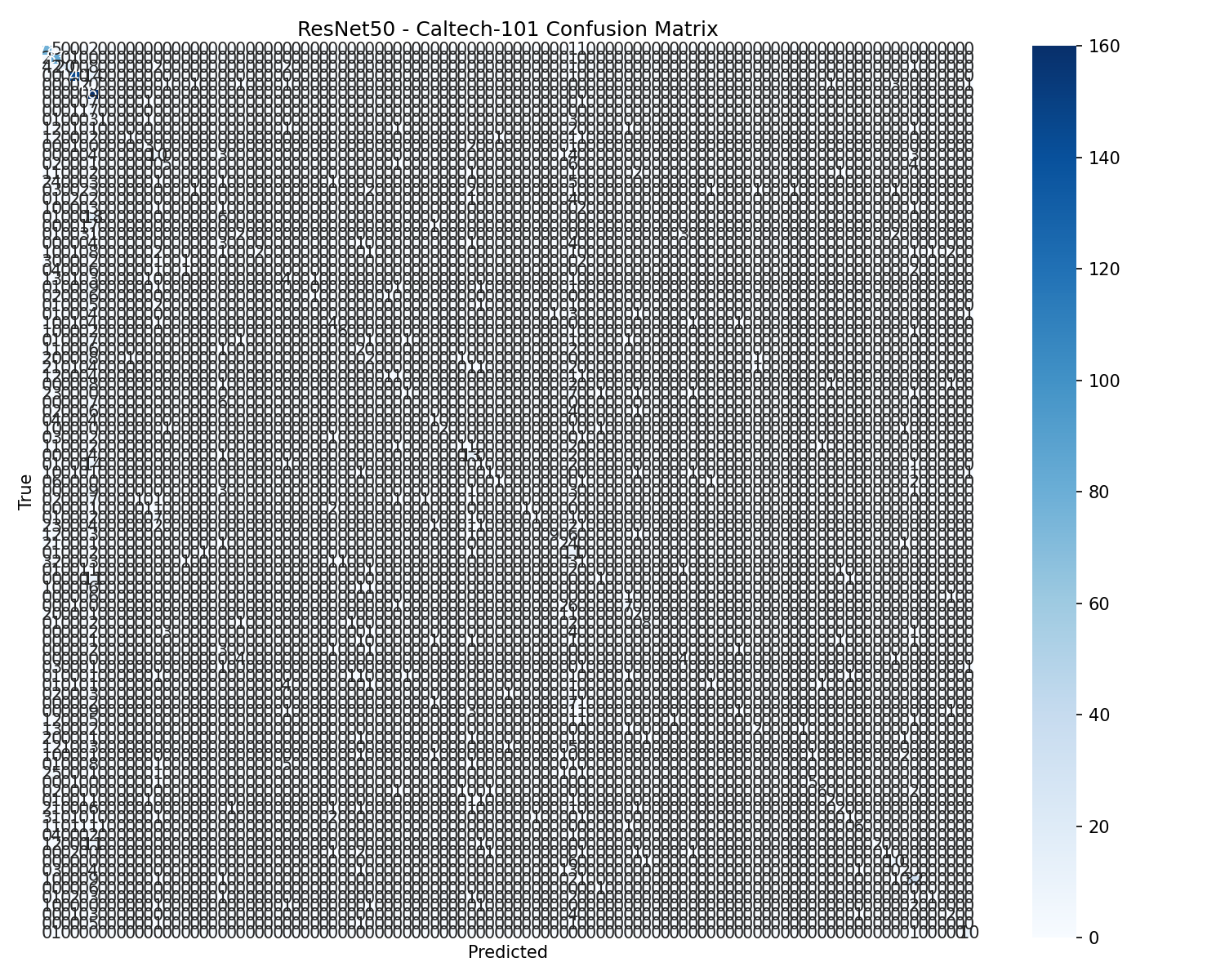
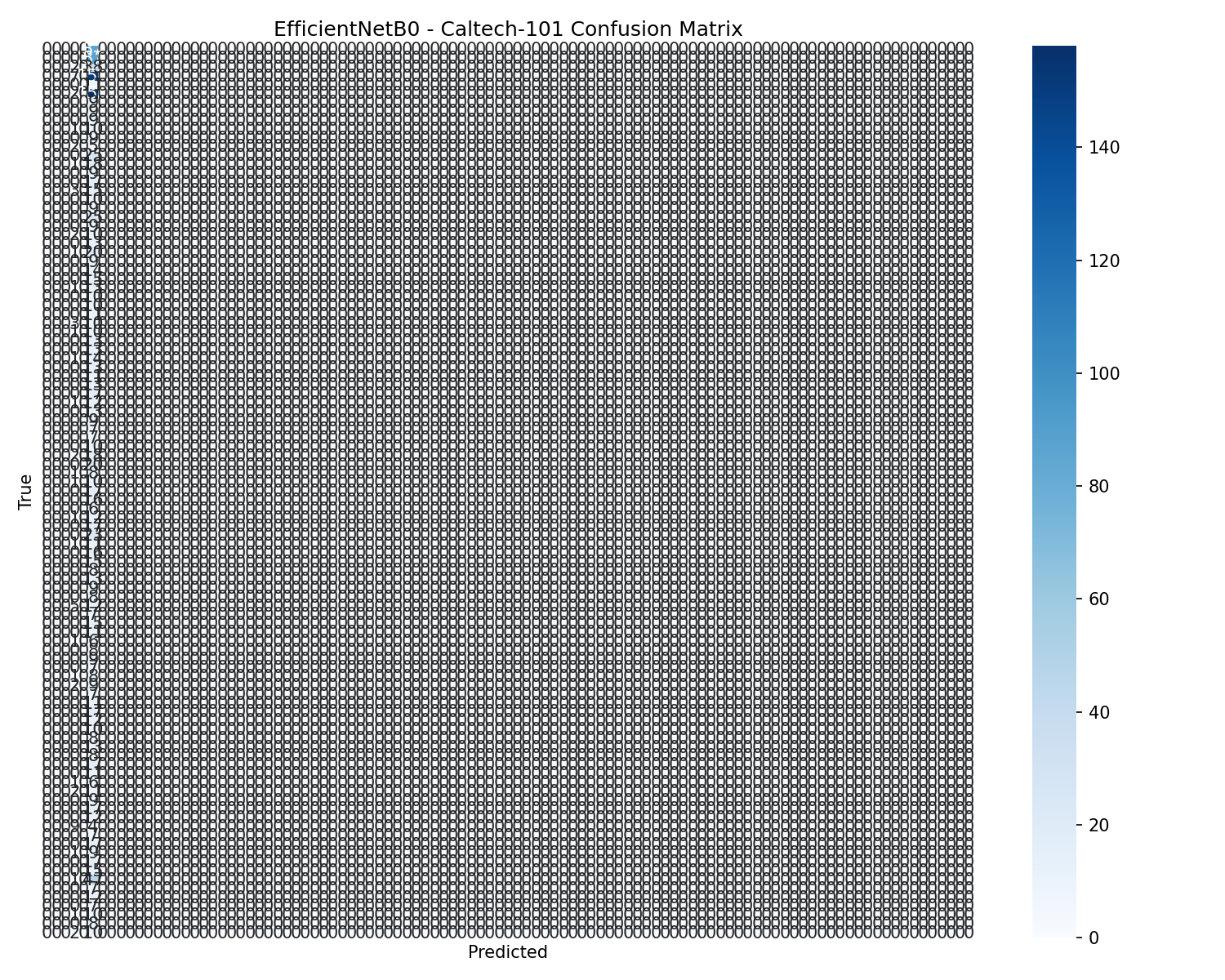
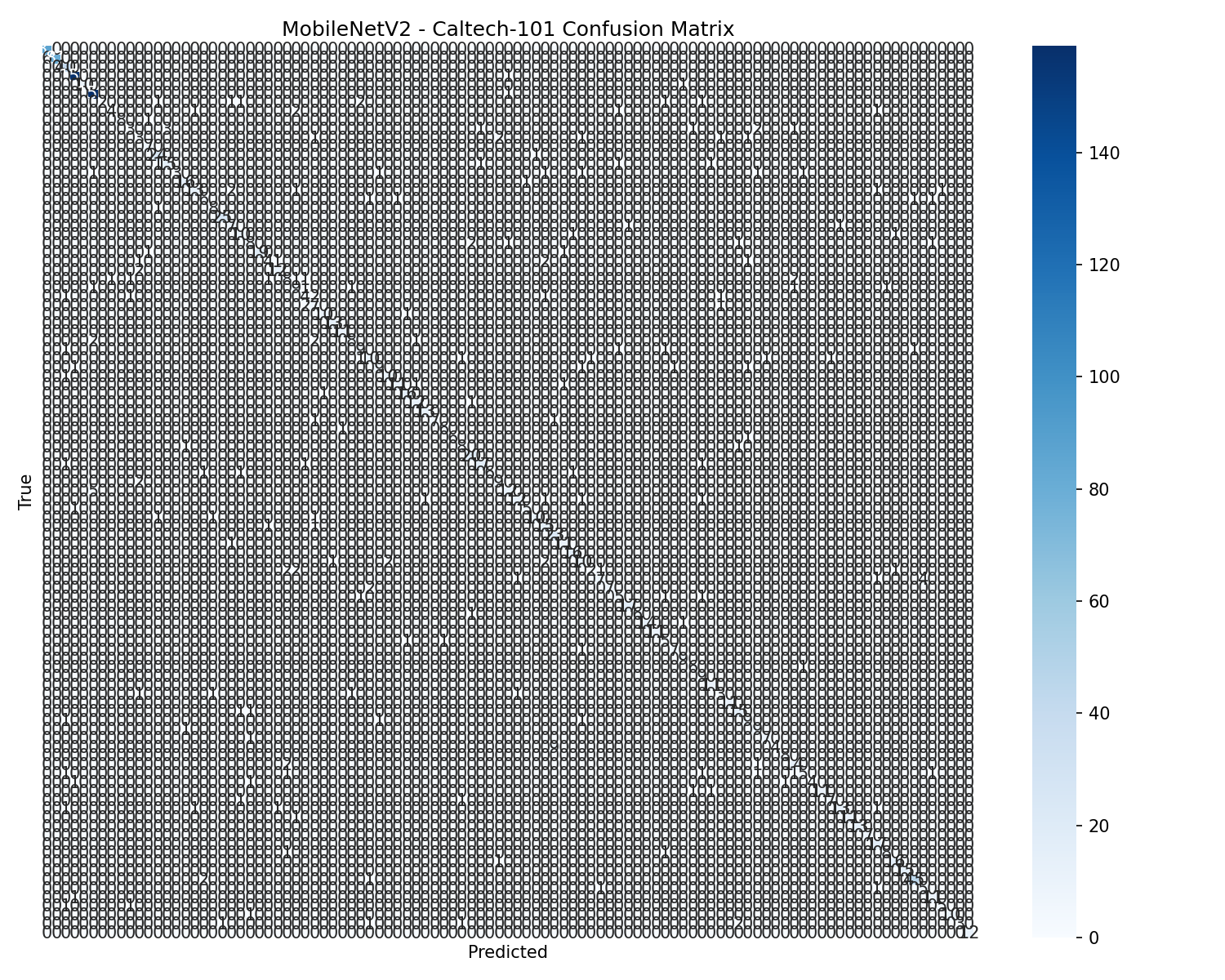
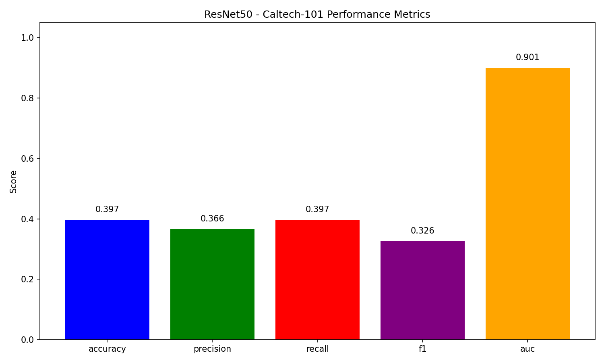
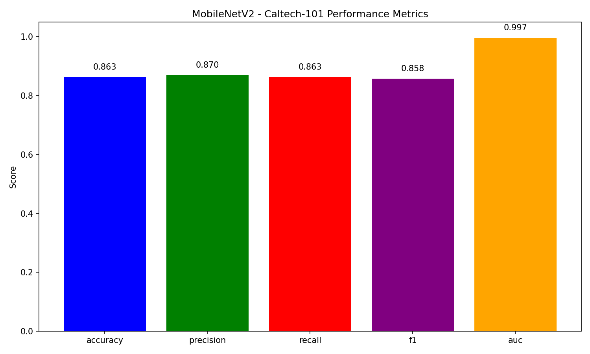
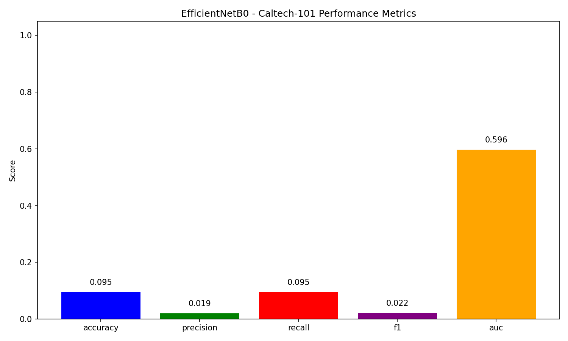
بیشترین خطا : سگ و گربه\_گربه و قورباغه\_قوباغه و پرنده\_هواپیما و کشتی

باید کاری کنیم که بتواند به صورت ریزتر انها را از هم تشخیص دهد.(فاز3)

****

راه های بهبود در این فاز : افزایش دور اموزش (چون هنوز شیب افزایش دقت در val مثبت است)،بهبود لایه ها و معماری و کانولوشن، مقدار دهی بهتر optimizer ( sgd نیاز به نرخ یادگیری دقیق دارد)، استفاده از optimizer دیگر(Nadam)

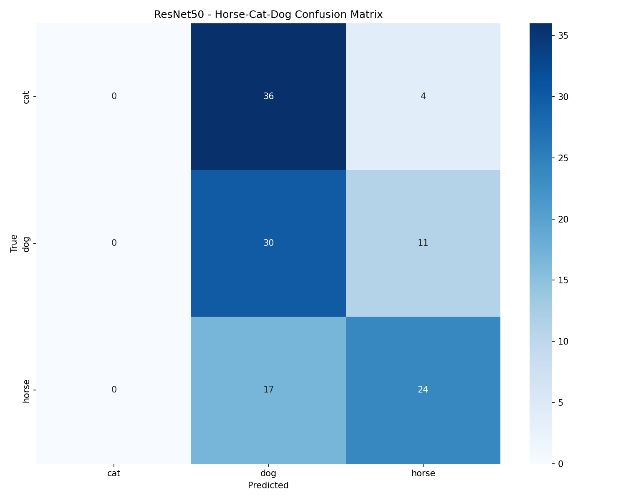
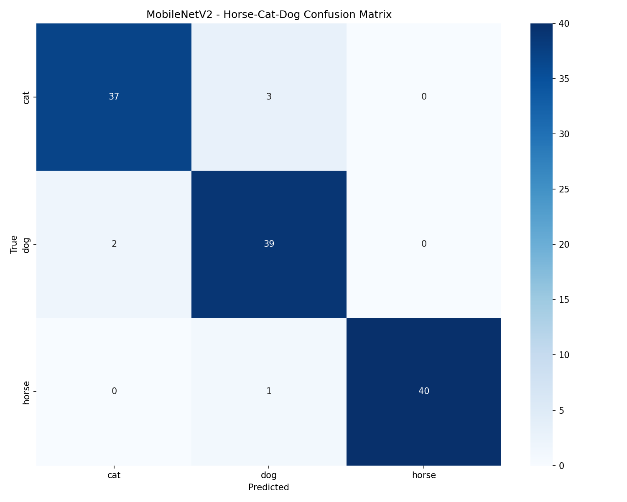
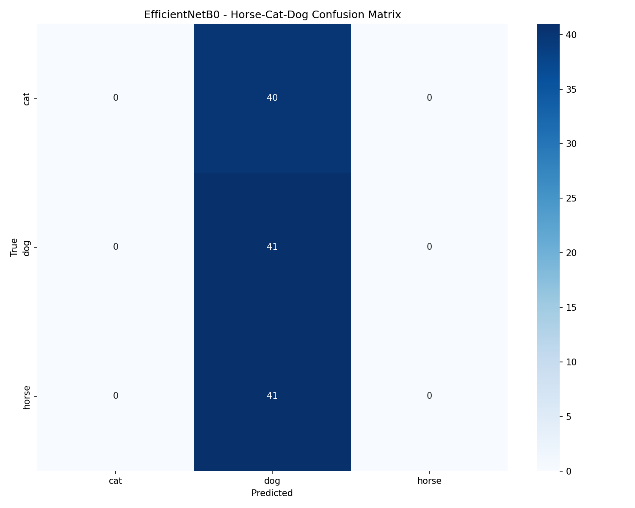
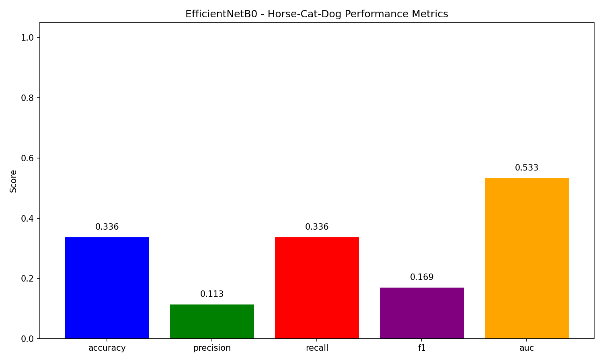
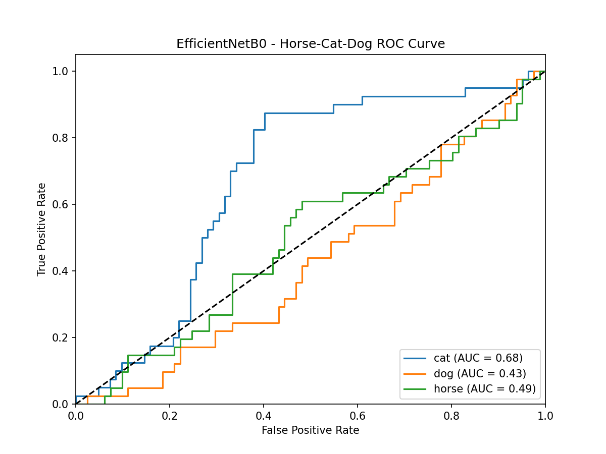
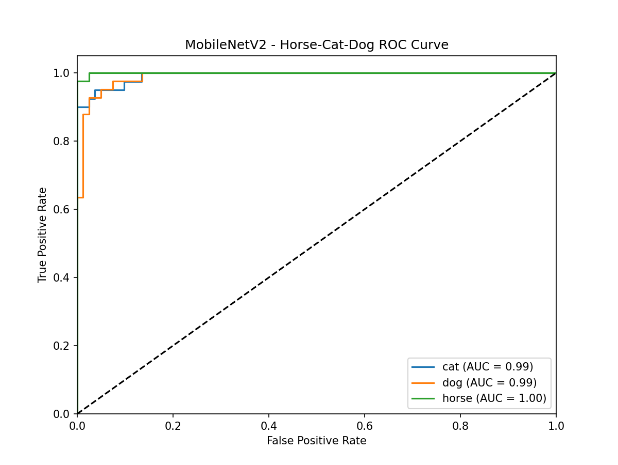
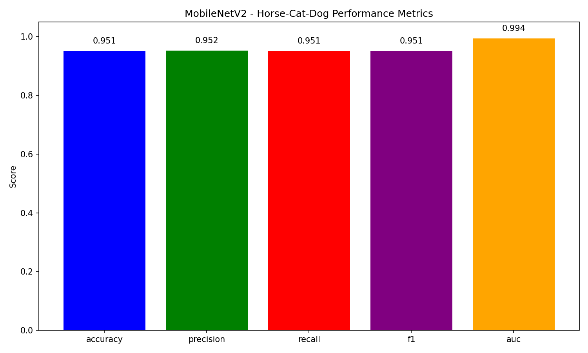
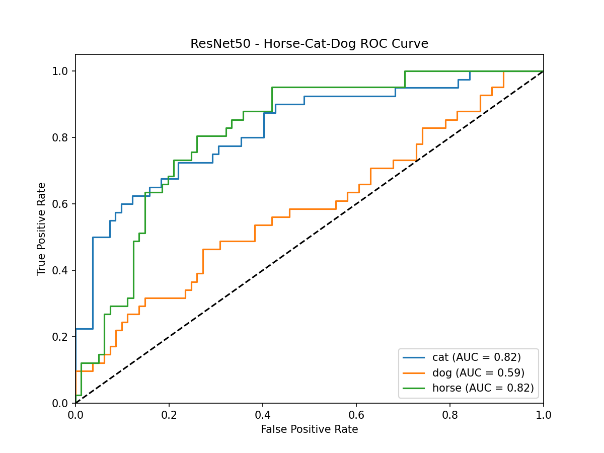
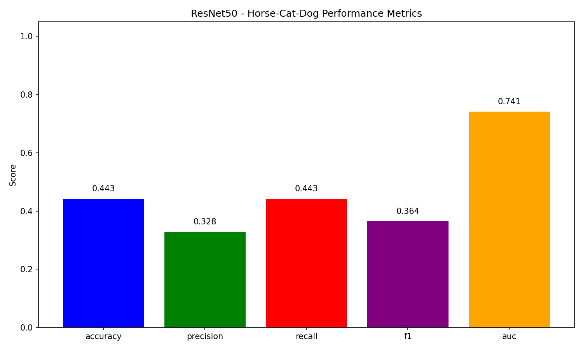
فاز 3-3 :

برای caltech-101

دلایل دقت پایین در restnet50 : بدلیل کم بودن تعداد دور train، (15،10) می باشد در حالی که restnet مدلی بزرگتر با لایه های بیشتر است (حدود 150 لایه)،همچنین کم بودن تعداد لایه های unfreeze، (5لایه)

Mobilevnetv2 به خوبی جواب داده ، اما برای بهبود می توان Augmentation,Hyperparameter Tuningبهتر و بیشتری استفاده کرد (در اینجا بدلیل اینکه می خواستم کل فاز 3-3 را سریع بدست اورم و با ویندوز gpu را ساپورت نمی کرد ، تعداد دور\_Hyperparameter Tuning\_ layer number(5)\_Augmentation همه پایین هستند و روی سرعت تمرکز داشته ام)

Efficientnetb0 عملا هیچ train انجام نمی دهد و دلیل ان هم کم بودن تعداد دور و لایه قابل تغییر می باشد

همان نتیجه گیری ها قبلی را در اینجا نیز می توان گرفت