به نام خدا

دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر

شبکه های عصبی و یادگیری عمیق

مینی پروژه ۱

نام و نام خانوادگی	مرصاد اصالتی
شماره دانشجویی	۸۱۰۱۹۹۰۸۹
تاریخ ارسال گزارش	۱۴۰۰/۰۲/۲۸

فهرست گزارش سوالات

سوال 1 – مفاهیم تئوری

سوال ۱-۱)

الف) مشکلات Gradient Descent

- همگرایی آهسته: در برخی شرایط خاص روش Gradient Descent ممکن است برای همگرایی دچار چالش شود به طور مثال: قرار گرفتن در ناحیه low slope یا وجود پارامتر های sparse و dense به طور هم زمان (پارامتر sparse: پارامتری که میزان گرادیان آن کم باشد. پارامتر dense: پارامتری که میزان گرادیان آن زیاد باشد.)
- گرفتار شدن در بهینه های محلی یا شانه: با توجه به تابع هزینه بکار رفته در آموزش مدل فضای بهینه سازی ممکن است non-convex باشد که در این صورت مدل به یکی از نقاط بهینه محلی همگرا می شود.

ب) توسعه های روش Gradient Descent

Momentum-based Gradient Descent

در این روش مشکل همگرایی آهسته در نواحی low slope را تا حدودی برطرف می کند. به این صورت که با مشاهده تکرار حرکت در یک جهت یکسان در بروزرسانی های متوالی اقدام به برداشتن گام با طول زیاد می کند.

$$w_t = w_{t-1} - m_t$$

$$m_t = \gamma m_{t-1} + \eta \frac{\partial J}{\partial w_{t-1}}$$

AdaDelta

این روش با استفاده از تکنیک adaptive learning rate مشکل مربوط به همگرایی آهسته در حالت وجود پارامتر های sparse و dense را تا حدودی برطرف می کند. به این صورت که در هر مرحله بروزرسانی برای هر یک از پارامتر های مدل مقدار learning rate را با توجه به exponential decaying average مربوط به مربع گرادیان و مربع تغییرات پارامتر تنظیم می کند. (این روش نیاز به تنظیم هیچ پارامتری برای learning rate نمی باشد)

$$w_t = w_{t-1} - \frac{\sqrt{s_{t-1}} + \epsilon}{\sqrt{v_t} + \epsilon} \frac{\partial J}{\partial w_{t-1}}$$

$$v_t = \beta_1 v_{t-1} + (1 - \beta_1) \left(\frac{\partial J}{\partial w_{t-1}} \right)^2$$

$$s_t = \beta_2 s_{t-1} + (1 - \beta_2)(\Delta w_t)^2$$

RMSProp

این روش مشابه روش AdaDelta می باشد با این تفاوت که تنها از exponential decaying average مربوط به مربع گرادیان ها استفاده می کند و دارای پارامتر learning rate می باشد

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{v_t} + \epsilon} \frac{\partial J}{\partial w_{t-1}}$$

$$v_t = \beta_1 v_{t-1} + (1 - \beta_1) \left(\frac{\partial J}{\partial w_{t-1}} \right)^2$$

Adam

این روش ترکیبی از دو روش Momentum-based GD و RMSProp می باشد یا به عبارتی دیگر دارای قابلیت های adaptive learning rate و adaptive momentum به طور همزمان می باشد.

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\left(\frac{\partial J}{\partial w_{t-1}}\right), \ \hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)\left(\frac{\partial J}{\partial w_{t-1}}\right)^2, \ \hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

توسعه های ارائه شده در بالا تا حد خوبی مشکل همگرایی آهسته مربوط به Gradient Descent را برطرف می کند ولی راهکاری برای مشکل همگرایی به بهینه های محلی یا شانه که وابسته به ماهیت تابع هزینه می باشد ارائه نمی دهند.

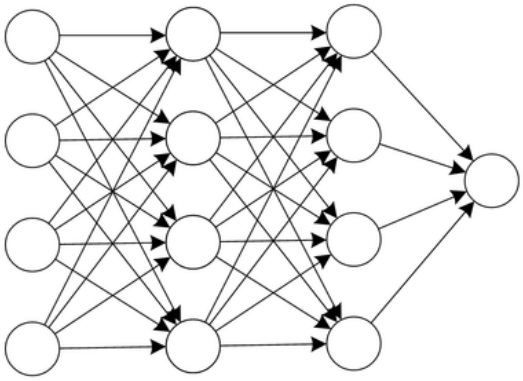
سوال ۱-۲)

زمانی که، با افزایش epoch ها loss داده های آموزشی کم شده و loss داده های ارزیابی از epoch مشخصی به بعد از loss داده های آموزشی فاصله گرفته و افزایش یابد بیش برازش (overfitting) رخ داده است.

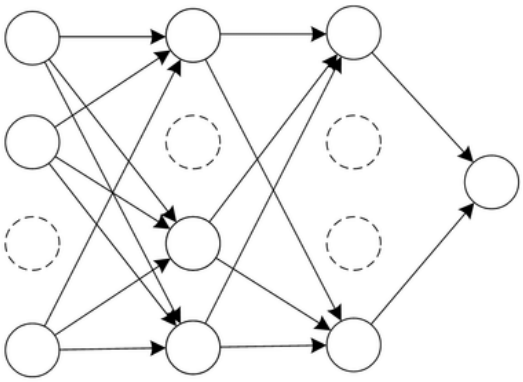
راهکارهایی برای جلوگیری از این مشکل وجود دارد که هر کدام توضیح داده شده اند.

Dropout

در این راهکار یکسری (درصدی) از نورون ها به صورت رندوم در هر iteration حذف شده و در مرحله Feed forward، وزن های آنها در محاسبه مقدار خروجی نقشی نداشته و همچنین در مرحله back-propagation در زمان به روزرسانی وزن ها، وزن هایشان ثابت باقی خواهد ماند. در واقع با این عمل در هر iteration شبکه با نورون های مختلف به روز رسانی شده و این خود، هم ارز یک میانگین گیری از شبکه های مختلف خواهد بود تا تعمیم پذیری مدل را افزایش داده و به نوعی از بیش برازش مدل جلوگیری کند.



(a) Standard Neural Network



(b) Network after Dropout

norm penalty

در این روش سعی بر آن است که با جلوگیری از بزرگ شدن زیاد وزن ها از حساس شدن مدل به برخی از ویژگی ها روی داده های آموزشی اجتناب نموده و همچنین پیچیدگی مدل را کاهش داد. این روش با دو رویکرد L1 Regularization و L2 Regularization انجام می شود که به ترتیب با اضافه شدن 1norm و 2norm به loss function اتفاق خواهد افتاد.

L1 Regularization

هدف از رویکرد کاهش قدر مطلق وزن ها است

$$L(x,y) \equiv \sum_{i=1}^{n}(y_i - h_\theta(x_i))^2 + \lambda \sum_{i=1}^{n}|\theta_i|$$

L2 Regularization

هدف از رویکرد کاهش مجذور وزن هاست

$$L(x,y) \equiv \sum_{i=1}^{n} (y_i - h_\theta(x_i))^2 + \lambda \sum_{i=1}^{n} \theta_i^2$$

L1 Regularization	L2 Regularization
1. L1 penalizes sum of absolute values of weights.	1. L2 penalizes sum of square values of weights.
2. L1 generates model that is simple and interpretable.	2. L2 regularization is able to learn complex data patterns.
3. L1 is robust to outliers.	3. L2 is not robust to outliers.

انتخاب بین این دو رویکرد بستگی به مسئله و داده ها دارد. اگر داده ها پیچیده باشند رویکرد 2L و اگر داده ساده باشند 1L بهتر خواهد بود. نکته دیگر اینکه رویکرد 1L به داده های پرت حساس نیست.

early stopping

در این روش آموزش روی داده های آموزشی را تا جایی پیش می بریم که loss داده های آموزشی و ارزیابی در حال کم شدن باشند و اگر loss داده های ارزیابی شروع به فاصله گرفتن از داده های آموزشی کرد و افزایش یافت فرآیند آموزش را متوقف می کنیم. این عمل موجب خواهد شد که با یافتن epoch بهینه برای مدل، روی داده های ارزیابی هم دقت خوبی به دست آورده شود.

سوال ۱-۳)

همانطور که می دانیم شبکه عصبی با دو لایه ی مخفی دارای قابلیت general function approximation می باشد یا به عبارتی دیگر پتانسیل مدلسازی هر تابع دلخواه را دارد. اما در برخی مسائل داده ها دارای disturbance بوده و یا بین ابعاد مختلف بردار ویژگی رابطه correlation وجود دارد که باعث دشوار شدن فرایند partitioning یا mapping می شود. به همین منظور با اضافه کردن تعداد لایه های مخفی شبکه به شبکه قابلیت استخراج ویژگی و کاهش بعد را اضافه می کنیم تا با بدست آوردن بازنمایی مناسب از داده ها عملیات partitioning یا mapping مد نظر را با کیفیت مناسب انجام دهد.

سوال ۱-۴)

در طراحی شبکه های عصبی عمیق بهتر است از اصل پارسیمونی استفاده کنیم، به طوری که ابتدا شبکه را با تعداد لایه ها و نورون های کمتر ساخته و بعد از به دست آوردن نتیجه، تا جایی که عملکرد شبکه هنوز افت نکرده است تعداد نورون و تعداد لایه های میانی را افزایش دهیم. البته افزایش بیش از حد لایه ها و نورون ها، پارامتر های شبکه را افزایش داده، و برای اینکه همه ی پارامتر ها طوری تنظیم شوند که بتوانند روی داده های ارزیابی تعمیم پذیری بالایی داشته باشیم باید تعداد داده های زیادی داشته باشیم، در غیر اینصورت مشکل overfitting رخ خواهد داد و همچنین زمان آموزش مدل با توجه به افزایش تعداد پارامتر ها افزایش خواهد یافت.

سوال ۱-۵)

در صورت مقدار دهی اولیه وزن ها با مقدار ثابت مشتق جزئی تابع هزینه نسبت به وزن های هر لایه یکسان می شود. در نتیجه وزن های بروزرسانی شده در هر لایه نیز یکسان باقی می مانند. این موضوع باعث می شود که تمام نورون های مربوط به یک لایه تنها یک پردازش یکسان را روی خروجی لایه ی قبل انجام دهند که این مشکل باعث کاهش چشمگیر قدرت مدل می شود و اصلا برای شبکه مطلوب نیست.

در صورت مقدار دهی اولیه وزن ها با مقادیر تصادفی بسیار کوچک خروجی تابع فعال ساز (sigmoid) به شدت به صفر نزدیک می شود. در نتیجه مقدار مشتق جزئی بدست آمده برای پارامتر ها به سمت صفر میل می کند. این موضوع باعث مشکل Gradient Vanishing شده و فرایند آموزش را دچار مشکل می کند. همین مشکل برای مقدار دهی اولیه وزن ها با مقادیر تصادفی بسیار بزرگ نیز وجود دارد.

مقدار دهی اولیه مناسب برای وزن ها باید به صورتی باشد که میانگین خروجی هر لایه از شبکه صفر و واریانس خروجی تمام لایه ها یکسان باشد به طور مثال مقدار دهی اولیه با استفاده از روش Xavier Initialization

$$\sqrt{\frac{1}{size^{[l-1]}}}$$

$$W^{[l]} = np.random.randn(size\_l, size\_l\text{-}1) * np.sqrt(1/size\_l\text{-}1)$$

سوال ۱-۶)

زمانی که ضرب مشتقات توابع فعالساز اعمال شده در فرآیند back-propagation بزرگ باشد، مقدار گرادیان به صورت نمایی تا رسیدن به لایه های اولیه بزرگ خواهد شد تا جایی که به اصطلاحی منفجر شده و به بی نهایت برسد. به این معضل در شبکه های عصبی exploding gradient می گویند.

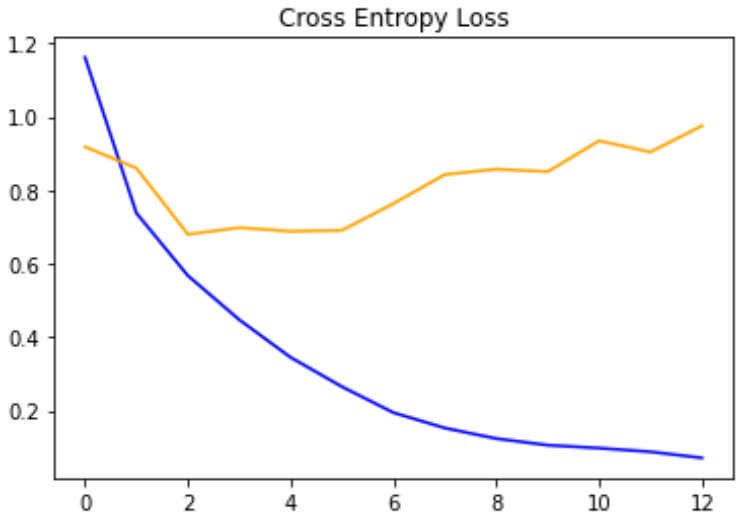
اما زمانی که ضرب این مشتقات در هم کوچک شود، مقدار گرادیان هم تا رسیدن به لایه های اولیه بسیار ناچیز باقی خواهد ماند تا جایی که وزن های لایه های اولیه آپدیت نشده و ثابت باقی می مانند، به این پدیده vanishing gradient می گویند.

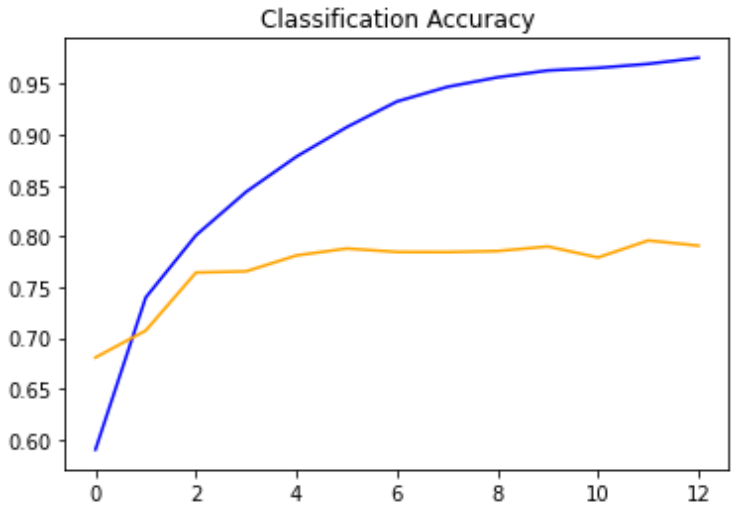
سوال 2 – CNN

سوال ۲-۱)

- شبکه ساخته شده شامل سه لایه CNN و یک لایه fully connected میانی و یک لایه fully connected خروجی است، هر لایه CNN از دو لایه کانولوشن دو بعدی با سایز پنجره 3 در 3 و stride (گام) 1 در 1 و یک لایه max-pooling با سایز پنجره 2 در 2 می باشد،که لایه های کانولوشنی لایه اول 32 نورون، لایه دوم 64 نورون و لایه سوم 128 نورون دارند.
- تابع استفاده شده در همه لایه های میانی تابع relu و در لایه آخر softmax می باشد.
- تعداد نورون لایه fully connected میانی 128 و لایه آخر 10 می باشد.
- تابع loss مورد استفاده تابع categorical_crossentropy و از روش adam برای بهینه سازی استفاده شده است.
- اندازه mini-batch برابر 64 بوده، یعنی رابطه بروزرسانی وزن ها به ازای هر 64 نمونه داده های آموزشی آپدیت می شود.

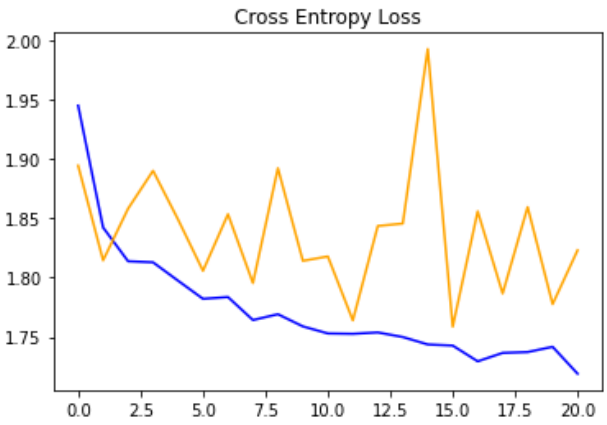
سوال ۲-۲)

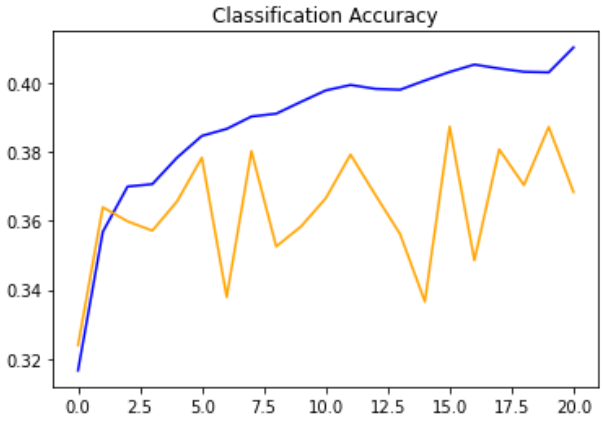




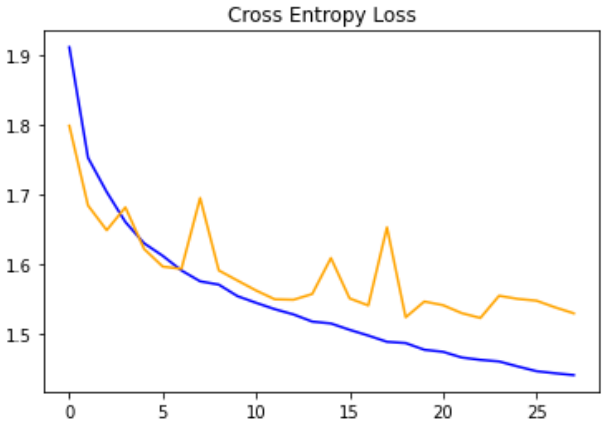
سوال ۲-۳)

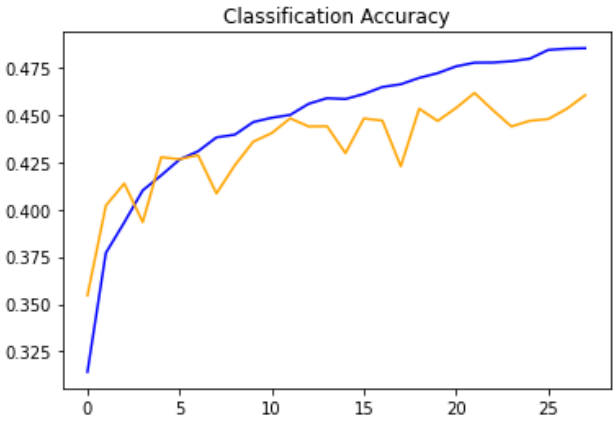
طراحي مدل با صفر لايه مخفي



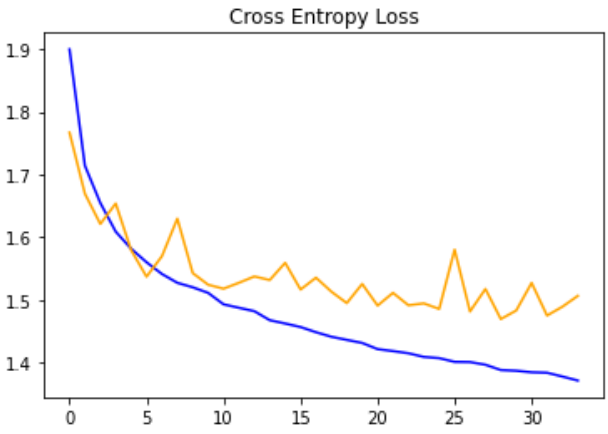


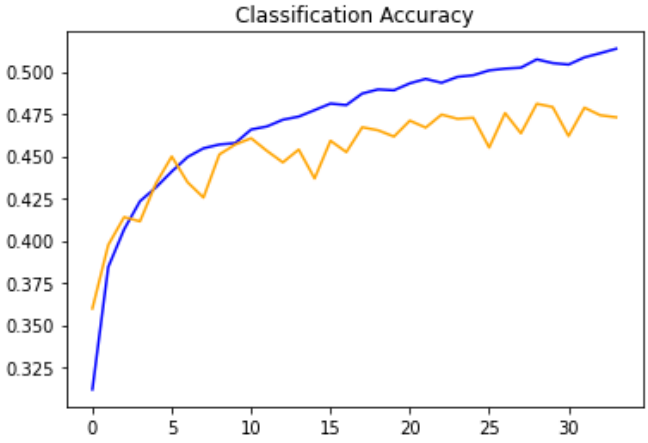
طراحي مدل با يك لايه مخفي





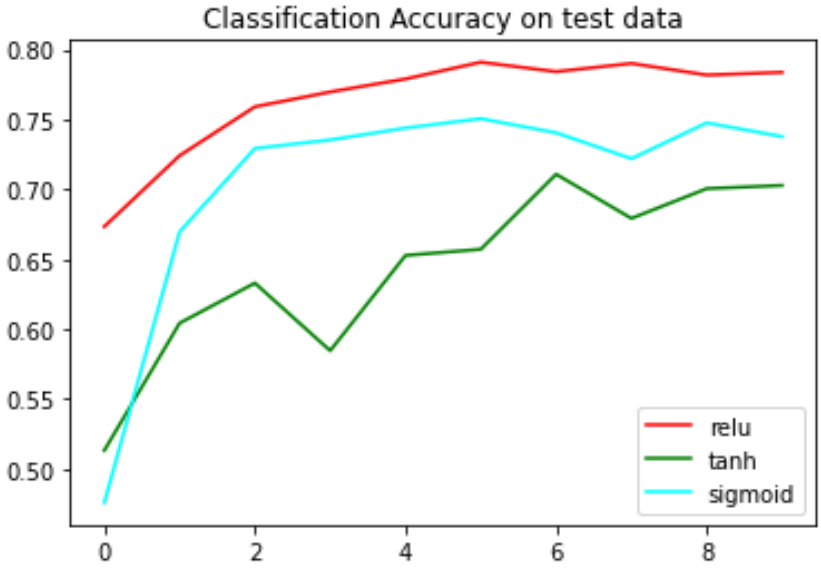
طراحي مدل با دو لايه مخفي





با توجه به نمودار هاي دقت و خطاي شبكه كه كارايي شبكه را نشان مي دهند، مشاهده مي شود با افزايش لايه هاي مياني از صفر لايه تا دو لايه، دقت شبكه افزايش و خطاي آن كاهش يافته است.

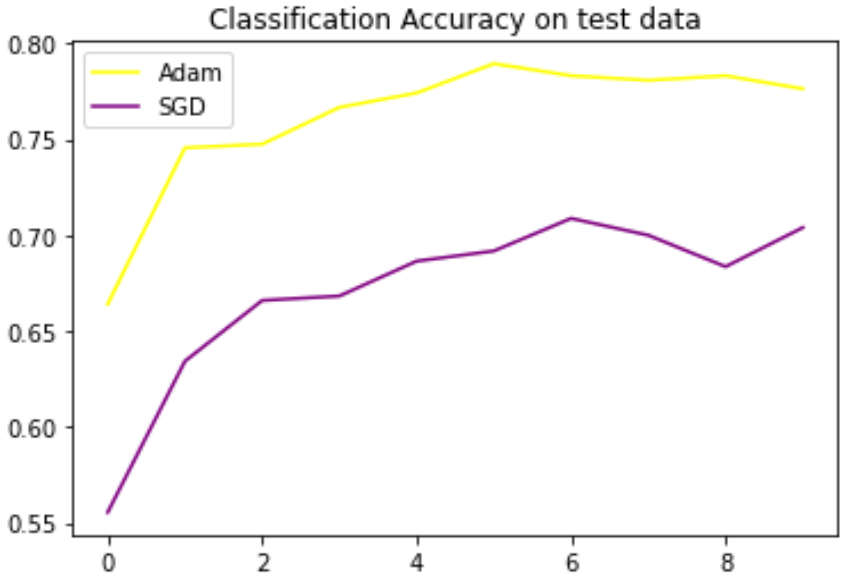
سوال ۲-۴)



دقت شبكه با تابع فعال ساز relu بهتر از sigmoid و تابع فعال ساز sigmoid بهتر از tanh بوده است. چون مشتق تابع relu براي مقادير مثبت يك بوده ضرب مشتقات جزيي در هم در عمليات back-propagation كوچك نشده به همين خاطر پديده vanishing gradient اتفاق نخواهد

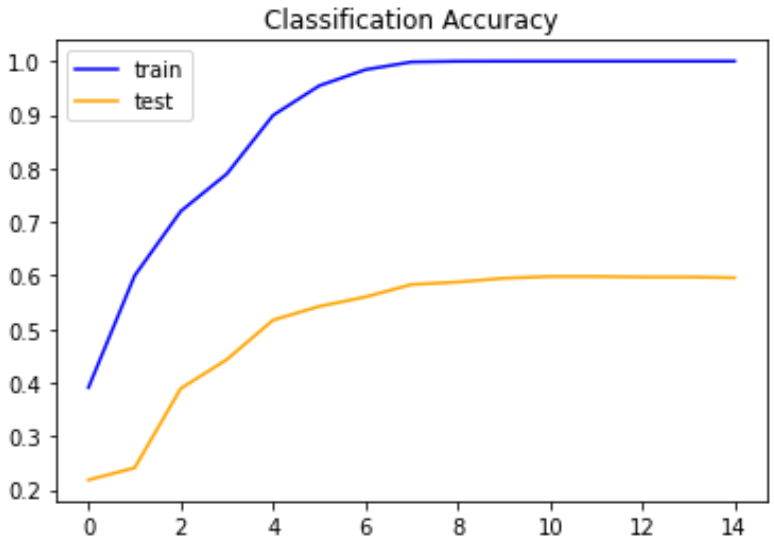
افتاد و چون مشتق تابع relu برا مقادیر منفی صفر می شود وزن های همه نورون ها در رابطه بروزرسانی وزن ها آپدیت نشده که سبب می شود سرعت شبکه هم بالاتر برود.

سوال ۲-۵)



همانطور که در شکل بالا مشاهده می شود روش بهینه سازی adam نسبت به sgd دقت های بالاتری در همه ی epoch ها به دست آورده است. بهینه ساز adam چون از Adaptive learning rate و همچنین Adaptive momentum استفاده می کند سریعتر خطای آن کاهش پیدا کرده و زودتر به نقطه بهینه همگرا می شود در نتیجه در هر epoch دقت آن روی داده های آموزشی نسبت به sgd بیشتر شده است.

سوال ۲-۶)

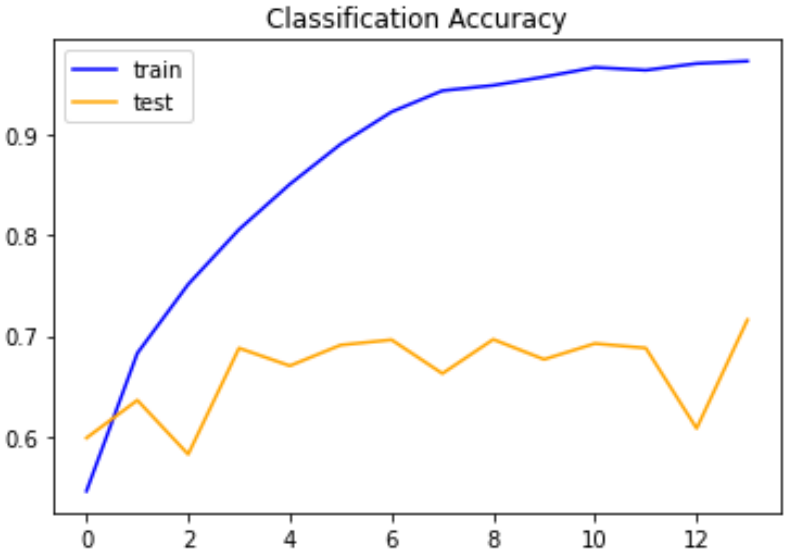


چون تعداد داده های آموزشی از 50000 به 6000 کاهش یافته است، دقت روی داده های ارزیابی نیز به دلیل کمبود مشاهدات کاهش یافته و حتی مدل در epoch چهاردهم به بعد دچار overfitting شده که به وسیله امکان early stopping از ادامه آن جلوگیری شده است که این نشان می دهد با کاهش داده ها overfitting هم سریعتر اتفاق خواهد افتاد.

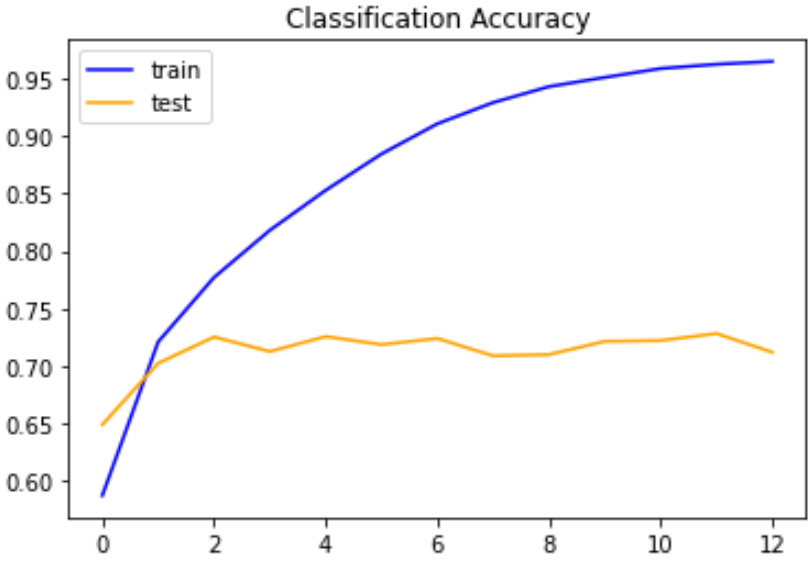
سوال ۲-۷)

شبکه طراحی شده با یک لایه کانولوشن در هر لایه CNN با کرنل سایز بزرگتر (7x7) نسبت به شبکه با ساختار دو لایه کانولوشن در هر لایه CNN عملکرد ضعیف تری داشته، که نتیجه گرفته می شود عمیق تر کردن مدل (اضافه کردن لایه کانولوشنی) بهتر از پهن کردن مدل (افزایش سایز پنجره) برای افزایش کارایی شبکه می باشد.

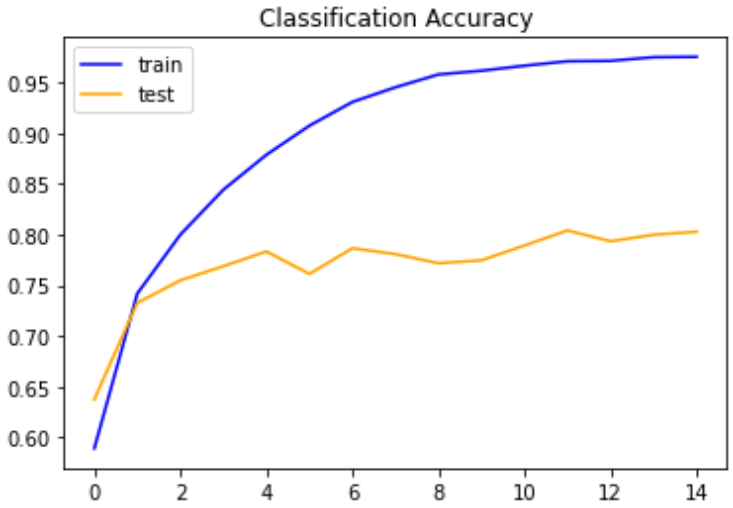
نمودار دقت برای داده های آموزشی و تست با یک لایه کانولوشنی با کرنل سایز 7x7:



نمودار دقت برای داده های آموزشی و تست با یک لایه کانولوشنی با کرنل سایز 3x3:



نمودار دقت برای داده های آموزشی و تست با دو لایه کانولوشنی با کرنل سایز 3x3:



همانطور که پیداست و توضیح داده شد دقت در شبکه با دو لایه کانولوشنی در هر لایه CNN بهتر شده زیرا در لایه اول کانولوشن فیچر های ریزتر را تشخیص داده (مانند خط ها و لبه های موجود در شکل و ...) و وقتی لایه دوم کانولوشن اضافه شده با ترکیب با لایه اول کانولوشن، فیچر های درشت تر تصاویر تشخیص داده می شوند. (مانند اشکال و ...)

سوال 3 – Data Augmentation

سوال ۳-۱)

مدل های Deep Learning وظیفه ی انجام task های پیچیده را برعهده داشته و متناسب با پیچیدگی این task ها تعداد پارامتر های قابل یادگیری آنها افزایش یافته است. برای اینکه در طی فرآیند یادگیری پارامتر های مدل به درستی تنظیم شوند نیاز به حجم زیادی از داده های آموزشی با تنوع کافی هستیم. اما همیشه حجم داده زیادی در اختیار نداشته و مجبور به استفاده از تکنیک Data Augmentation برای باز تولید تصاویر جدید از روی تصاویر اولیه هستم.

از طرفی همواره از تکنیک Data Augmentation تنها برای افزایش حجم داده ها استفاده نمی کنیم گاهی اوقات با افزایش تنوع داده ها به دنبال افزایش robustness مدل و جلوگیری از توجه مدل به irrelevant feature ها هستیم.

عملیات Data Augmentation به دو صورت قابل انجام است:

- offline augmentation: در این روش تمام داده های آموزشی را به صورت یکجا augment می کنیم و در کنار داده های اولیه قرار می دهیم. (مناسب برای دیتاست های کوچک)
- online augmentation: در این روش تنها داده های مربوط به یک mini batch را augment کرده و پس از اتمام آموزش به روی mini batch داده ها را از حافظه پاک می کنیم. (مناسب برای دیتاست های بزرگ)

تکنیک های Data Augmentation برای image:

- flip: بازتاب تصویر به صورت افقی یا عمودی
- rotation: چرخش تصویر با زاویه ی دلخواه
- scale: بزرگنمایی یا کوچک نمایی تصویر
- crop: گرفتن برشی تصادفی از تصویر
- transition: انتقال تصویر در جهت افقی یا عمودی یا هر دو
- noise: افزودن gaussian noise یا salt and pepper noise یا ...
- color: تغییر ترکیب رنگی تصویر
- brightness: تغییر میزان روشنایی تصویر
- ...

نکته: برای انجام عملیات Data Augmentation از ترکیبی از تکنیک های فوق به صورت pipeline استفاده می شود. علاوه بر تکنیک های عمومی ارائه شده با توجه به مدل و task مد نظر می توانیم از تکنیک های پیشرفته تر مانند cGAN یا Style Transfer نیز استفاده کرد.

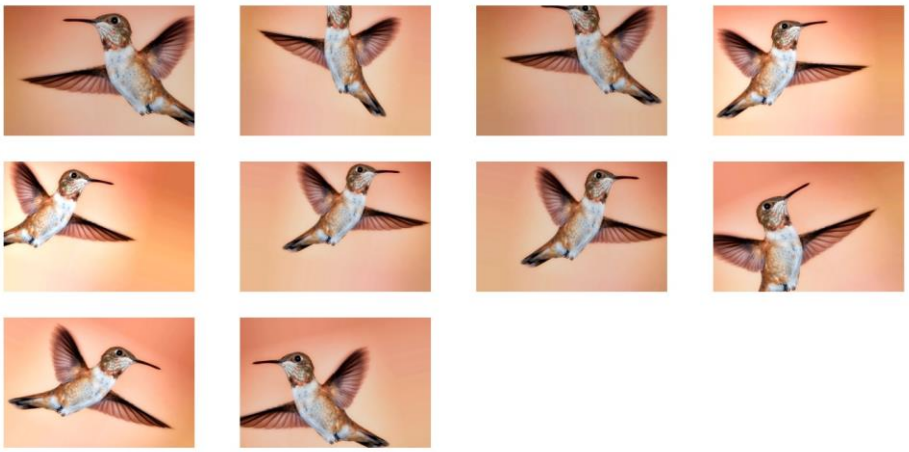
نکته: خروجی نهایی نیز باید ابعادی مشابه تصویر اصلی داشته باشد که در برخی موارد مانند rotation یا کوچک نمایی نیاز به interpolation برای پر کردن حاشیه های بوجود آمده داریم.

نکته: عملیات Data Augmentation تنها به روی داده های آموزشی انجام می شود و برای ارزیابی مدل تنها از داده های اصلی استفاده می کنیم.

سوال ۳-۲)

با استفاده از تکنیک های زیر اقدام به تولید تصاویر جدید از روی تصویر اصلی می کنیم:

- rotation با بازه ی 0 تا 30 درجه
- transition افقی با نهایت اندازه ۲۰ درصد پهنای تصویر اصلی
- transition عمودی با نهایت اندازه ۲۰ درصد ارتفاع تصویر اصلی
- brightness با بازه ی 0.85 تا 1.15 نسبت به تصویر اصلی
- scale با بازه ی 0.85 تا 1.15 نسبت به تصویر اصلی



سوال ۳-۳)

در این قسمت ابتدا به طور تصادفی ۹۰ درصد داده های مربوط به کلاس cat و dog را از مجموعه داده های آموزشی حذف کرده و اقدام به آموزش مدل می کنیم.

تعداد کل داده ها بعد از حذف :

```
Training Data: 41000 32*32*3
Test Data: 10000 32*32*3
```

حال با استفاده از معماری بدست آمده در سوال ۲ اقدام به آموزش مدل می کنیم:

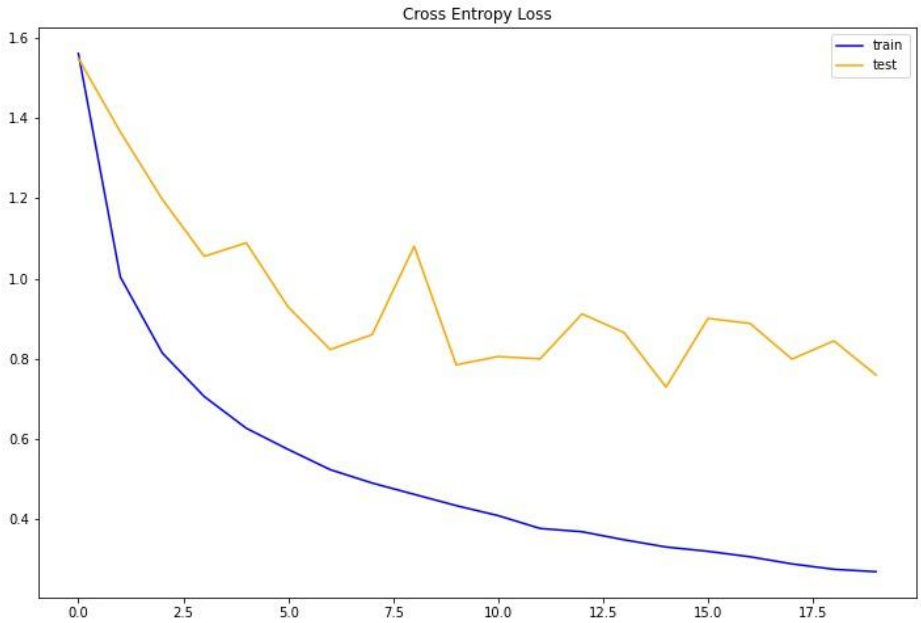
Model: "sequential"

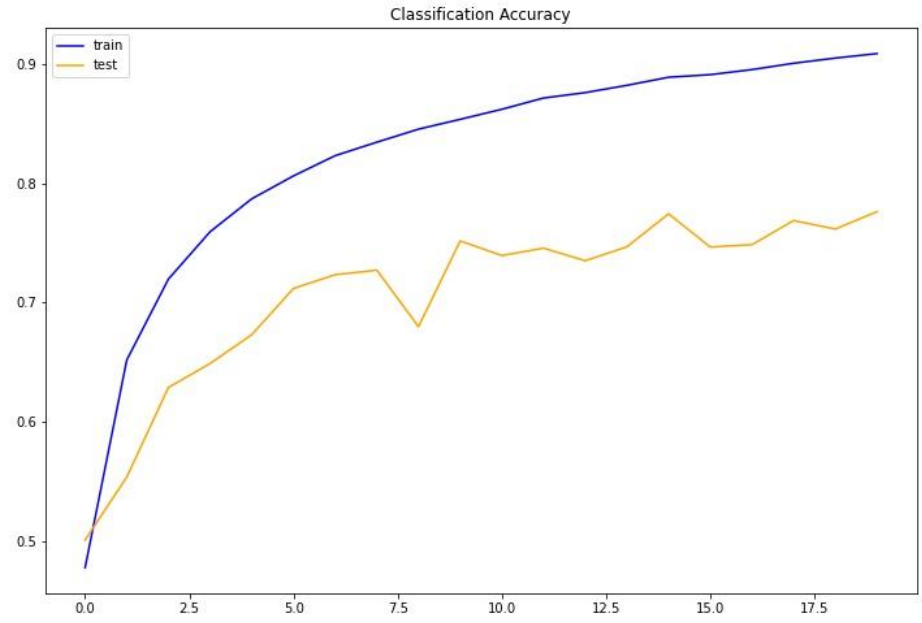
Layer (type)	Output Shape	Param #
conv2d_7 (Conv2D)	(None, 32, 32, 32)	896
batch_normalization_7 (Batch	(None, 32, 32, 32)	128
conv2d_8 (Conv2D)	(None, 32, 32, 32)	9248
batch_normalization_8 (Batch	(None, 32, 32, 32)	128
max_pooling2d_3 (MaxPooling2	(None, 16, 16, 32)	0
dropout_4 (Dropout)	(None, 16, 16, 32)	0
conv2d_9 (Conv2D)	(None, 16, 16, 64)	18496

```
batch_normalization_9 (Batch (None, 16, 16, 64)        256
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 16, 16, 64)        36928
_________________________________________________________________
batch_normalization_10 (Batc (None, 16, 16, 64)        256
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 8, 8, 64)          0
_________________________________________________________________
dropout_5 (Dropout)          (None, 8, 8, 64)          0
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 8, 8, 128)         73856
_________________________________________________________________
batch_normalization_11 (Batc (None, 8, 8, 128)         512
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 8, 8, 128)         147584
_________________________________________________________________
batch_normalization_12 (Batc (None, 8, 8, 128)         512
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 4, 4, 128)         0
_________________________________________________________________
dropout_6 (Dropout)          (None, 4, 4, 128)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2048)              0
_________________________________________________________________
dense_2 (Dense)              (None, 128)               262272
_________________________________________________________________
batch_normalization_13 (Batc (None, 128)               512
_________________________________________________________________
dropout_7 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                1290
=================================================================
Total params: 552,874
Trainable params: 551,722
Non-trainable params: 1,152
_________________________________________________________________
```

عملیات آموزش را با روش بهینه سازی Adam و با مقدار learning rate=0.001 و استفاده از تکنیک early stopping با patience=5 انجام می دهیم.

نمودار های آموزش

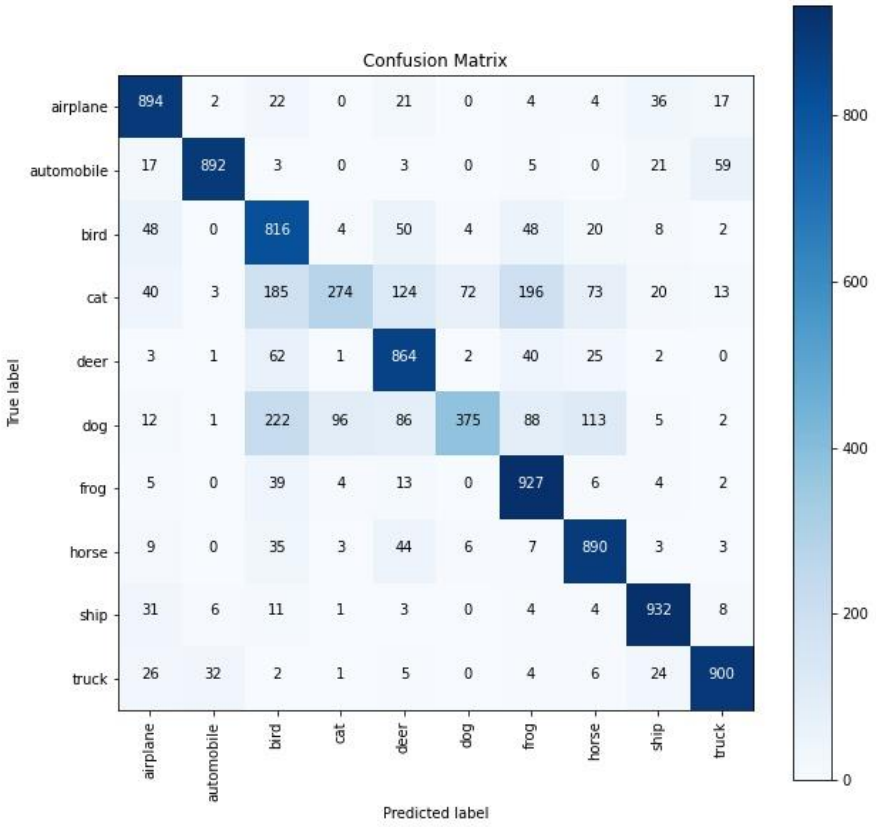




همانطور که در نمودار های فوق مشاهده می شود مدل در نهایت پس از طی کردن تقریبا 20 تا epoch به میزان دقت و خطای مناسبی همگرا می شود.

ارزیابی مدل

	precision	recall	f1-score	support
airplane	0.82	0.89	0.86	1000
automobile	0.95	0.89	0.92	1000
bird	0.58	0.82	0.68	1000
cat	0.71	0.27	0.40	1000
deer	0.71	0.86	0.78	1000
dog	0.82	0.38	0.51	1000
frog	0.70	0.93	0.80	1000
horse	0.78	0.89	0.83	1000
ship	0.88	0.93	0.91	1000
truck	0.89	0.90	0.90	1000
accuracy			0.78	10000
macro avg	0.79	0.78	0.76	10000
weighted avg	0.79	0.78	0.76	10000



مشاهده می شود که علیرغم اینکه مدل توانسته به دقت مطلوبی دست پیدا کند اما وقتی به عملکرد مدل برای هر یک از کلاس ها به تنهایی توجه می کنیم کاملا واضح است مدل در طبقه بندی داده های مربوط به دو کلاس cat و dog کاملا ضعیف عمل کرده است. به عبارتی دیگر

کمبود تعداد داده ی آموزشی کافی در این دو کلاس منجر به ناتوانی مدل در طبقه بندی این دو کلاس شده است.

سوال ۳-۴)

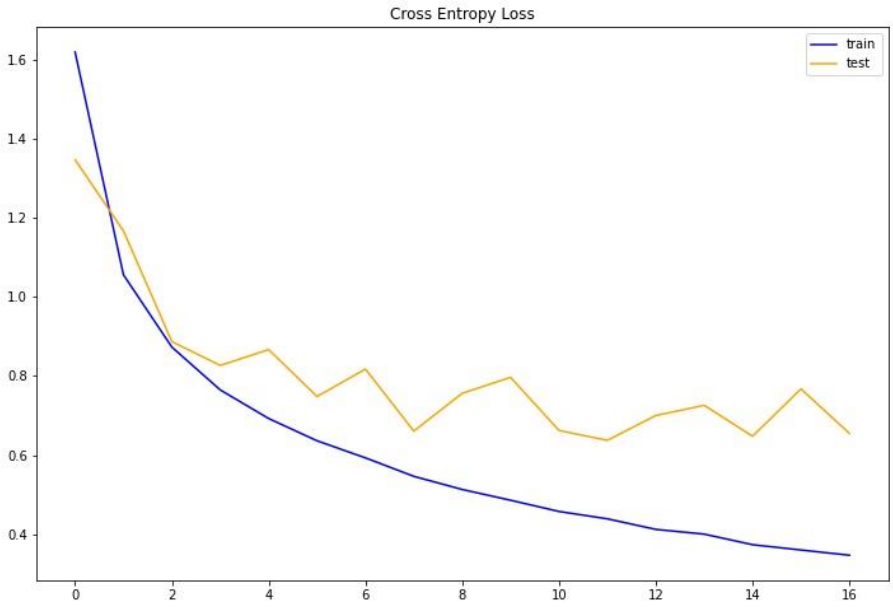
حال با استفاده از تکنیک Data Augmentation سعی می کنیم با تولید تصاویر مصنوعی از دو کلاس cat و dog تعداد داده های آموزشی این دو کلاس را با سایر کلاس ها برابر کنیم . سپس با استفاده از معماری بدست آمده در سوال ۲ مجددا اقدام به ساخت طبقه بند می کنیم.

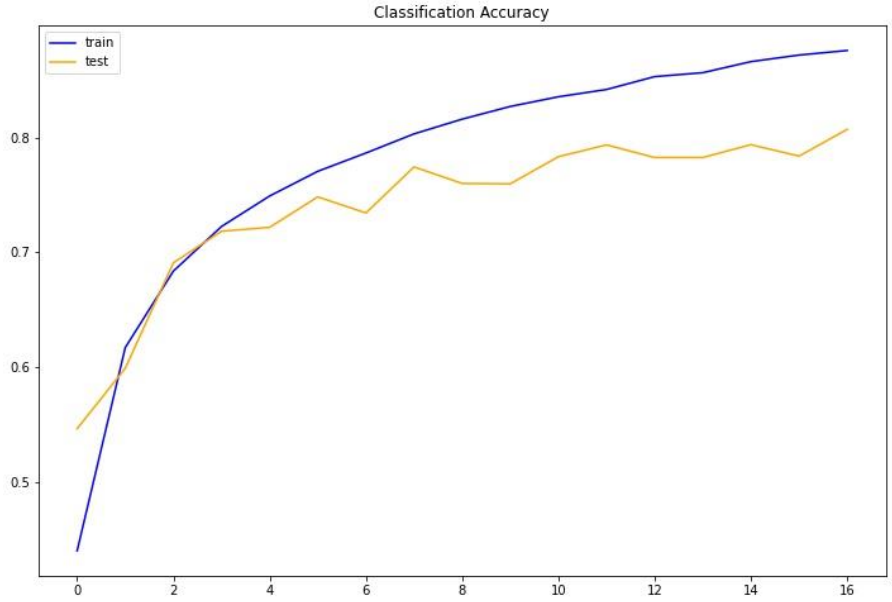
تکنیک های augmentation مورد استفاده:

- rotation با بازه ی 0 تا 30 درجه
- transition افقی با نهایت اندازه ۲۰ درصد پهنای تصویر اصلی
- transition عمودی با نهایت اندازه ۲۰ درصد ارتفاع تصویر اصلی
- brightness با بازه ی 0.85 تا 1.15 نسبت به تصویر اصلی
- scale با بازه ی 0.85 تا 1.15 نسبت به تصویر اصلی

عملیات آموزش را با روش بهینه سازی Adam و با مقدار learning rate=0.001 و استفاده از تکنیک early stopping با patience=5 انجام می دهیم.

نمودار های آموزش

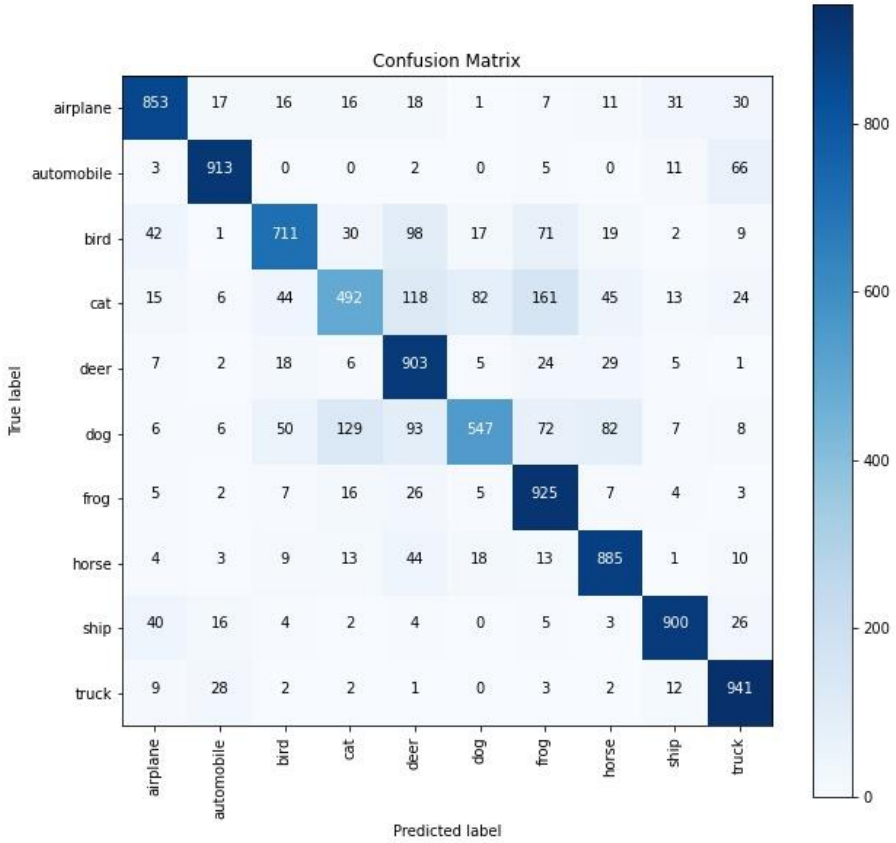




همانطور که در نمودار های فوق مشاهده می شود مدل در نهایت پس از طی کردن تقریبا 16 تا epoch به میزان دقت و خطای مناسبی همگرا می شود.

ارزیابی مدل

	precision	recall	f1-score	support
airplane	0.87	0.85	0.86	1000
automobile	0.92	0.91	0.92	1000
bird	0.83	0.71	0.76	1000
cat	0.70	0.49	0.58	1000
deer	0.69	0.90	0.78	1000
dog	0.81	0.55	0.65	1000
frog	0.72	0.93	0.81	1000
horse	0.82	0.89	0.85	1000
ship	0.91	0.90	0.91	1000
truck	0.84	0.94	0.89	1000
accuracy			0.81	10000
macro avg	0.81	0.81	0.80	10000
weighted avg	0.81	0.81	0.80	10000



با توجه به نتایج بدست آمده مشاهده می شود استفاده از تکنیک Data Augmentation باعث می شود عملکرد مدل برای دو کلاس cat و dog نسبت به حالت قبلی بهتر شود ولی همچنان نیز به

علت مصنوعی بودن داده های مربوط به این دو کلاس کیفیت طبقه بندی این دو کلاس نسبت به سایر کلاس ها پایین تر می باشد.

سوال 4 – Transfer Learning

سوال ۴-۱) Efficient Net

در این شبکه برخلاف مدل های پیشین که یک شبکه baseline را در تنها یکی از سه راستای width و depth و resolution بزرگ می کردند اقدام به تنظیم کردن و scale کردن این سه راستا به طور همزمان و با استفاده از یک compound scaling method می شود.

$$\begin{aligned} \text{depth: } & d = \alpha^\phi \\ \text{width: } & w = \beta^\phi \\ \text{resolution: } & r = \gamma^\phi \\ \text{s.t. } & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\ & \alpha \geq 1, \beta \geq 1, \gamma \geq 1 \end{aligned}$$

scaling coef:

- (depth)$\alpha = 1.2$
- (width)$\beta = 1.1$
- (resolution)$\gamma = 1.15$

با scale کردن شبکه در سه راستا به طور همزمان به شبکه ای با بازدهی مشابه ولی تعداد پارامتر های کمتر و هزینه محاسباتی کمتر دست پیدا می کنیم

ما در این مسئله از مدل baseline مربوط به Efficient Net یعنی EfficientNet-B0 به عنوان مدل pretrained استفاده می کنیم که ساختار زیر را دارد:

Table 1. **EfficientNet-B0 baseline network** – Each row describes a stage i with $\hat{L}_i$ layers, with input resolution $\langle \hat{H}_i, \hat{W}_i \rangle$ and output channels $\hat{C}_i$. Notations are adopted from equation 2.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

سایز ورودی شبکه تصاویر سه کاناله با ابعاد 224*224 می باشد و با توجه به اینکه تصاویر موجود در دیتاست ما دارای ابعاد 32*32 هستند نیاز به عملیات resize داریم. از طرفی خروجی شبکه یک بردار 1280 تایی می باشد که بیانگر بازنمایی تصویر ورودی است. که در نهایت با اضافه سه لایه dense با اندازه 256 و 256 و 10 نورونی عملیات طبقه بندی مد نظر را انجام می دهیم.

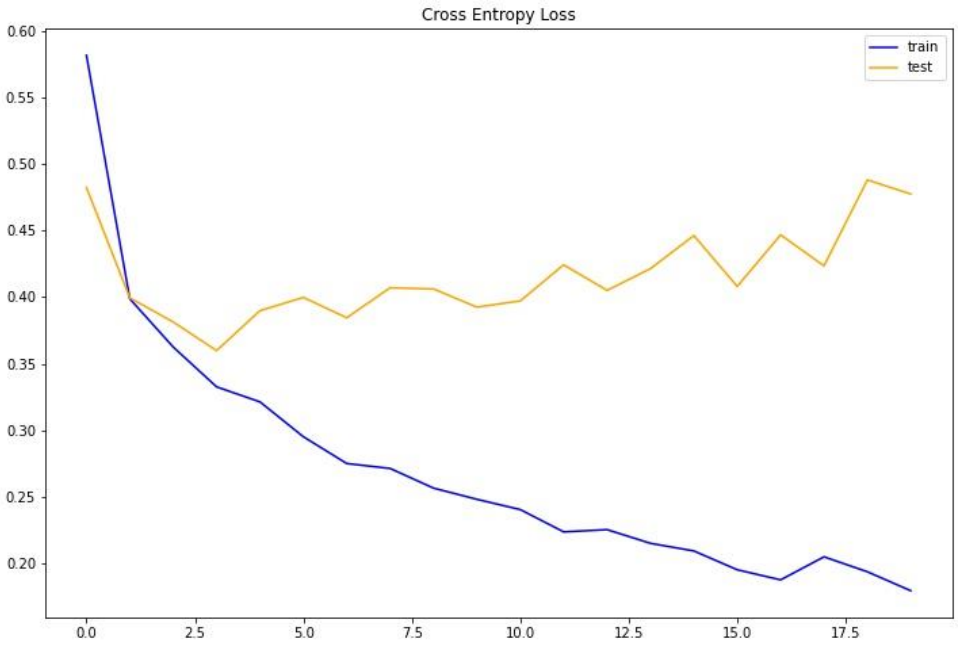
سوال ۴-۲)

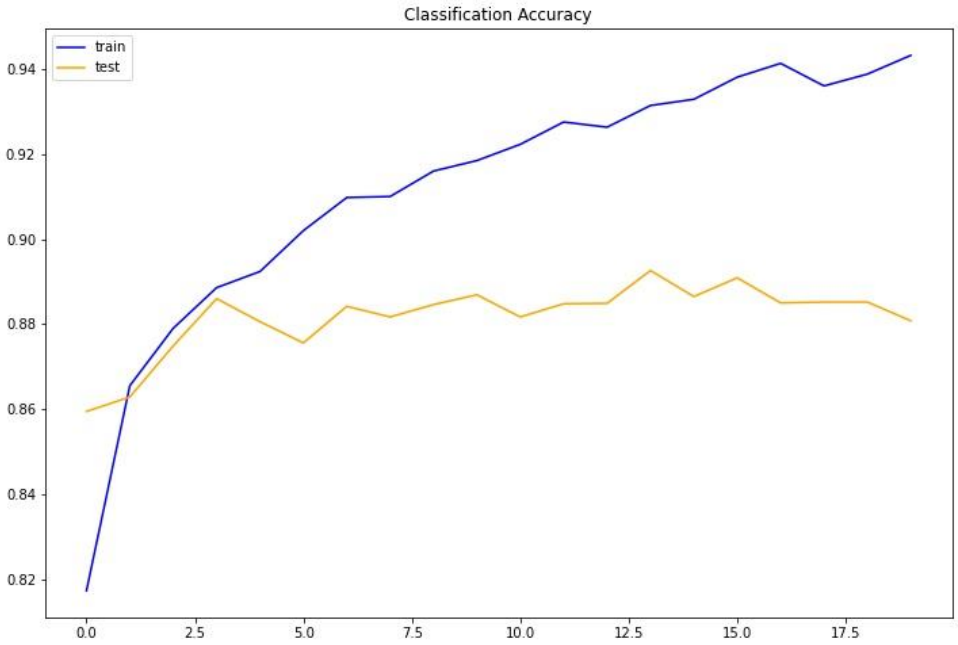
Transfer Learning: گاهی اوقات به علت محدود بودن منابع یا زمان برای آموزش شبکه یا نداشتن داده های آموزشی کامل اقدام به استفاده از شبکه های از پیش آموزش دیده (pretrained) به عنوان بخشی از شبکه جدید استفاده می کنیم. دقت شود که شبکه مورد استفاده باید با استفاده از داده های متجانس با داده های مسئله اصلی آموزش دیده باشد.

پس از اضافه کردن شبکه از پیش آموزش دیده به شبکه اصلی می توانیم فرایند یادگیری را برای پارامتر های کل شبکه انجام دهیم یا قسمت از پیش آموزش دیده (pretrained) را به صورت freeze شده در نظر بگیریم و سایر پارامتر ها را آموزش دهیم.

سوال ۴-۳)

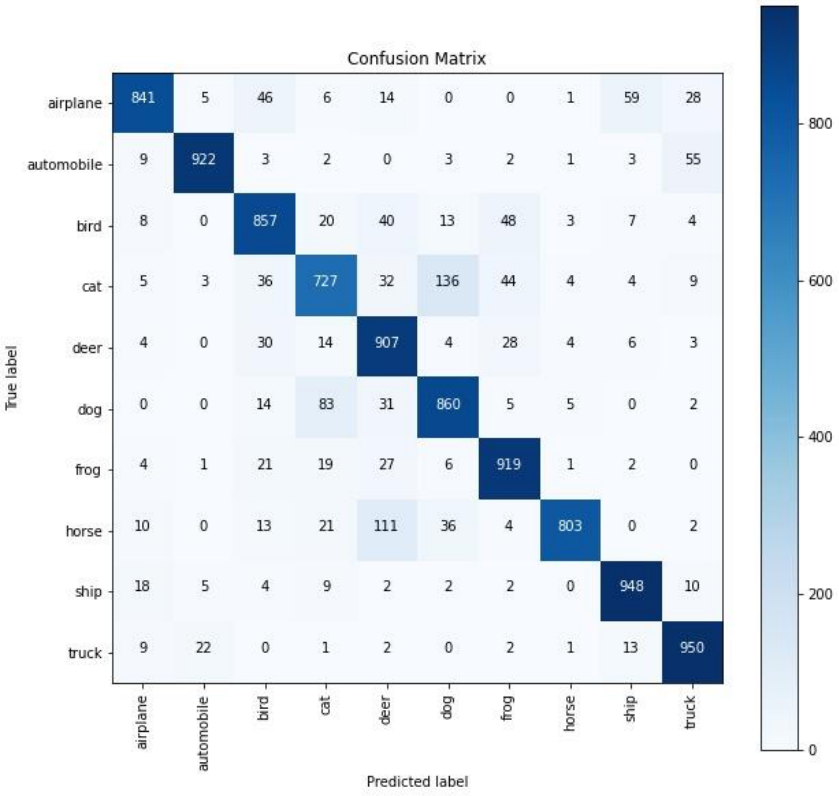
برای آموزش مدل قسمت pretrained شده را freeze در نظر میگیریم و پارامتر های مربوط به لایه های fully connected سه لایه مخفی انتهایی را آموزش می دهیم. (batch size=256 و epoch=20)





```
CPU times: user 4min 36s, sys: 25.3 s, total: 5min 1s
Wall time: 32min 45s
```

همانطور که مشاهده می شود مدل پس از گذراندن تقریبا 3 تا epoch همگرا شده و از آن به بعد شروع به overfit می کند. (زمان و نمودار گزارش شده مربوط به طی شدن تمام 20 تا epoch می باشد)



	precision	recall	f1-score	support
airplane	0.93	0.84	0.88	1000
automobile	0.96	0.92	0.94	1000
bird	0.84	0.86	0.85	1000
cat	0.81	0.73	0.76	1000
deer	0.78	0.91	0.84	1000
dog	0.81	0.86	0.83	1000
frog	0.87	0.92	0.89	1000
horse	0.98	0.80	0.88	1000
ship	0.91	0.95	0.93	1000
truck	0.89	0.95	0.92	1000
accuracy			0.87	10000
macro avg	0.88	0.87	0.87	10000
weighted avg	0.88	0.87	0.87	10000

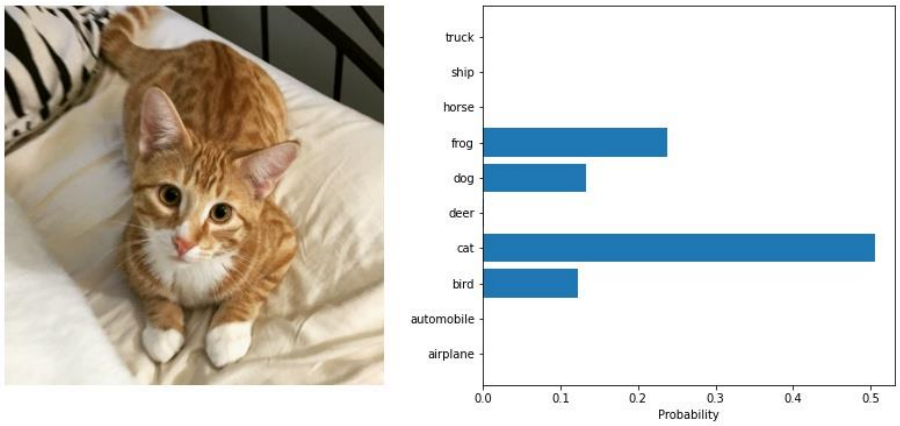
مدل در نهایت به دقت تقریبی 87 درصد برای داده های تست دست پیدا کرده است.

سوال ۴-۴)

طبقه های مدل :

- هواپیما (airplane)
- اتومبیل (automobile)
- پرنده (bird)
- گربه (cat)
- گوزن (deer)
- سگ (dog)
- قورباغه (frog)
- اسب (horse)
- کشتی (ship)
- کامیون ((truck

سوال ۴-۵)



با توجه به خروجی بدست آمده از شبکه تصویر به ترتیب احتمال در یکی از دسته های گربه , قورباغه و سگ قرار می گیرد که برچسب تصویر نیز همان محتمل ترین کلاس پیش بینی شده توسط مدل می باشد
