پروژه شبکه‌های کانولوشنی

سارا بارونی

.

آماده کردن داده

پس از لود کرده مجموعه داده‌ی cifar۱۰ نیاز داریم تا ورودی‌ها را به شکل زیر reshape کنیم. که ۴۵۰۰۰

```
X_train = X_train.reshape(45000,32,32,3)
X_validation = X_validation.reshape(5000,32,32,3)
X_test = X_test.reshape(10000,32,32,3)
```

شکل ۱: مشخص کردن بعد داده‌های ورودی

نشان‌دهنده‌ی تعداد داده‌ها و ۳۲ و ۳۲ ابعاد تصویر را نشان می‌دهد که در cifar۱۰ ۳۲*۲۳ است و عدد ۳ نشان میدهد که تصاویر ۳ کاناله هستند.

onehot کردن

همچنین ما نیاز داریم تا داده‌ها onehot شوند همانطور که در تمرین‌های گذشته هم داشتیم onehot کردن تبدیل لیبل هر داده به آرایه‌ای از صفر و یک ها به طوری که طول این آرایه برابر با تعداد کلاس‌ها و تنها خانه‌ای از آرایه که اندیس آن مقدار لیبل را نشان می‌دهد مقدار یک را میگیرد و دیگر خانه‌ها مقدارشان صفر است.
این بار برای برخلاف تمرین‌های قبل از کراس برای onehot کردن استفاده شد که دستور آن در ادامه آورده شده است.

```
from keras.utils import to_categorical
#one-hot encode target column
Y_train = to_categorical(Y_train)
Y_test = to_categorical(Y_test)
```

شکل ۲: onehot کردن

ساختن مدل

برای ساختم مدل از کتابخانه‌ی کراس استفاده شد ضمن این که از مدل sequential استفاده شد این مدل به این شکل است که به راحتی می‌توان هرلایه را به لایه‌ی قبلی اضافه کرد. در تصویر زیر دو لایه‌ی اول دولایه‌ی Conv۲D هستند این‌ها لایه‌های کانولوشنی دو بعدی هستند که در ادامه باتوجه به خواسته‌های تمیرین تغییر خواهند کرد عدد ۶۴ در لایه‌ی اول و ۳۲ در لایه‌ی دوم نشان دهنده‌ی تعداد نورون‌های هر لایه هستند که در ادامه باتوجه به خواسته‌های تمرین تغییر خواهند کرد ضمن اینکه تعداد نورون‌های لایه‌ی آخر به تعداد کلاس‌ها یعنی ۱۰ باقی میماند و این لایه Dence نام دارد. بین لایه‌های کانولوشنی و لایه آخر یک لایه‌ی flatten قرار میدهیم که یک رابط بین آنهاست. همچنین میتوان در kernel size سایز کرنل را مشخص کرد و دیگر پارامترها هم واضح است که چه چیزی را نشان می‌دهند.

کامپایل کردن مدل

کامپایل کردن مدل سه پارامتر میگیرد که شامل optimizer که learning rate را تنظیم میکند که من با توجه به نتایج تمرین قبل که الگوریتم adam اثر بهتری برروی این داده‌ها داشت و همچنین به طور معمول الگوریتم بهتری برای بهینه‌سازی است این گزینه را بر روی adam قرار دادم
پارامتر بعدی loss است که طبق مطالعاتی که داشتم درصورتی که بیش از دو کلاس داشته باشیم مثلا در این تمرین که ۱۰ کلاس داریم و خروجی به این شکل است که فقط یک مورد از این ده مورد مقدار یک را خواهد

```
from keras.models import Sequential
from keras.layers import Dense, Conv2D, Flatten
#create model
model = Sequential()
#add model layers
model.add(Conv2D(64, kernel_size=3, activation='relu', input_shape=(32,32,3)))
model.add(Conv2D(32, kernel_size=3, activation='relu'))
model.add(Flatten())
model.add(Dense(10, activation='softmax'))
```

شکل ۳: نمونه‌ای از مدل کانولوشنی در کراس

داشت و بقیه صفر خواهند شد بهتر است از categorical crossentropy برای loss استفاده شود.
پارامتر بعدی metrics است که در واقع یک متریک برای ارزیابی باید انتخاب کنیم که در این از accuracy استفاده شده است.

```
#compile model using accuracy to measure model performance
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

شکل ۴: نمونه‌ای از مدل کانولوشنی در کراس

training
در ادامه مدل را trainn کردم که برای این کار از تابع fit استفاده میشود.که پارامترهای آن شامل ورودی به همراه لیبل‌های آن میباشد و همچنین ورودی و لیبل‌های ولیدیشن را به آن میدهیم که با syntax نشان داده شده در شکل ۵ میباشد.

```
#train the model
model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=3)
```

شکل ۵: tain کردن

تاثیر افزایش یا کاهش تعداد لایه‌های کانولوشنی بر عملکرد شبکه
ابتدا شبکه را با یک لایه‌ی کانولوشنی ترین کردم که خروجی آن در شکل زیر نشان داده شده است. سپس تا ۴ تا لایه تعداد لایه‌ها را افزایش دادم به طوری که مشخصات لایه‌ی قبلی را دارند و فقط تعداد را تغییر دادم که مقادیر خطا و دقت در هر حالت در جدول ۱ آورده شده است.
تعداد و اندازه‌ی کرنل

با توجه به جدول ۲ مشاهده میشود که بهترین اندازه برای کرنل در بین مقادیر داده شده مقدار ۵ میباشد که در ادامه برای بررسی اثر تغییرات تعداد کرنل‌ها بر دقت و خطا اندازه‌ی کرنل ۵ باقی میماند و تعداد کرنل‌ها را

جدول ۱: اثر تغییر تعداد لایه‌ها بر دقت و خطای داده‌ی آموزش

number of convolutional layer	train accuracy	loss train	validation accuracy	loss validation
۱	60.21	1.1599	55.22	1.2997
۲	55.77	1.2820	51.78	1.3744
۳	52.38	1.3568	49.47	1.4112
۴	48.81	1.4449	48.70	1.4460
۵	46.29	1.5126	45.56	1.5257

جدول ۲: اثر تغییر اندازه‌ی کرنل بر دقت و خطای داده‌ی آموزش

Size of kernel	train accuracy	loss train	validation accuracy	loss validation
۳	64.42	1.1412	54.53	1.3383
۵	60.21	1.1599	55.22	1.2997
۱۰	55.19	1.2974	5084	1.4225
۱۵	54.45	1.3028	51.50	1.3711

جدول ۳: اثر تغییر تعداد کرنل ها بر دقت و خطای داده‌ی آموزش

number of kernels	train accuracy	loss train	validation accuracy	loss validation
۵	56.03	1.2863	51.41	1.4250
۱۰	60.21	1.1599	55.22	1.2997
۲۰	61.25	1.1330	57.16	1.2458
۳۰	64.16	1.0591	57.12	1.2425
۵۰	66.64	1.0725	56.55	1.2749
۱۰۰	62.37	1.0899	57.02	1.2597

جدول ۴: اثر تغییر تعداد کرنل ها بر دقت و خطای داده‌ی آموزش

Pooling	train accuracy	loss train	validation accuracy	loss validation
Max pooling	59.3	1.1744	57.06	1.2429
Average pooling	55.88	1.2687	55.97	1.2590
Min pooling	58.42	1.2004	57.16	1.2468

جدول ۵: بررسی اثر droupoout بر عملکرد مدل

Droupout	train accuracy	loss train	validation accuracy	loss validation
without Droupout	59.3	1.1744	57.06	1.2429
0.1	62.74	1.0714	60.96	1.1404
0.2	59.29	1.1683	57.99	1.2097
0.3	58.47	1.1982	55.99	1.2660
0.5	54.38	1.2996	47.58	1.4895
0.7	49.87	1.4205	44.44	1.5587

تغییر میدهیم که نتایج آن در جدول ۳ آمده است.
با توجه به جدول ۳ با افزایش تعداد کرنل‌ها به طور کلی خطا کم شده و دقت زیاد شده است اما از تعدادی که بیشتر میشود تقریبا پرفورمنس ثابت میماند یا حتی بدتر میشود با توجه به جدول تعداد ۲۰ یا ۳۰ کرنل عملکرد بهتری داشته است در ادامه با تعداد ۲۰ کرنل به دیگر قسمت‌های تمرین پاسخ داده خواهد شد. باتوجه به نتایج بدست آمده در این مدل بهترین تعداد کرنل ۲۰ و اندازه آن را ۵ درنظر میگیریم.

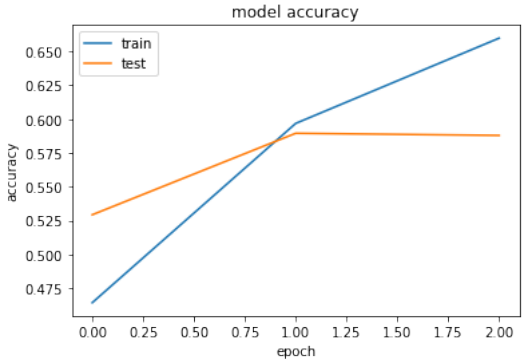
Down sampeling

با توجه به جدول ۴ به نظر میرسد که pooling max و pooling min بهتر عمل می‌کنند زیرا هم دقتشان بیشتر از pooling Average است و هم اینکه خطایشان از pooling Average کمتر است.
لازم به ذکر است که تابع minpooling در کراس به طور آماده وجود نداشت که جدا تعریف شده است به این شکل که میتوان خروجی را منفی کرد و از تابع maxpooling استفاده کنیم و maxpool بگیریم.

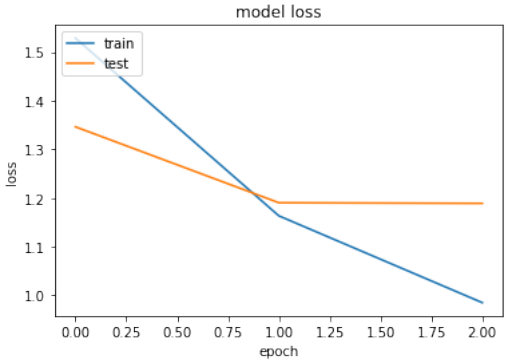
Droupout

droupout به معنای این است که تعداد از نورون‌ها به صورت رندوم حذف میشوند به گونه ای که آن نورون‌ها درنظرگرفته نمیشوند. درواقع مدل از یک حدی که پیچیده‌تر شود میتواند به سمت overfit شدن برود و اینکه اگر تعداد نورون‌ها را کم کنیم با این کار از پیچیدگی مدل کم خواهد شد و این میتواند جلو overfit شدن را بگیرد.

subcaption

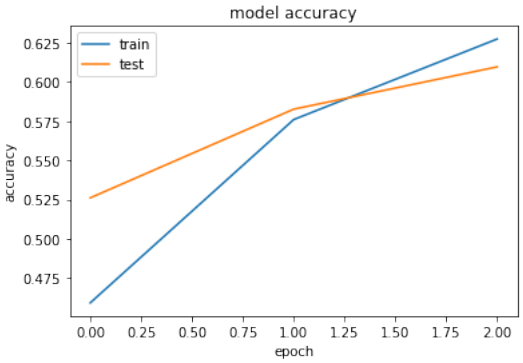


شکل ۶: دقت داده‌های train، validation قبل از اعمال droupout

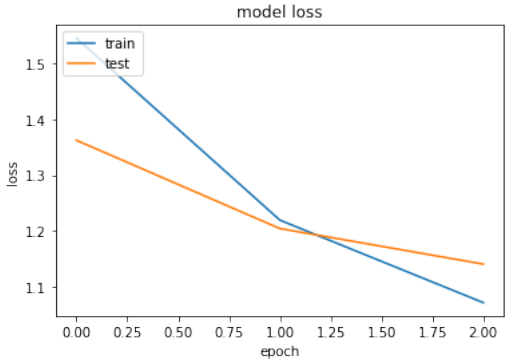


شکل ۷: خطا داده‌های train، validation قبل از اعمال droupout

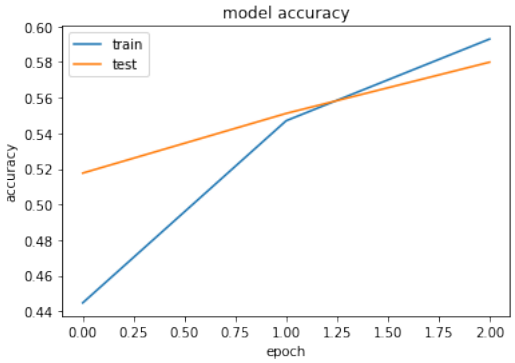
همانطور که در جدول شماره ۵ مشاهده میشود با افزایش درصد نورون‌هایی که دور میریزیم دقت کم می‌شود و خطا زیاد میشود ضمن اینکه وقتی ۱۰ درصد و یا ۲۰ درصد نورون‌ها را نادیده میگیریم دیگر overfit نداریم اما وقتی درصد نورون‌هایی که دور میریزیم را ازین بیشتر کنیم ضمن کاهش پرفورمنس مشکل overfit هم حل نشده است. بنابراین از یک مقداری زیاد حذف کردن اثر نورون‌ها میتواند اثر منفی برروی پرفورمنس بگذارد.
به طور کلی بهترین دقتی که من گرفتم ۶۳ درصد بود که یک لایه کانولوشن با یک لایه maxpooling و با لایه‌ی maxpooling و یک لایه‌ی Droupout با ورودی 0.2 یعنی ۲۰ درصد نورون ها را که خاموش کنیم بود.



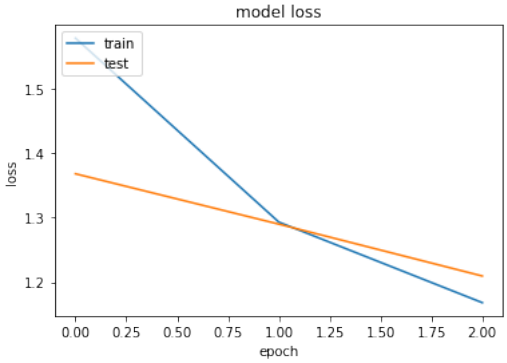
شکل ۸: نمودار دقت با اعمال droupout 0.1



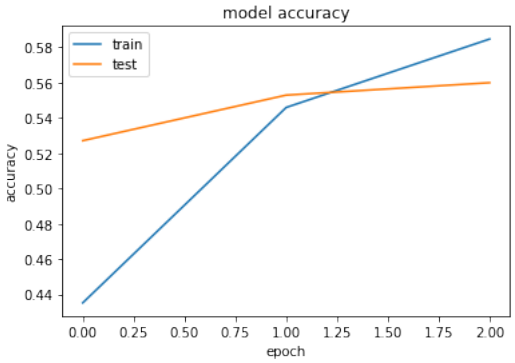
شکل ۹: نمودار خطا با اعمال droupout 0.1



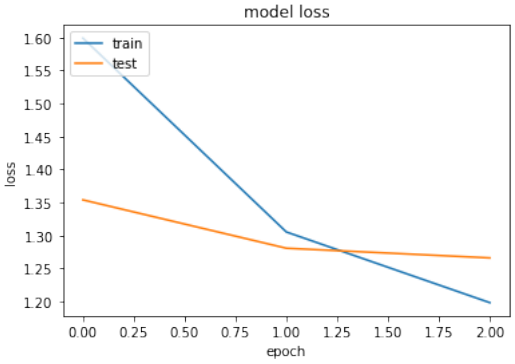
شکل ۱۰: نمودار دقت با اعمال droupout 0.2



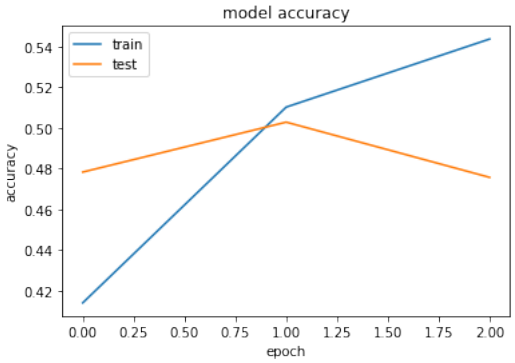
شکل ۱۱: نمودار خطا با اعمال droupout 0.2



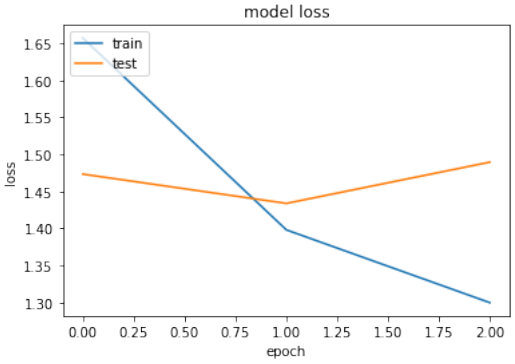
شکل ۱۲: نمودار دقت با اعمال droupout 0.3



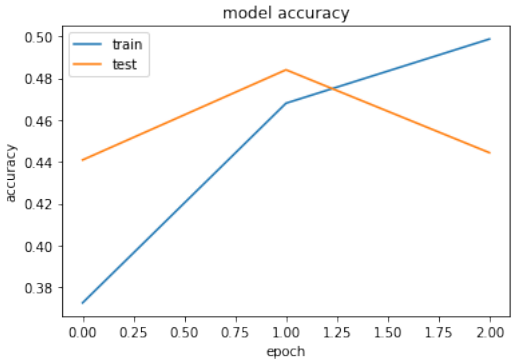
شکل ۱۳: نمودار خطا با اعمال droupout 0.3



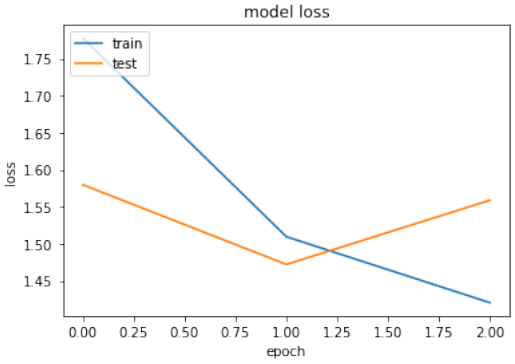
شکل ۱۴: نمودار دقت با اعمال droupout 0.5



شکل ۱۵: نمودار خطا با اعمال droupout 0.5



شکل ۱۶: نمودار دقت با اعمال droupout 0.7



شکل ۱۷: نمودار خطا با اعمال droupout 0.7