دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

پروژه درس مقدمه ای بر هوش محاسباتی

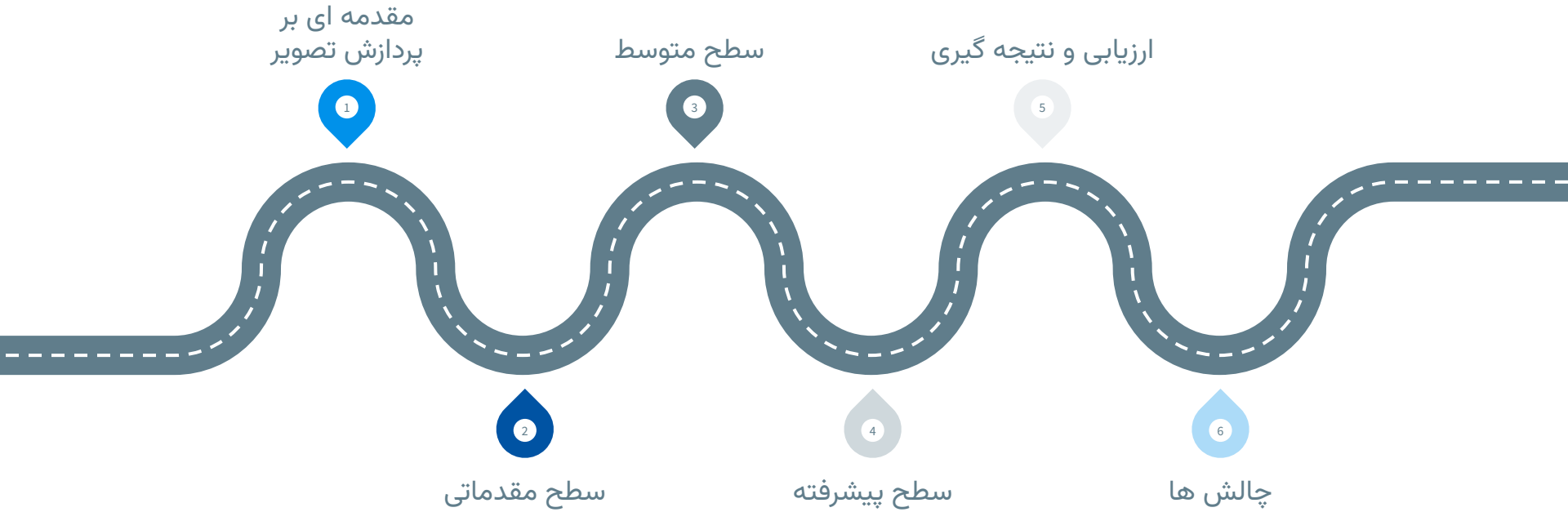
پردازش تصویر

استاد درس: دکتر فرزانه عبدالهی

اعضای گروه

- ریحانه آهنی
- فاطمه رفیعی
- مهدیه سادات بنیس
- سمیرا سلجوقی

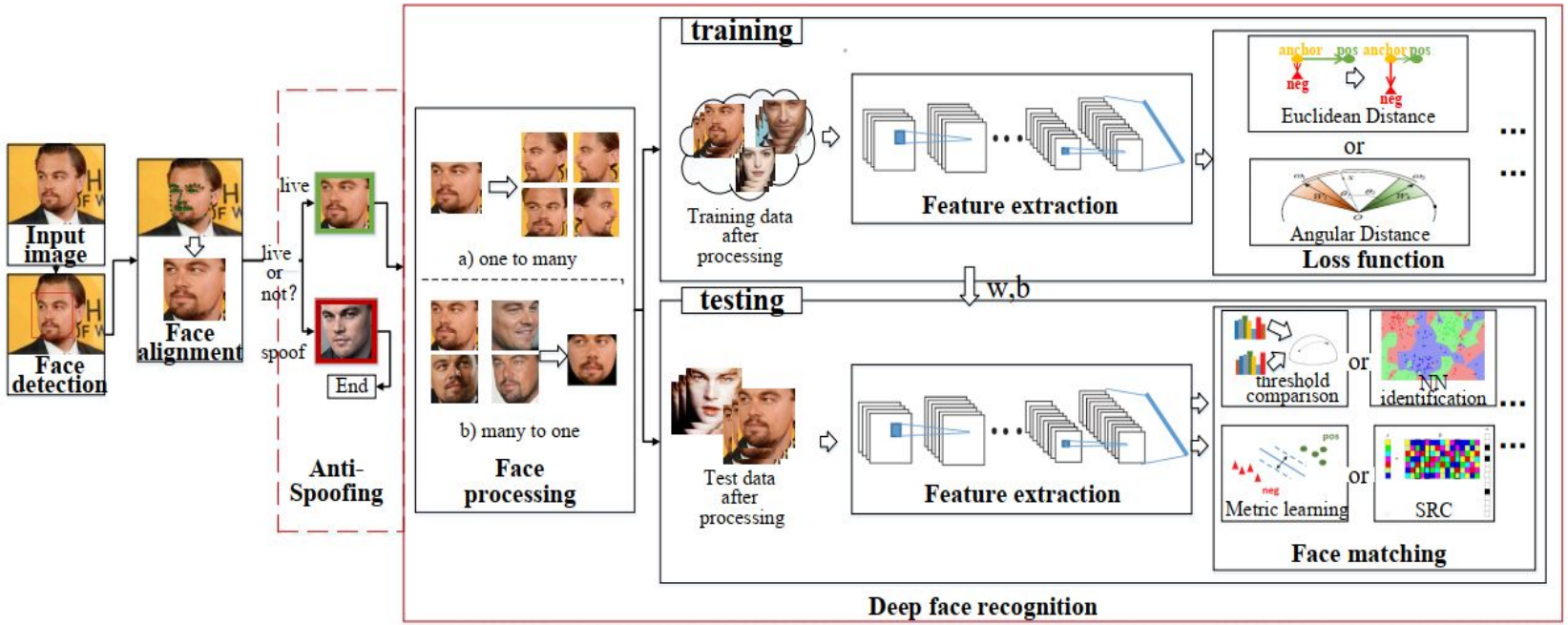
Roadmap


مقدمه ای بر
پردازش تصویر
1
سطح مقدماتی
2
سطح متوسط
3
سطح پیشرفته
4
ارزیابی و نتیجه گیری
5
چالش ها
6

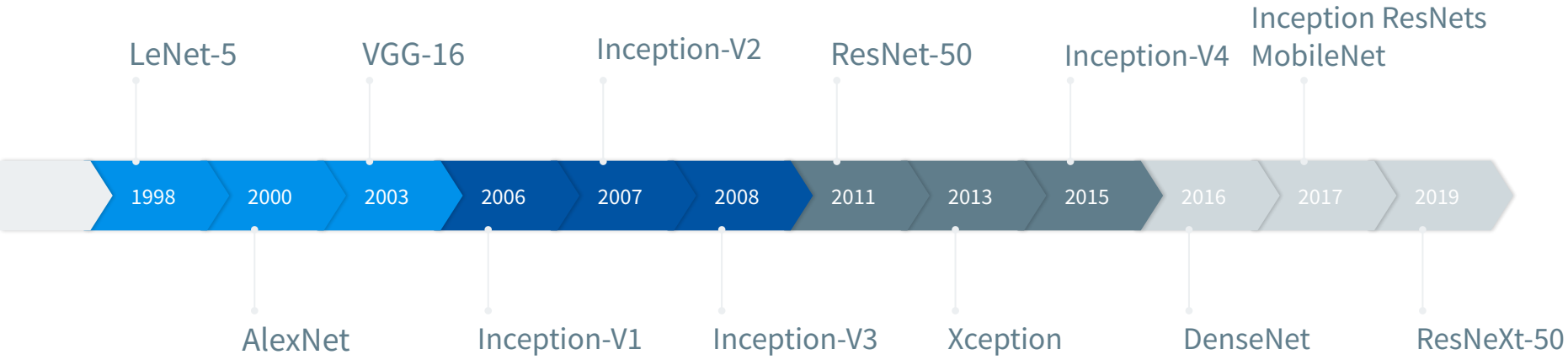
1.

مقدمه ای بر پردازش تصویر

تشخیص چهره چگونه عمل می کند؟



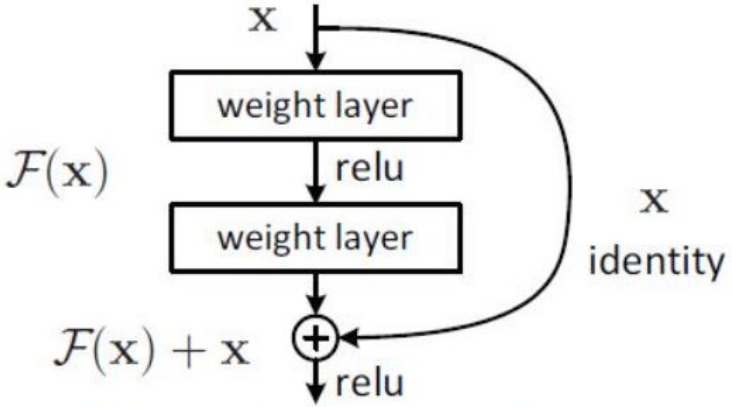
تاریخچه روش های مختلف پردازش چهره



معماری ResNet

وقتی شبکه‌ی ساده (Plain Networks) که لایه‌های کاملاً متصل دارند، عمیق‌تر شوند (یعنی لایه‌ها افزایش می‌یابند)، مشکل محو شدگی گرادیان (Vanishing Gradient) یا انفجار گرادیان (Exploding Gradient) رخ می‌دهد.

اتصالات میانبر (Skip Connections) یا اتصالات اضافی (Residual Connections) راه‌حلی بود که شبکه رزنت (ResNet) برای حل مشکل شبکه‌های عمیق ارائه کرد.

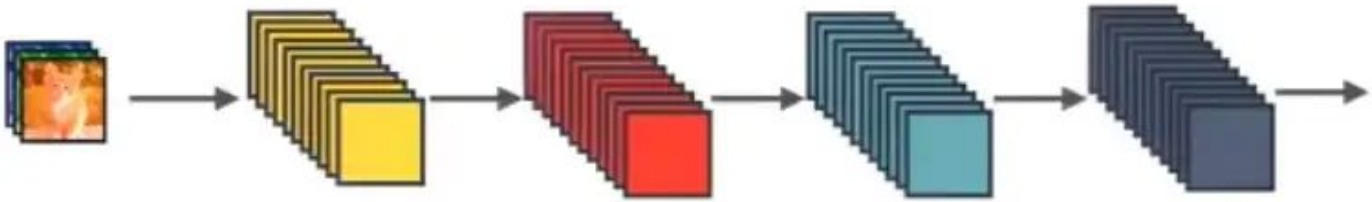


معماری ResNet

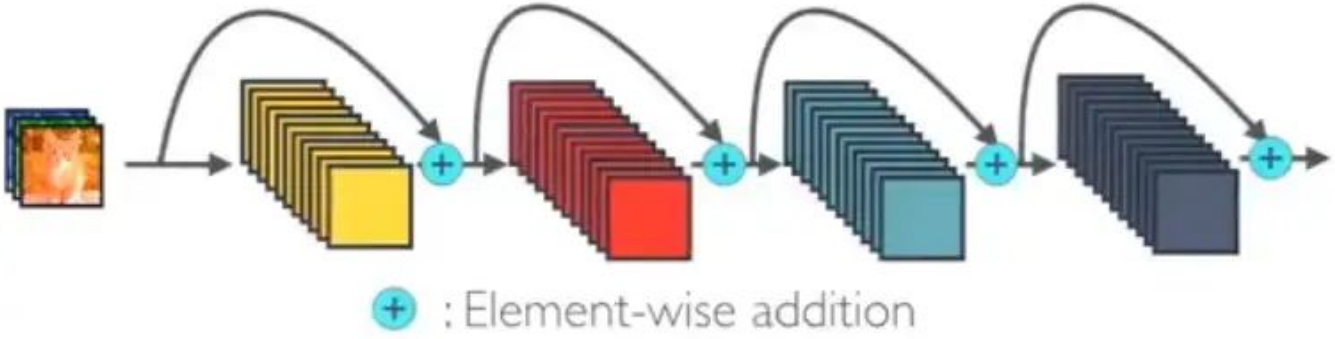
مقایسه ساختار شبکه Residual و شبکه Plain:

معماری DenseNet

- شبکه ConvNet استاندارد

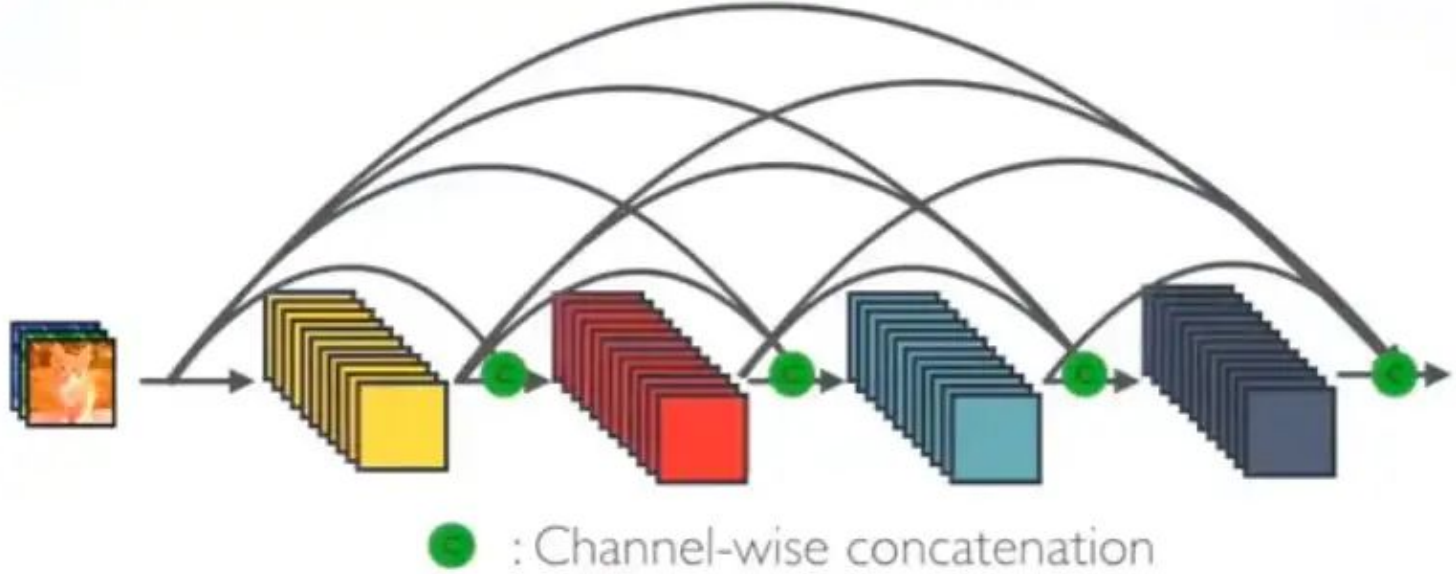


- شبکه ResNet

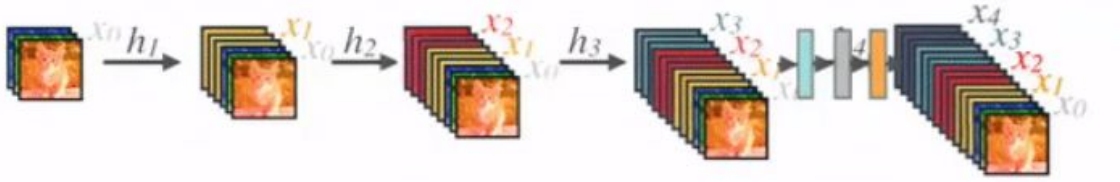


معماری DenseNet

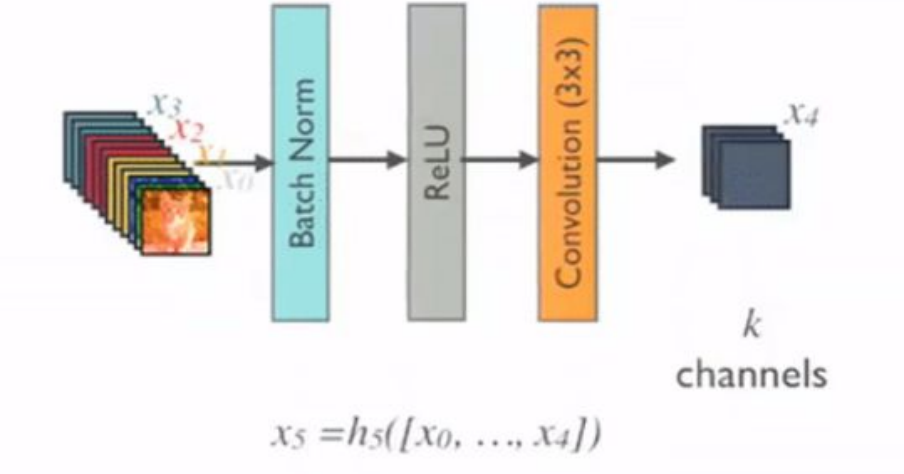
- شبکه **DenseNet**



معماری DenseNet



روش الحاق طی انتشار رو به جلو



معماری DenseNet

مزایای شبکه DenseNet

- **گردش گرادیان قوی:**

سیگنال خطا را می‌توان به راحتی و به صورت مستقیم‌تر به لایه‌های قبلی انتشار داد.

- **کارایی محاسباتی و پارامتری:**

اندازه‌ی شبکه پیچشی متراکم بسیار کوچک‌تر از ResNet است.

- **ویژگی‌های متفاوت‌تر (نامتجانس‌تر):**

چون هر لایه‌ی شبکه پیچشی متراکم همه‌ی لایه‌های قبلی را به عنوان ورودی دریافت می‌کند، ویژگی‌های متفاوت‌تر و الگوهای غنی‌تری دارد.

- **نگهداری ویژگی‌هایی با پیچیدگی کمتر:**

در شبکه پیچشی متراکم، ویژگی‌هایی که طبقه بند استفاده می‌کند سطوح متفاوتی از پیچیدگی دارند. به همین دلیل شبکه پیچشی متراکم روی داده‌های آموزشی محدود، همچنان عملکرد خوبی از خود نشان می‌دهد.

معماری EfficientNet

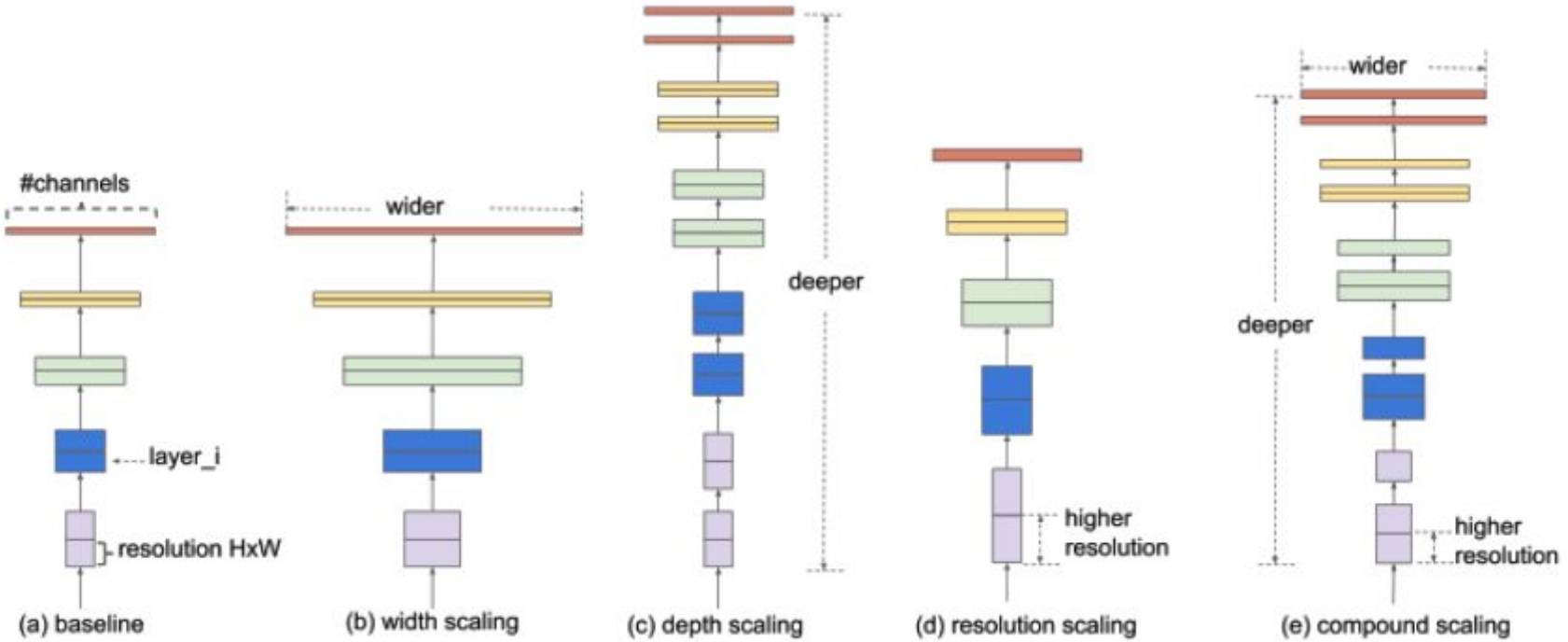
در معماري هاي طراحي شده در شبكه هاي كانولوشني، سه روش براي افزايش دقت استفاده ميشود:

- افزايش عمق شبكه: افزايش تعداد لايه هاي يك شبكه
- افزايش عرض شبكه: افزايش مقدار كانال هاي يك شبكه
- افزايش رزولوشن ورودي

افزايش هر كدام از اين ويژگي ها ميتواند باعث بهبود عملكرد شبكه شود.

در معماری EfficientNet ايده اي در رابطه با طراحي شبكه جديد مطرح نشده است. بلكه با توجه به اينكه دستگاه هاي مختلف از توان پردازشي متفاوتي بهره مند هستند مي‌خواهيم شيوه اي داشته باشيم كه با توجه به دستگاه در دسترس و توانايي پردازش موجود چگونه يك شبكه را Scale كنيم.

معماری EfficientNet



معماری MobileNet و Xception

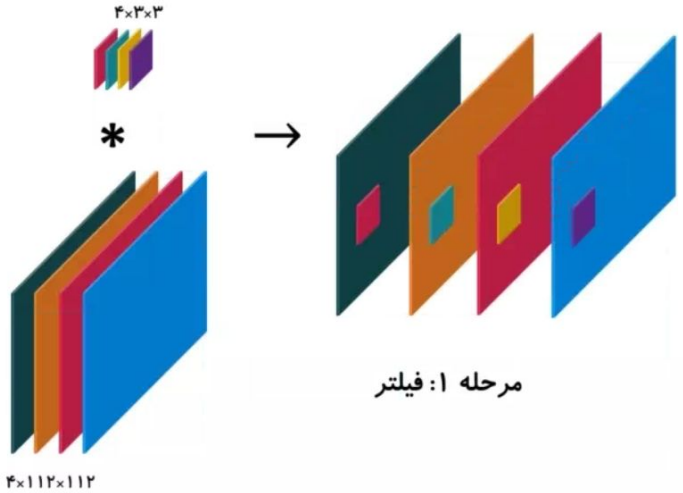
شبکه عصبی mobilenet و xception از شاخص‌ترین شبکه‌های سبک هستند که از شبکه‌های کانولوشنی سبک با پارامترهای کم، سرعت اجرای بالا و دقت قابل‌قبول استفاده می کنند.

در هر دو شبکه برای کاهش تعداد پارامتر های شبکه کانولوشنال از دو عملیات کانولوشن به نام های کانولوشن نقطه ای(pointwise) و کانولوشن بر حسب عمق کانال(depthwise) استفاده میشود که در حین کاهش قابل توجه پارامتر ها،دقت نیز در حد مطلوب باقی میماند.

معماری MobileNet و Xception

- **کانولوشن depth-wise:**

معادل همان فیلتر کردن در کانولوشن استاندارد است اما با یک تفاوت مهم، در کانولوشن استاندارد M کرنل k×k وجود دارد اما در کانولوشن عمقی تنها یک کرنل k×k استفاده می شود. به این مرحله کانولوشن عمقی گفته می‌شود. چون در راستای عمق یا صفحات، روی هر صفحه کانولوشن انجام داده‌ایم و صفحات خروجی را با هم جمع نکردیم.

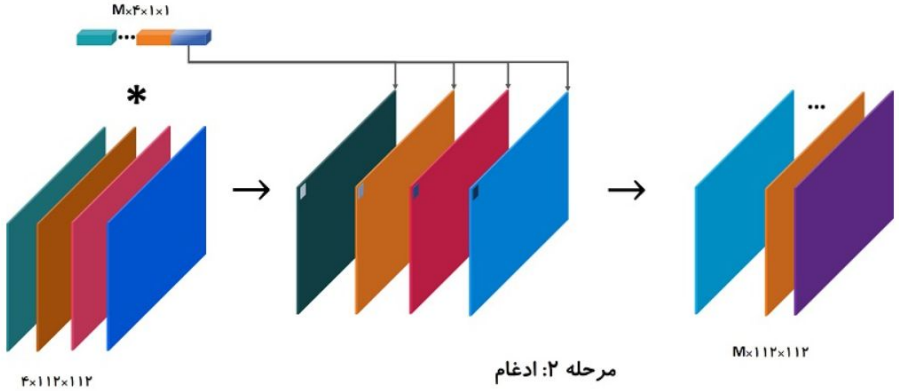


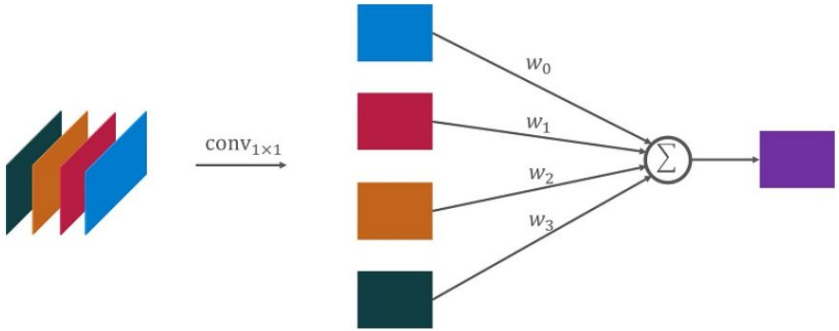
مرحله ۱: فیلتر

معماری MobileNet و Xception

- **کانولوشن point-wise:**

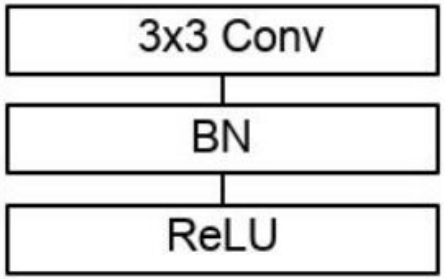
این مرحله معادل با مرحله ادغام در کانولوشن استاندارد است. اما یک تفاوت اساسی بین مرحله ادغام در کانولوشن استاندارد و کانولوشن dws وجود دارد، مرحله ادغام در کانولوشن استاندارد، یک جمع ساده است اما مرحله ادغام در کانولوشن عمقی شامل یک کانولوشن 1×1 است. *کانولوشن 1×1 همچون یک نورون. با وزن‌دهی به صفحات مختلف آنها را باهم جمع می‌کند.*



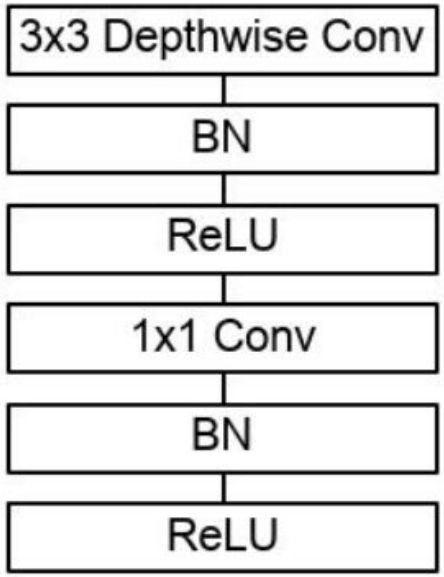


معماری MobileNet و Xception

Standard Convolution

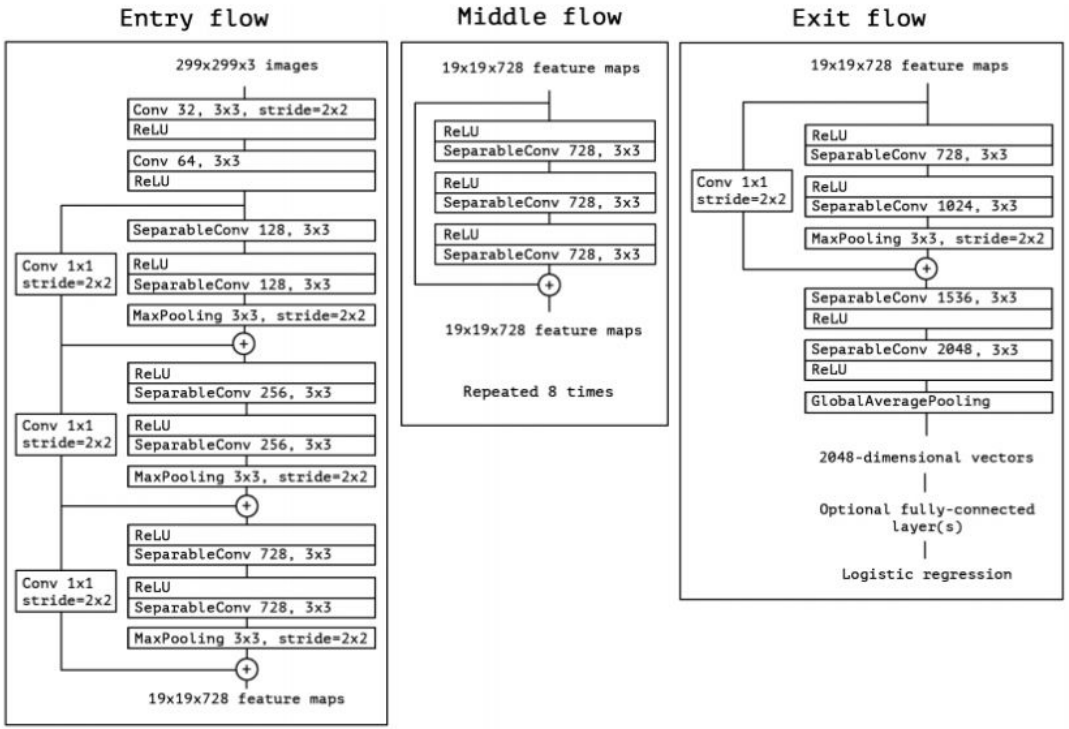


Depth-wise Separable Convolution



معماری MobileNet و Xception

- **ساختار داخلی Xception:**



معماری MobileNet و Xception

- **ساختار داخلی MobileNet:**

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1 Conv / s1	$3 \times 3 \times 512$ dw $1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$ $14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Table 2. Resource Per Layer Type

Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%

2.

سطح مقدماتی پروژه

مجموعه داده ها

Petals to the Metal

Flower Classification on TPU

- 104 types of flowers

پیش پردازش داده ها

Data_augmentation:

- **Resize**
- **Crop**
- **Rotation**
- **Flip (left/right,up/down)**
- **Transpose**

مدل Xception

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
xception (Functional)        (None, 11, 11, 2048)      20861480
_________________________________________________________________
global_average_pooling2d (Gl (None, 2048)              0
_________________________________________________________________
batch_normalization_4 (Batch (None, 2048)              8192
_________________________________________________________________
dense (Dense)                (None, 512)               1049088
_________________________________________________________________
batch_normalization_5 (Batch (None, 512)               2048
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 104)               53352
=================================================================
Total params: 21,974,160
Trainable params: 1,107,560
Non-trainable params: 20,866,600
_________________________________________________________________
```

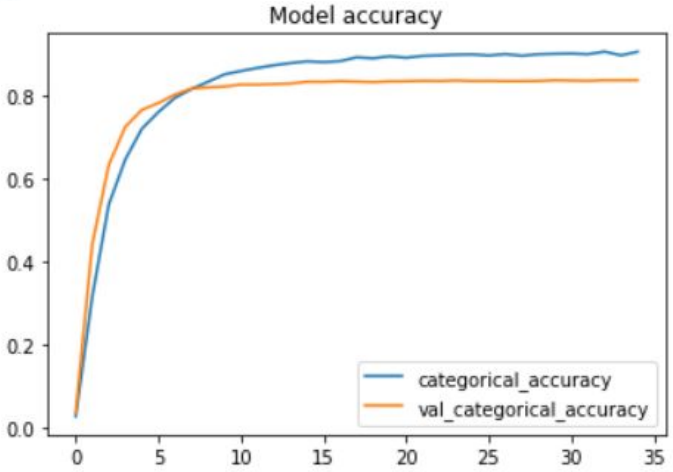
مدل Xception

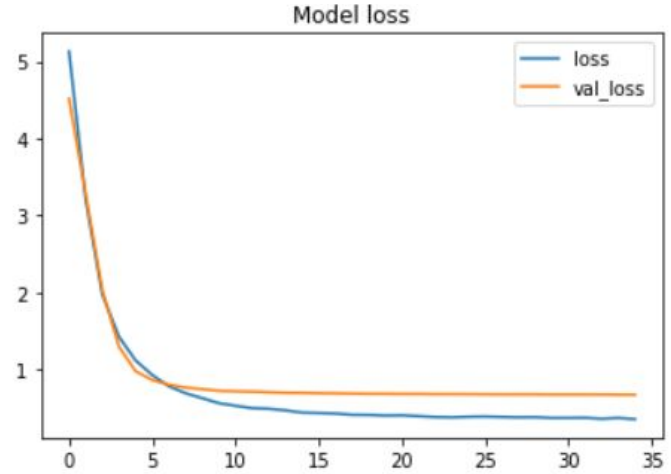
0.8357
Best validation accuracy

0.6770
Best validation loss

0.9061
Best train accuracy

0.3525
Best train loss





مدل DenseNet

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
densenet201 (Functional)     (None, 10, 10, 1920)      18321984
_________________________________________________________________
global_average_pooling2d (Gl (None, 1920)              0
_________________________________________________________________
dense (Dense)                (None, 104)               199784
=================================================================
Total params: 18,521,768
Trainable params: 18,292,712
Non-trainable params: 229,056
_________________________________________________________________
```

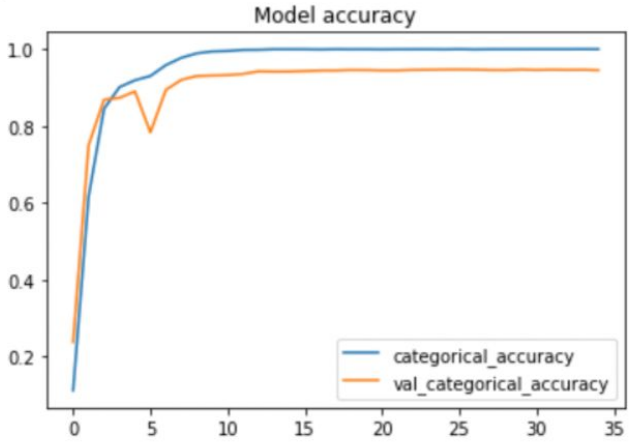
مدل DenseNet

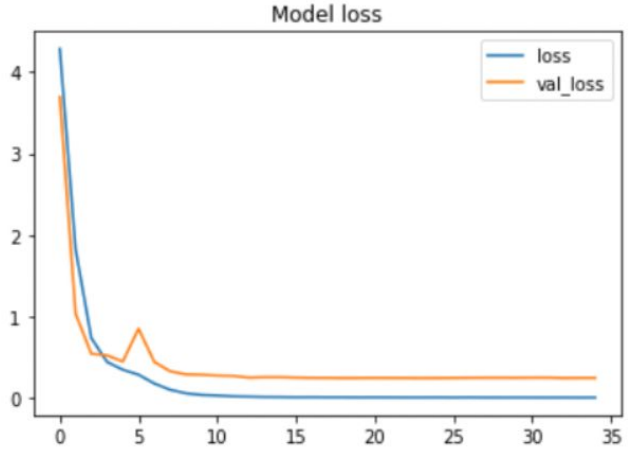
0.9450
Best validation accuracy

0.2462
Best validation loss

0.9998
Best train accuracy

0.0052
Best train loss





مدل EfficientNet

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
efficientnet-b7 (Functional) (None, 2560)              64097680
_________________________________________________________________
dense_1 (Dense)              (None, 104)               266344
=================================================================
Total params: 64,364,024
Trainable params: 64,053,304
Non-trainable params: 310,720
_________________________________________________________________
```

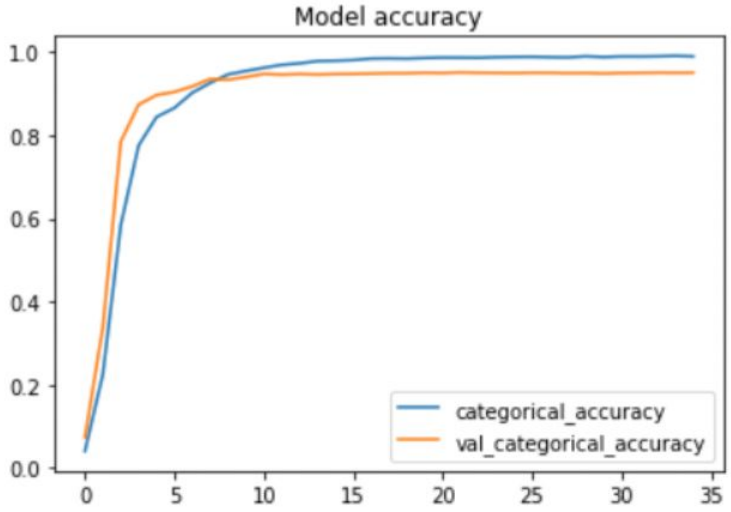
مدل
EfficientNet

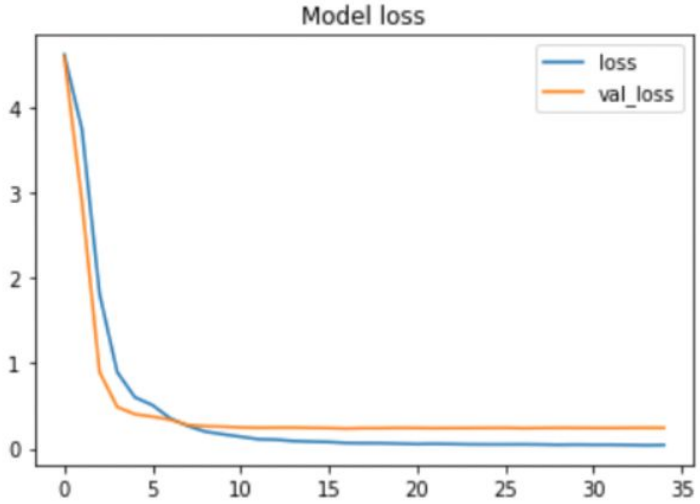
0.9504
Best validation accuracy

0.2402
Best validation loss

0.9889
Best train accuracy

0.0415
Best train loss





نتایج تست

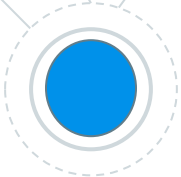
ترکیب دو مدل DenseNet و EfficientNet

0.9832
Best Train accuracy

0.0669
Best Train loss

0.9992
Best Train accuracy

0.0096
Best Train loss



Best Score

0.95461

3.

سطح متوسط پروژه

مجموعه داده ها

14-celebrity-faces-dataset

- 220 images for training
- 70 images for validation

مدل ResNet

Layer (type)	Output Shape	Param #	Connected to
input_4 (InputLayer)	(None, 224, 224, 3)	0	
conv1_pad (ZeroPadding2D)	(None, 230, 230, 3)	0	input_4[0][0]
conv1 (Conv2D)	(None, 112, 112, 64)	9472	conv1_pad[0][0]
bn_conv1 (BatchNormalization)	(None, 112, 112, 64)	256	conv1[0][0]
activation_148 (Activation)	(None, 112, 112, 64)	0	bn_conv1[0][0]
max_pooling2d_4 (MaxPooling2D)	(None, 55, 55, 64)	0	activation_148[0][0]
res2a_branch2a (Conv2D)	(None, 55, 55, 64)	4160	max_pooling2d_4[0][0]
bn2a_branch2a (BatchNormalizati	(None, 55, 55, 64)	256	res2a_branch2a[0][0]
activation_149 (Activation)	(None, 55, 55, 64)	0	bn2a_branch2a[0][0]
res2a_branch2b (Conv2D)	(None, 55, 55, 64)	36928	activation_149[0][0]
bn2a_branch2b (BatchNormalizati	(None, 55, 55, 64)	256	res2a_branch2b[0][0]
activation_150 (Activation)	(None, 55, 55, 64)	0	bn2a_branch2b[0][0]
res2a_branch2c (Conv2D)	(None, 55, 55, 256)	16640	activation_150[0][0]
res2a_branch1 (Conv2D)	(None, 55, 55, 256)	16640	max_pooling2d_4[0][0]
bn2a_branch2c (BatchNormalizati	(None, 55, 55, 256)	1024	res2a_branch2c[0][0]
bn2a_branch1 (BatchNormalizatio	(None, 55, 55, 256)	1024	res2a_branch1[0][0]
add_49 (Add)	(None, 55, 55, 256)	0	bn2a_branch2c[0][0] bn2a_branch1[0][0]
activation_151 (Activation)	(None, 55, 55, 256)	0	add_49[0][0]
res2b_branch2a (Conv2D)	(None, 55, 55, 64)	16448	activation_151[0][0]
bn2b_branch2a (BatchNormalizati	(None, 55, 55, 64)	256	res2b_branch2a[0][0]
activation_152 (Activation)	(None, 55, 55, 64)	0	bn2b_branch2a[0][0]
res2b_branch2b (Conv2D)	(None, 55, 55, 64)	36928	activation_152[0][0]
bn2b_branch2b (BatchNormalizati	(None, 55, 55, 64)	256	res2b_branch2b[0][0]
activation_153 (Activation)	(None, 55, 55, 64)	0	bn2b_branch2b[0][0]
res2b_branch2c (Conv2D)	(None, 55, 55, 256)	16640	activation_153[0][0]
bn2b_branch2c (BatchNormalizati	(None, 55, 55, 256)	1024	res2b_branch2c[0][0]
add_50 (Add)	(None, 55, 55, 256)	0	bn2b_branch2c[0][0] activation_151[0][0]
sequential_4 (Sequential)	(None, 14)	6308878	avg_pool[0][0]

Total params: 29,896,590
Trainable params: 29,843,470
Non-trainable params: 53,120

مدل ResNet

0.7656

Best validation accuracy

0.8883

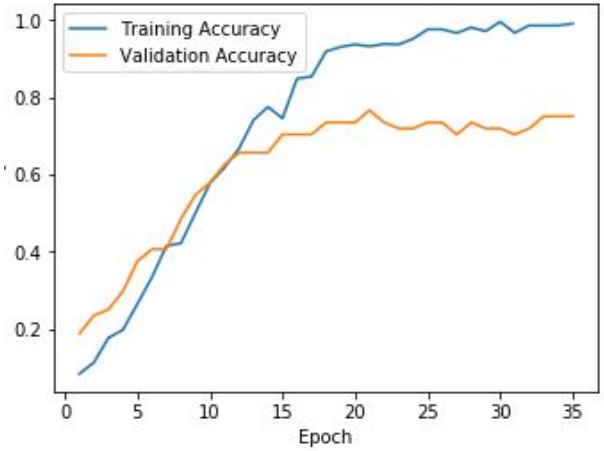
Best validation loss

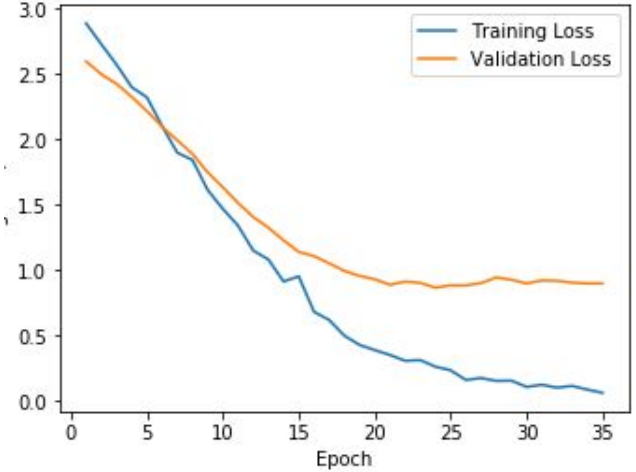
0.9904

Best train accuracy

0.0611

Best train loss





مدل DenseNet

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
densenet201 (Functional)     (None, 8, 8, 1920)        18321984
_________________________________________________________________
global_average_pooling2d (Gl (None, 1920)              0
_________________________________________________________________
dense_2 (Dense)              (None, 4096)              7868416
_________________________________________________________________
dropout_1 (Dropout)          (None, 4096)              0
_________________________________________________________________
dense_3 (Dense)              (None, 14)                57358
=================================================================
Total params: 26,247,758
Trainable params: 26,018,702
Non-trainable params: 229,056
_________________________________________________________________
```

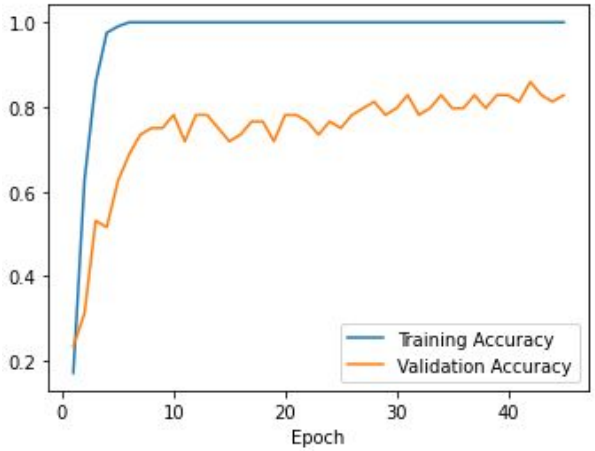
مدل DenseNet

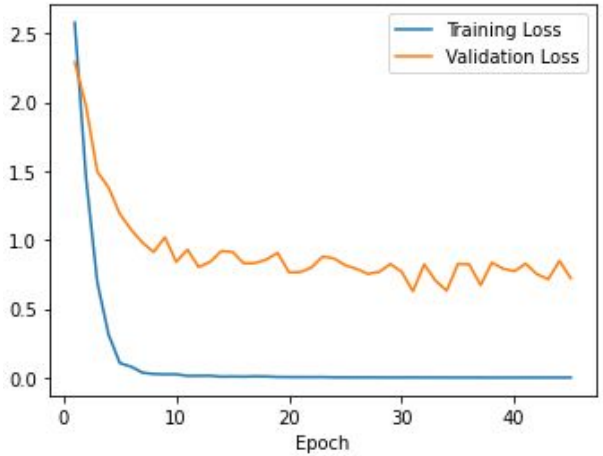
0.8594
Best validation accuracy

0.7149
Best validation loss

1.000
Best train accuracy

0.0012
Best train loss





4.

سطح پیشرفته پروژه

مدل MobileNet

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
mobilenet_1.00_224 (Function (None, 7, 7, 1024)        3228864
_________________________________________________________________
global_average_pooling2d (Gl (None, 1024)              0
_________________________________________________________________
flatten (Flatten)            (None, 1024)              0
_________________________________________________________________
dropout (Dropout)            (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 14)                14350
=================================================================
Total params: 3,243,214
Trainable params: 3,221,326
Non-trainable params: 21,888
_________________________________________________________________
```

تست مدل Mobilenet بر دیتاست celebrities

0.765625

Best validation accuracy

0.637119

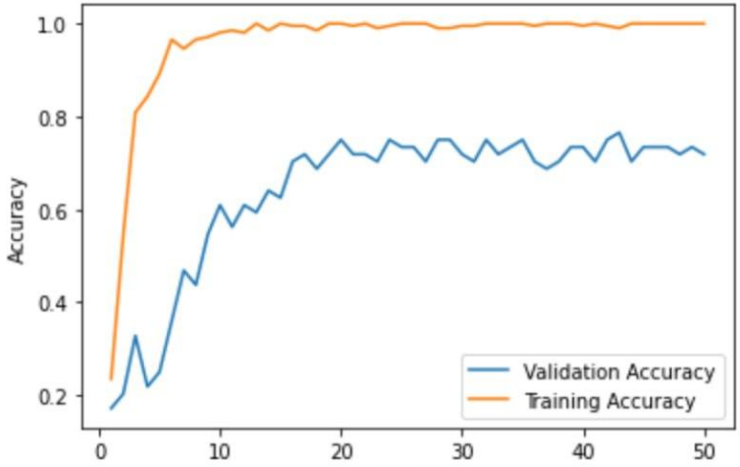
Best validation loss

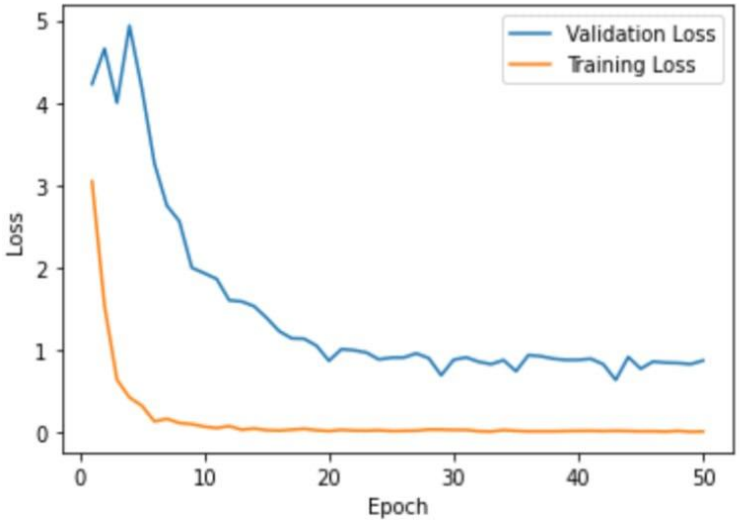
0.98976

Best train accuracy

0.18887

Best train loss





مدل MobileNet

5.

نتیجه گیری و چالش ها

مقایسه بخش دوم و سوم

359.11 MB
ResNet model size

29,896,590
ResNet number of parameters

315.83 MB
DenseNet model size

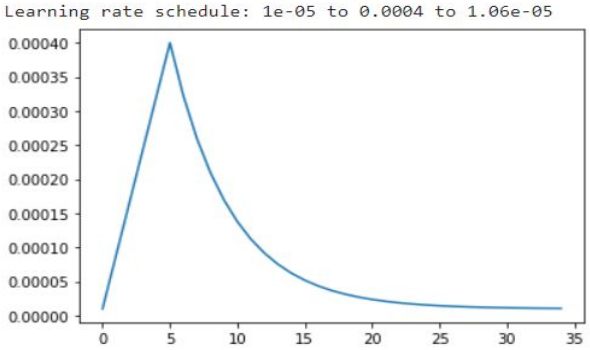
26,018,702
DenseNet number of parameters

25 MB
MobileNet model size

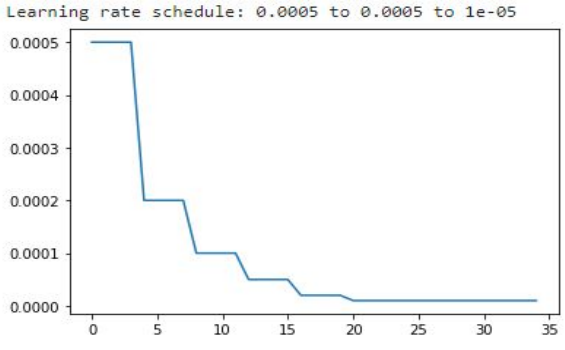
3,221,326
MobileNet number of parameters

چالش ها

- یکی از بزرگترین چالش ها در این پروژه پایین بودن سرعت اجرای کد ها بود حتی با استفاده از TPU
- متوجه شدیم برای استفاده از learning rate بهتر است در طول آموزش شبکه ثابت نباشد و به تدریج کاهش یابد. در بخش اول پس از بررسی هر دو حالت مشاهده کردیم که مدل learning rate الف نتیجه بهتری می دهد.



الف



ب

چالش ها

- برای رسیدن به عملکرد بهتر شبکه می توان از دو مدل استفاده کرد و پس از چک کردن صحت هر دو مدل، در صورت مطلوب بودن آن ها می توان برای کل داده ها از ترکیب این دو مدل استفاده کرد.
- در بخش دیتاست 14 سلبریتی از آن جایی که حجم داده ها کم بود در ابتدا با استفاده از مدل cnn ساده شبکه را آموزش دادیم اما صحت در حدود 50 درصد بود که مطلوب نبود و در نتیجه معماری شبکه را تغییر دادیم.
- در بخش سوم در ابتدا تنها از مدل mobile net استفاده کردیم که به اورفیت منجر شد و در ادامه با استفاده از drop out این مشکل را برطرف کردیم.
- در بخش سوم برای تغییر learning rate در ابتدا از حالت قسمت الف(غیر پلکانی) استفاده کردیم اما صحت مطلوب نبود که با تغییر آن به حالت قسمت ب(پلکانی) صحت بر روی دادگان validation بسیار بهتر شد.

ممنون از توجه شما