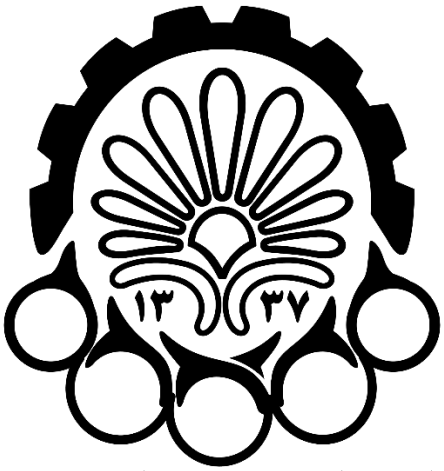


دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

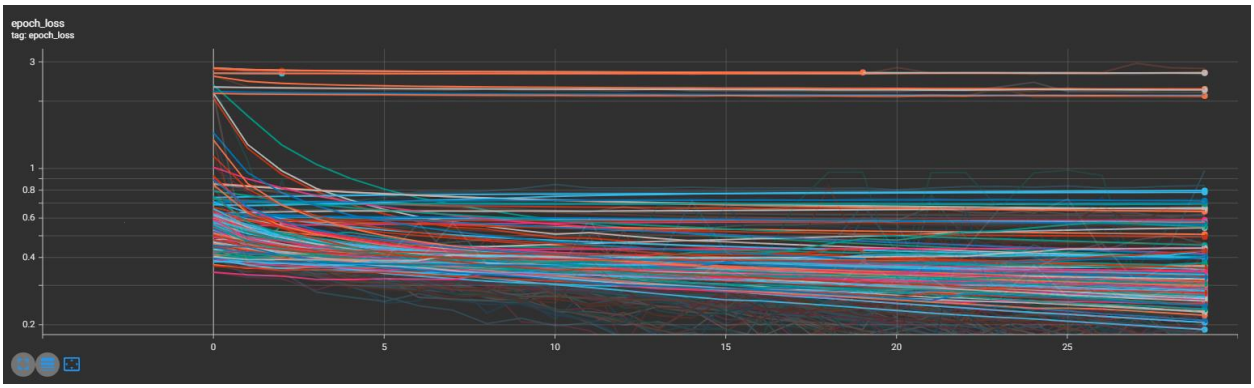
دانشکده مهندسی کامپیوتر

تمرین پنجم درس شبکه عصبی

دکتر صفابخش

غلامرضا دار ۴۰۰۱۳۱۰۱۸

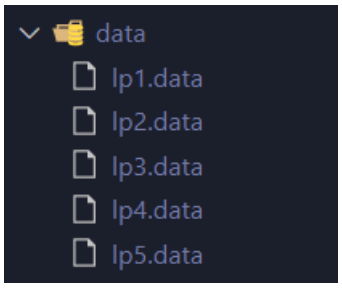
بهار ۱۴۰۱



فهرست مطالب

مقدمه)

آماده سازی داده: قبل از شروع پیاده سازی شبکه های المن و جردن لازم است کمی با دیتاست داده شده در این سوال آشنا شویم. همانطور که در تصویر زیر مشاهده می کنید اطلاعات ذخیره شده توسط سنسور ها به صورت دسته های ۱۵ تایی جدا شده اند و به هر دسته ۱۵ تایی از داده ها یک Label نسبت داده شده است.




```
normal
    -1  -1  63  -3  -1  0
    0   0   62  -3  -1  0
    -1  -1  61  -3  0   0
    -1  -1  63  -2  -1  0
    -1  -1  63  -3  -1  0
    -1  -1  63  -3  -1  0
    -1  -1  63  -3  0   0
    -1  -1  63  -3  -1  0
    -1  -1  63  -3  -1  0
    -1  -1  61  -3  0   0
    -1  -1  61  -3  0   0
    -1  -1  64  -3  -1  0
    -1  -1  64  -3  -1  0
    -1  -1  60  -3  0   0
    -1  0   64  -2  -1  0
```

همچنین دیتا در ۵ فایل مجزا ذخیره شده است این پنج فایل را خط به خط می‌خوانیم و خطوط عددی و لیبل را جدا می کنیم در نهایت به یک دیتاست به ابعاد زیر می رسیم که در مراحل بعد برای آموزش و ارزیابی مدل ها استفاده می شود.

```
X.shape: (463, 15, 6)
X.dtype: float32
y.shape: (463,)
y.dtype: float32
```

در مرحله بعد طبق خواسته سوال داده ها را به مجموعه‌های Train, Test, Validation تقسیم می‌کنیم.

```
X_train.shape: (323, 15, 6)
X_validation.shape: (47, 15, 6)
X_test.shape: (93, 15, 6)
```

پیاده سازی شبکه ها: در ادامه به پیاده‌سازی شبکه های جردن و المن با بهره‌گیری از کلاس پدر Model از کتابخانه Keras می پردازیم. با مراجعه به نوت بوک تمرین میتوانید جزئیات پیاده سازی این دو شبکه را مشاهده کنید.

سوال ۱)

پس از انجام پیاده‌سازی های لازم آزمایش هایی برای پیدا کردن تعداد لایه مخفی بهینه و همچنین مقایسه دو مدل انجام می-دهیم. نتایج آزمایش ها را می‌توانید در جدول زیر مشاهده کنید.

نتایج آزمایش‌های شبکه **المن**			
توضیحات مدل	صحت Train	صحت Validation	صحت Test
المن با ۱۰ نورون مخفی	0.8947	0.9362	0.8602
المن با ۲۵ نورون مخفی	0.9226	0.9362	0.8817
المن با ۵۰ نورون مخفی	0.9536	0.9362	0.8924
المن با ۷۵ نورون مخفی	**0.9783**	**0.8723**	**0.9462**
المن با ۱۰۰ نورون مخفی	0.9783	0.9362	0.9032
المن با ۲۰۰ نورون مخفی	0.9133	0.8723	0.8924

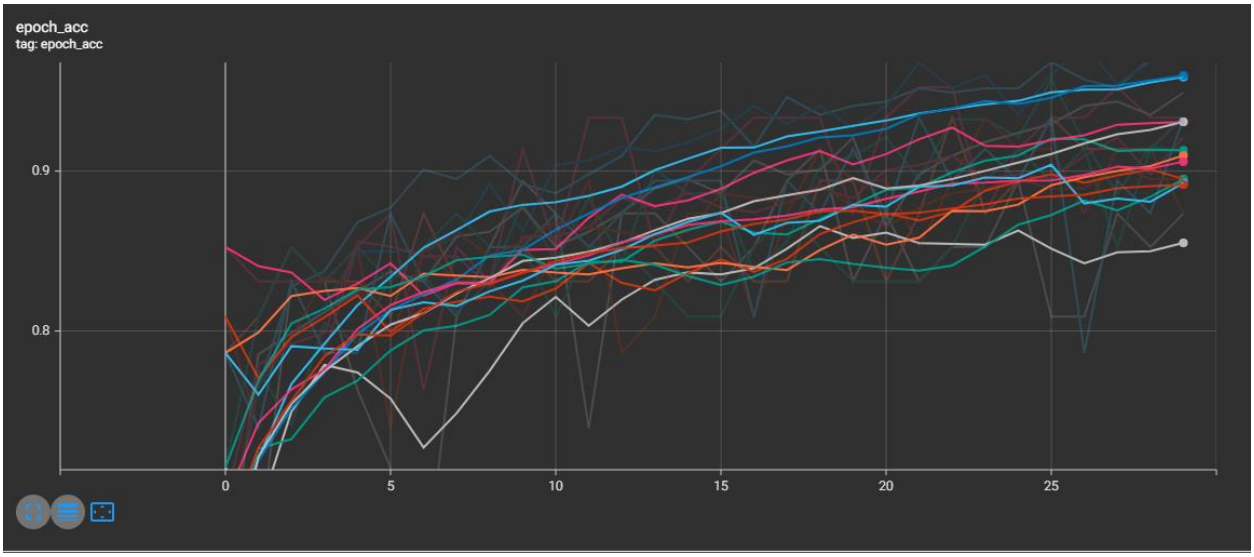
همانطور که مشاهده میکنید افزایش تعداد لایه های مخفی در شبکه المن تا حدی باعث افزایش صحت مدل میشود اما از یک حدی به بعد افزایش تعداد لایه های مخفی صحت مدل را کاهش میدهد.

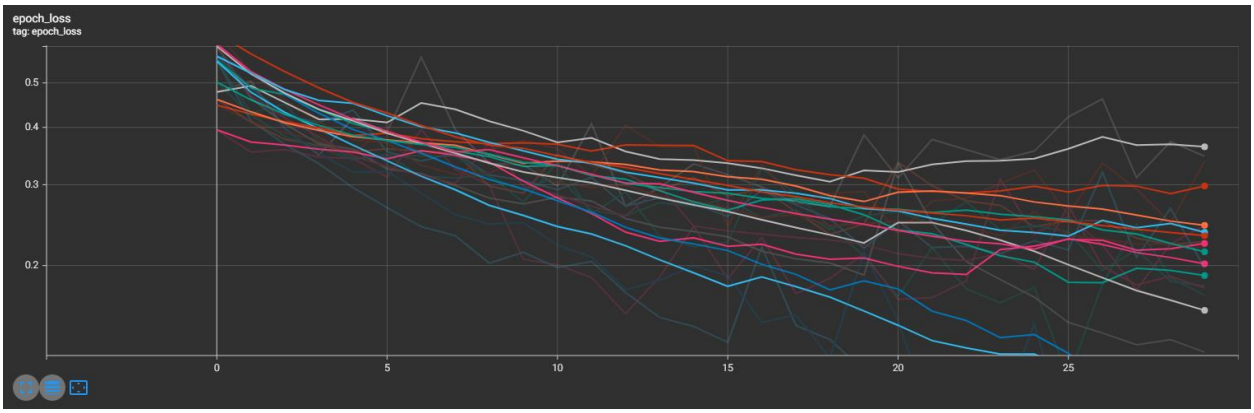
نتایج آزمایش‌های شبکه **جردن**			
توضیحات مدل	صحت Train	صحت Validation	صحت Test
جردن با ۱۰ نورون مخفی	0.8916	0.8723	0.8602
جردن با ۲۵ نورون مخفی	0.8793	0.8936	0.8817
جردن با ۵۰ نورون مخفی	0.9257	0.8298	0.8924
جردن با ۷۵ نورون مخفی	**0.9412**	**0.9149**	**0.8924**
جردن با ۱۰۰ نورون مخفی	0.8576	0.8936	0.8064

مدل جردن نیز مانند مدل المن با افزایش تعداد نورون مخفی تا حدی بهبود می‌یابد اما از حدی به بعد دچار کاهش میزان صحت می‌شود. تعداد نورون بهینه برای هر دوی این شبکه ها عدد **۷۵** بود.

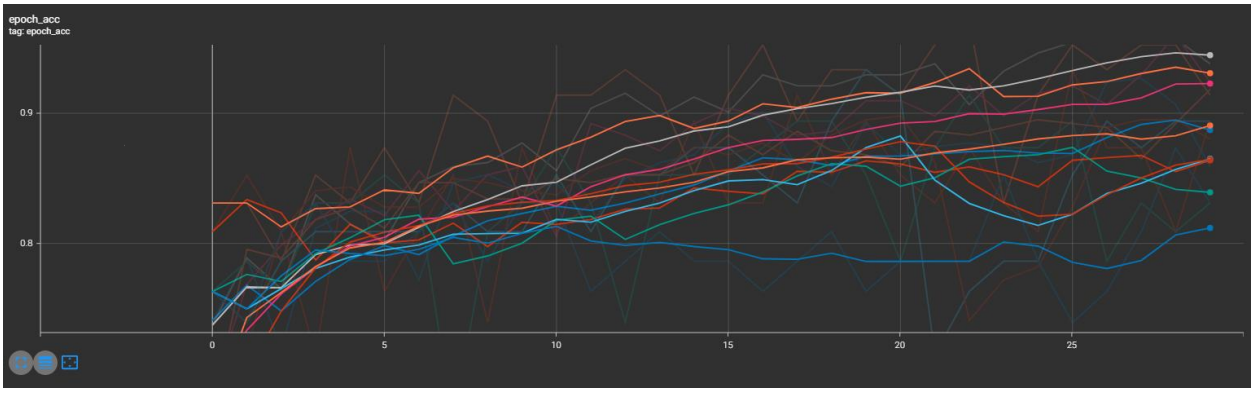
در ادامه نمودار Accuracy و Loss مربوط به تعدادی از آزمایش های مربوط به شبکه های المن و جردن را مشاهده میکنید.
مشاهده می‌شود که مدل المن همگرایی بهتری دارد و مدل جردن در بعضی حالات رفتارهای ناهنجاری دارد.

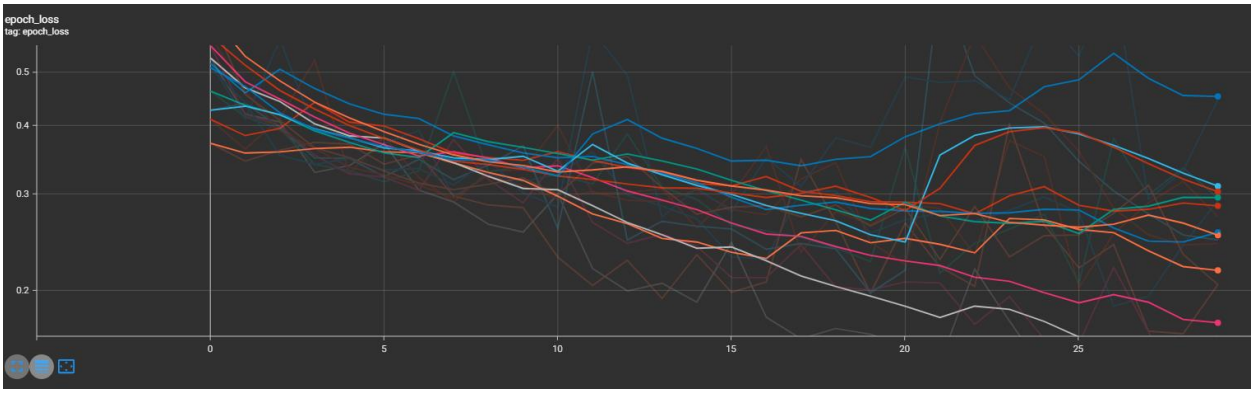
مدل المن



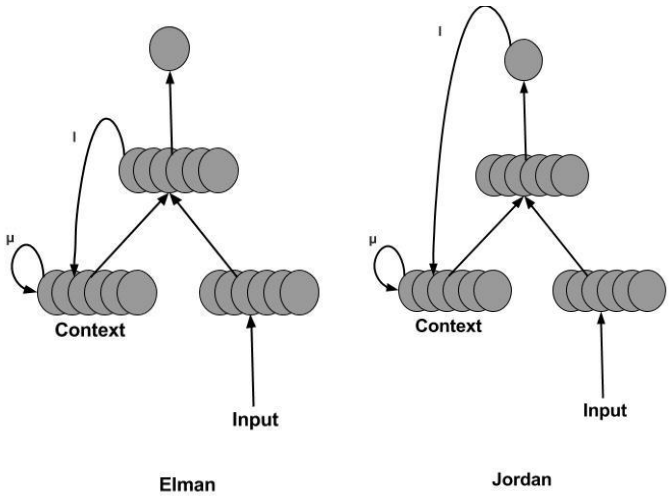


مدل جردن



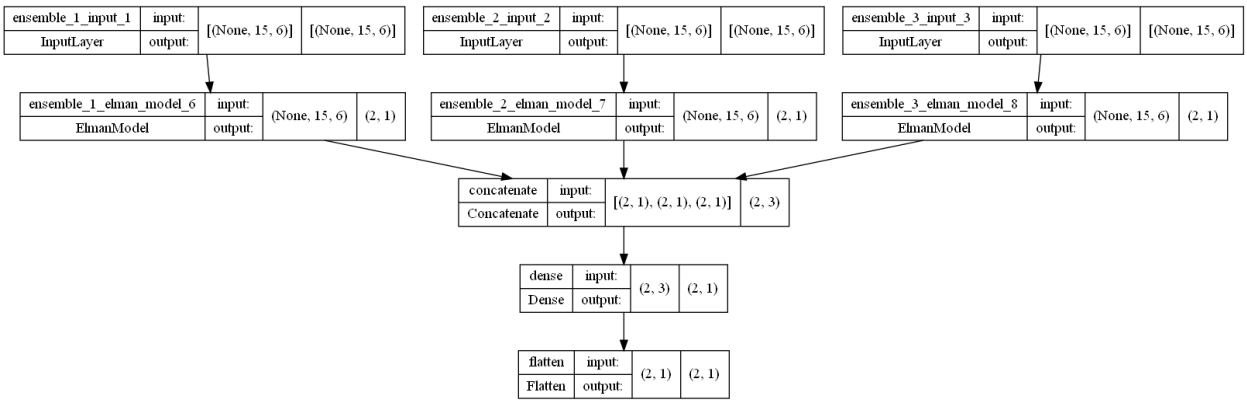


مقایسه نتایج المن و جردن: همانطور که در تصاویر زیر میبینید شبکه علمن خروجی لایه مخفی مرحله قبل را به ورودی لایه مخفی مرحله بعد می‌دهد. ما شبکه جردن خروجی مدل را برمیگرداند و به ورودی لایه مخفی مرحله بعد می‌دهد. با توجه به نتایج به دست آمده مدل **المن** برای این مسئله **مناسب‌تر** است. اگر از دید تاریخی نیز به قضیه بنگریم، مدل المن به عنوان مدل جایگزینی برای مدل جردن معرفی شده بود.



سوال ۲)

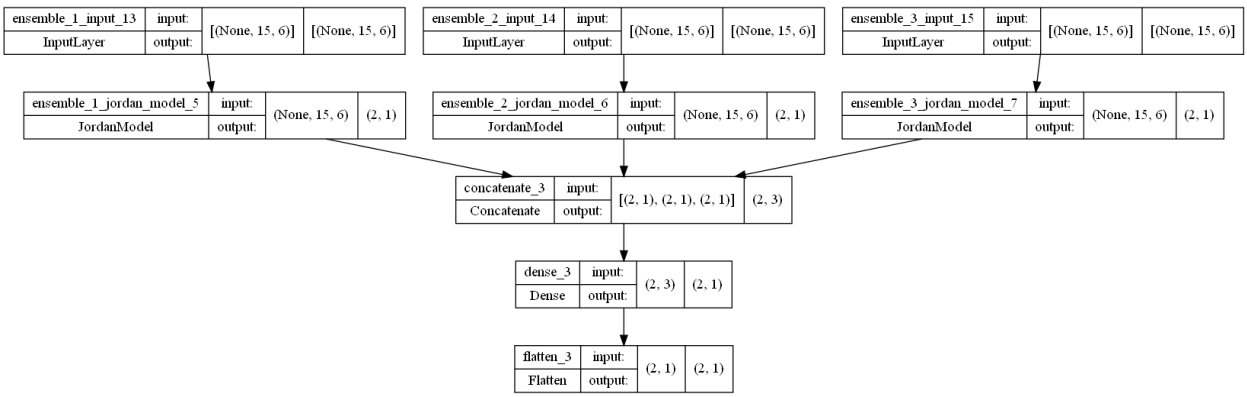
بنابر خواسته سوال ابتدا مدل المن را به صورت گروهی در می‌آوریم. مدل گروهی حاصل را در تصویر زیر مشاهده می کنید.



این مدل از ۳ زیرمدل المن تشکیل شده است این زیرمدل ها ورودی یکسانی می‌گیرند و در نهایت مدل گروهی تصمیم می‌گیرد که این مدل ها چگونه با هم ترکیب شوند تا به بهترین نتیجه دست یابد (لزوما میانگین یا مُد گرفته نمی‌شود). نتیجه تعدادی از آزمایش های این مدل گروهی را در جدول زیر مشاهده می کنید.

نتایج آزمایش‌های شبکه گروهی با زیرمدل های **المن**	
تعداد نورون‌های مخفی زیرمدل‌ها	صحت Test
[10, 25, 50]	**0.9139**
[10, 25, 50, 75]	0.8817
[10, 25, 50, 75, 100]	0.9032

پس از شبکه المن نوبت به مدل گروهی ساخته شده توسط شبکه های جردن رسیده است. مانند مدل المن این مدل گروهی را با زیرمدل های جردن می‌سازیم. تصویر زیر گراف یک مدل با ۳ زیر مدل جردن است.

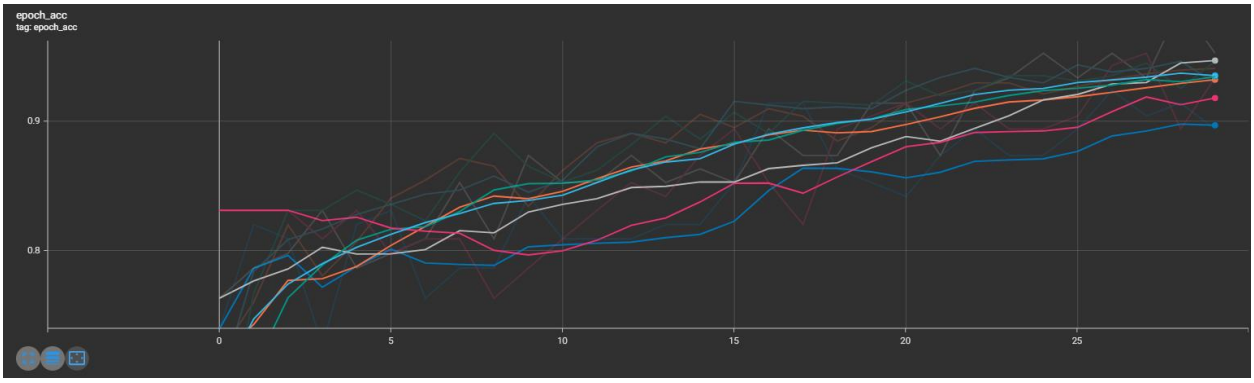


نتیجه تعدادی از آزمایش های مدل گروهی ساخته شده با زیرمدل های جردن را در جدول زیر مشاهده میکنید.

نتایج آزمایش‌های شبکه گروهی با زیرمدل های **جردن**	
تعداد نورون‌های مخفی زیرمدل‌ها	صحت Test
[10, 25, 50]	0.8709
[10, 25, 50, 75]	**0.9139**
[10, 25, 50, 75, 100]	0.8602

در این جا نیز پترن مشابهی مشاهده میکنیم. افزایش تعداد زیرمدل ها تا حدی به بهبود صحت مدل کمک میکنند اما از یک حدی به بعد باعث کاهش آن میشوند. بهترین مدل های گروهی برای هر دو حالت با **رنگ سبز** مشخص شده اند.

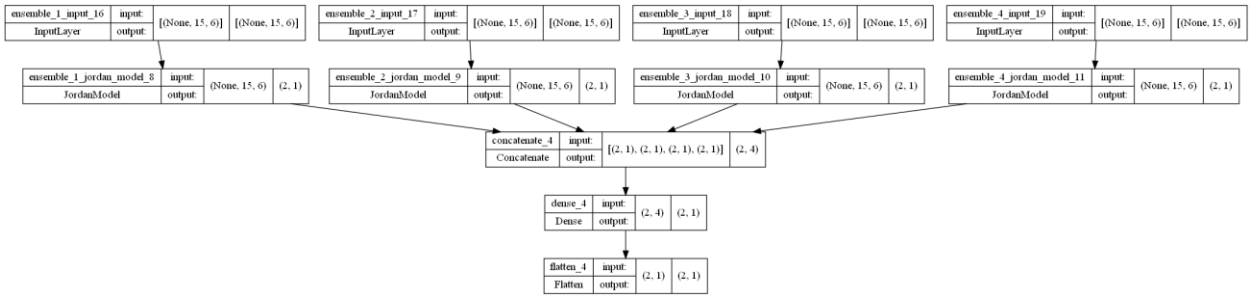
نمودار Accuracy مدل های گروهی با زیرمدل های المن



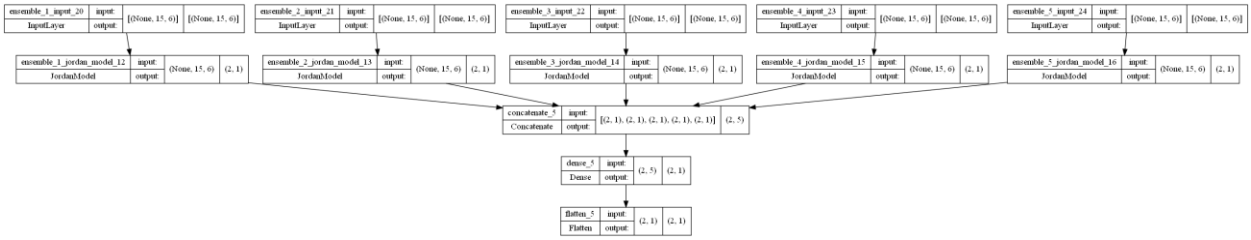
نمودار Accuracy مدل های گروهی با زیرمدل های جردن

تصاویر مربوط به گراف سایر مدل های گروهی در فایل ارسالی تمرین موجود است. چهار نمونه دیگر از این گراف ها را مشاهده میکنید.

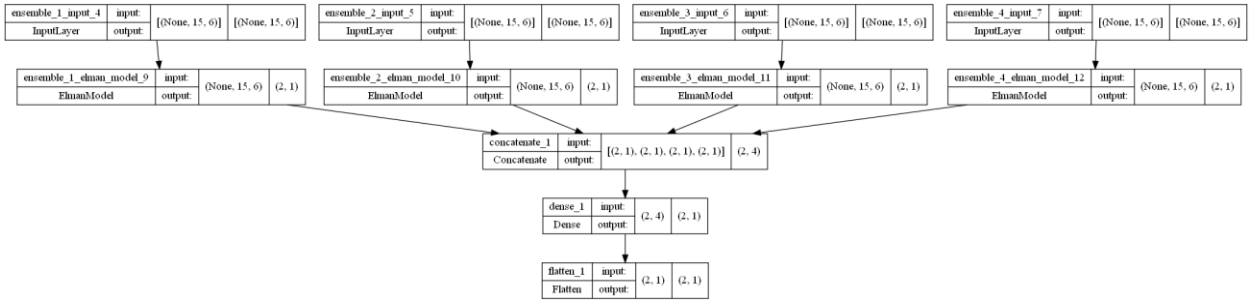
جردن با ۴ زیرمدل



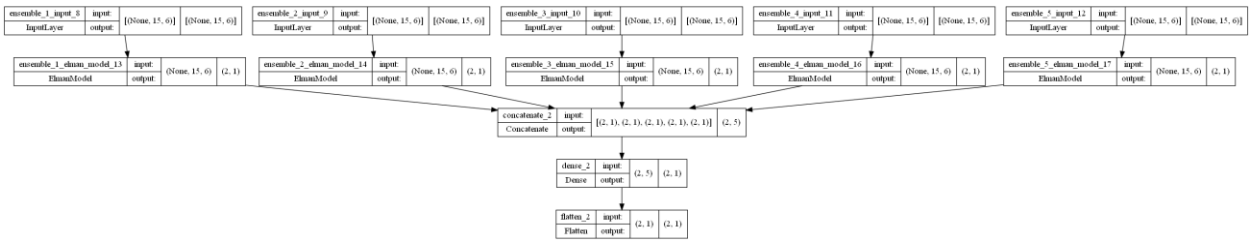
جردن با ۵ زیرمدل



المن با ۴ زیرمدل



المن با ۵ زیرمدل



در نهایت به بخش آخر تمرین می‌رسیم. در این بخش می‌خواهیم با ترکیب زیرمدل های **جردن** و **المن**، بهترین **مدل گروهی** ممکن را بسازیم. همچنین در این بخش با تقسیم بندی مجموعه داده آموزش به تعدادی مجموعه داده کوچکتر هر مجموعه داده کوچک را به یکی از زیرمدل هامی‌دهیم. با این کار زیرمدل ها به صورت مستقل روی دیتاست های مجزا آموزش میبینند. البته این رویکرد ممکن است بهترین رویکرد نباشد و شاید بهتر بود مانند روشهای Boosting اشتباهات مدل اول را بوست کنیم و به مدل دوم بدهیم و الی آخر. با این کار زیر مدل ها به نوعی مکمل هم می‌شدند.

با این حال به پیاده سازی این بخش از سوال میپردازیم. ابتدا مجموعه داده را به تعداد زیرمدل های جردن و المن تقسیم میکنیم.

```
# Split X_train for each sub-model
X_train_list = []
y_train_list = []
X_train_fixed = X_train[:-(X_train.shape[0]%total_submodels)]
y_train_fixed = y_train[:-(X_train.shape[0]%total_submodels)]
for i in range(total_submodels):
    X_train_current = X_train_fixed[int(i*X_train_fixed.shape[0]/total_submodels):int((i+1)*X_train_fixed.shape[0]/total_submodels)]
    y_train_current = y_train_fixed[int(i*X_train_fixed.shape[0]/total_submodels):int((i+1)*X_train_fixed.shape[0]/total_submodels)]
    X_train_list.append(X_train_current)
    y_train_list.append(y_train_current)
```

سپس هر کدام از این زیرمجموعه ها را به عنوان ورودی یکی از زیرمدلها در نظر میگیریم. باید توجه کنیم که هنگامی که یک زیرمدل در حال آموزش دیدن است باید وزن سایر زیرمدل ها را غیرقابل آموزش(فریز) کنیم. این کار را با توجه به اسمی که برای هر لایه در نظر گرفتیم انجام می‌دهیم.

```
def freeze_unfreeze_submodels(ensemble_model, active_index):
    for i in range(len(ensemble_model.layers)):
        # print(i, ensemble_model.layers[i].name)
        if ensemble_model.layers[i].name.startswith(f"ensemble_{active_index}"):
            ensemble_model.layers[i].trainable = True
        elif ensemble_model.layers[i].name.startswith(f"ensemble_"):
            ensemble_model.layers[i].trainable = False

    ensemble_model.compile(optimizer=tf.keras.optimizers.Adam(1e-3), loss='binary_crossentropy', metrics=['acc'])

    return ensemble_model
```

سپس به آموزش دادن مدل نهایی میپردازیم و با آزمایش مقادیر مختلف سعی میکنیم بهترین مدل گروهی ممکن را پیدا کنیم. در جدول زیر تعدادی از آزمایش‌های مربوط به این مدل گروهی را مشاهده میکنید. باید در نظر داشته باشیم که در این حالت افزودن زیرمدل های بیشتر به معنی کاهش داده آموزش هر کدام از زیرمدل هاست که این ممکن است باعث شود حتی از حالت بدون تقسیم داده ها نیز نتیجه بدتری بدست بیاوریم. اگر داده آموزش بیشتری داشتیم ممکن بود بتوانیم با این روش به نتایج بهتری برسیم اما در این مورد این شبکه گروهی نتیجه **بدتری** نسبت به شبکه های گروهی **بدون تقسیم داده‌ها** داد.

نتایج آزمایش‌های شبکه گروهی **ترکیبی** با داده های مجزا	
توضیحات مدل	صحت Test
[10, 75] Elman + [10, 75] Jordan	0.6501
[50, 75] Elman + [50] Jordan	**0.71**
[75] Elman + [75] Jordan	0.688

منابع:

https://machinelearningmastery.com/stacking-ensemble-for-deep-learning-neural-networks/

https://www.tensorflow.org/guide/keras/custom_layers_and_models

https://www.tensorflow.org/guide/keras/custom_layers_and_models#the_layer_class_the_combination_of_state_weights_and_some_computation

https://datascience.stackexchange.com/questions/82416/difference-between-jordan-elman-and-normal-rnn

https://www.youtube.com/watch?v=e2sGq_vI41s&list=PLC112AD1C69432FDB&index=4

https://www.data-blogger.com/elman-rnn-implementation-in-tensorflow/

پایان