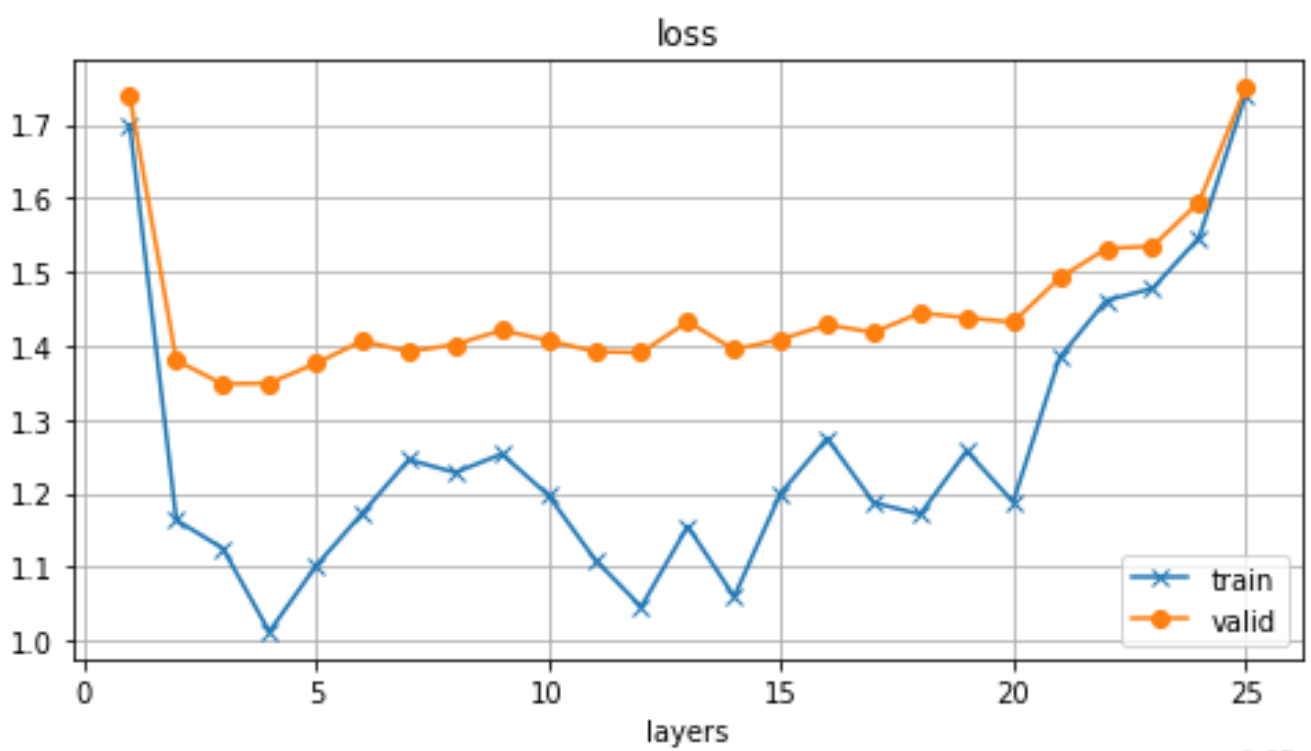
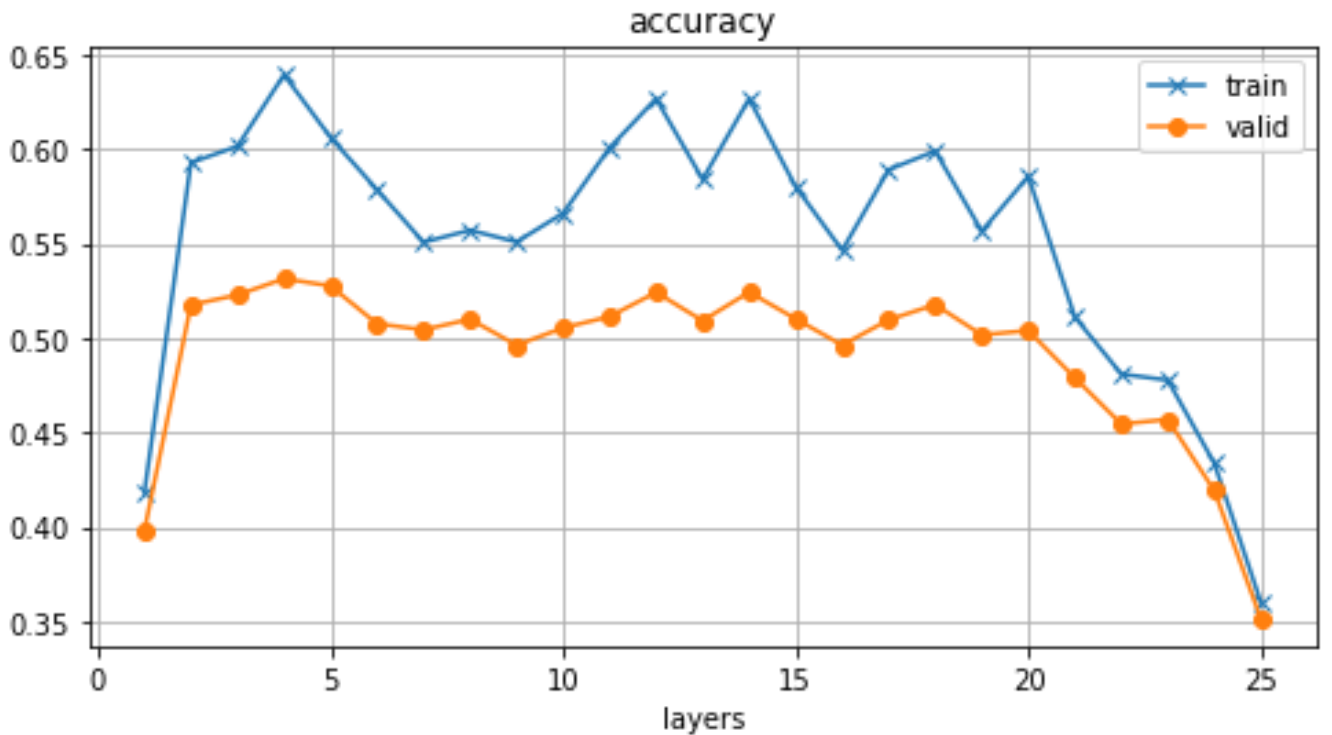

بسم الله الرحمن الرحيم

# شبکه‌های عصبی کانولوشنی

# Convolutional Neural Networks

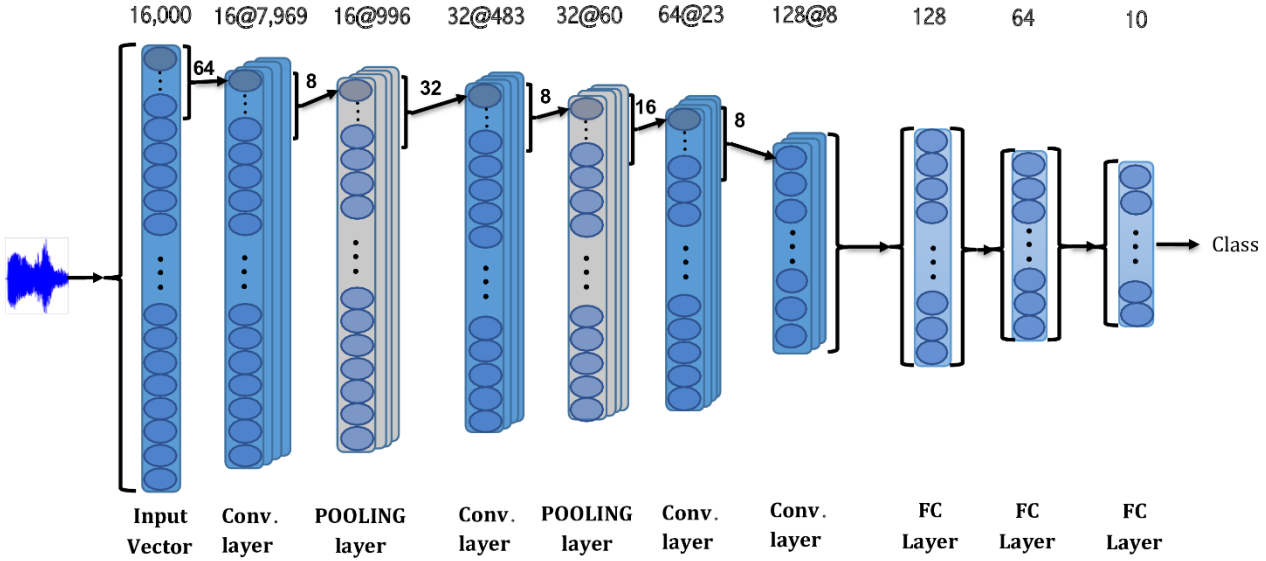
```
early_stopping = keras.callbacks.EarlyStopping(monitor="val_loss",
                                               min_delta=0,
                                               patience=5,
                                               restore_best_weights=True)

for idx, num_layers in enumerate(range(25)):
    # define model
    model = keras.Sequential()
    model.add(keras.layers.Input(shape=x_train[0].shape))
    model.add(keras.layers.Flatten())
    for l in range(num_layers):
        model.add(keras.layers.Dense(units=512, activation='elu'))
    model.add(keras.layers.Dense(units=num_classes, activation='softmax'))

    # compile model
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

    # train model
    history = model.fit(x_train, y_train,
                        batch_size=256,
                        epochs=100,
                        validation_data=(x_test, y_test),
                        callbacks=[early_stopping])

    #
    train_loss[idx], train_acc[idx] = model.evaluate(x_train, y_train, verbose=0)
    valid_loss[idx], valid_acc[idx] = model.evaluate(x_test, y_test, verbose=0)
```

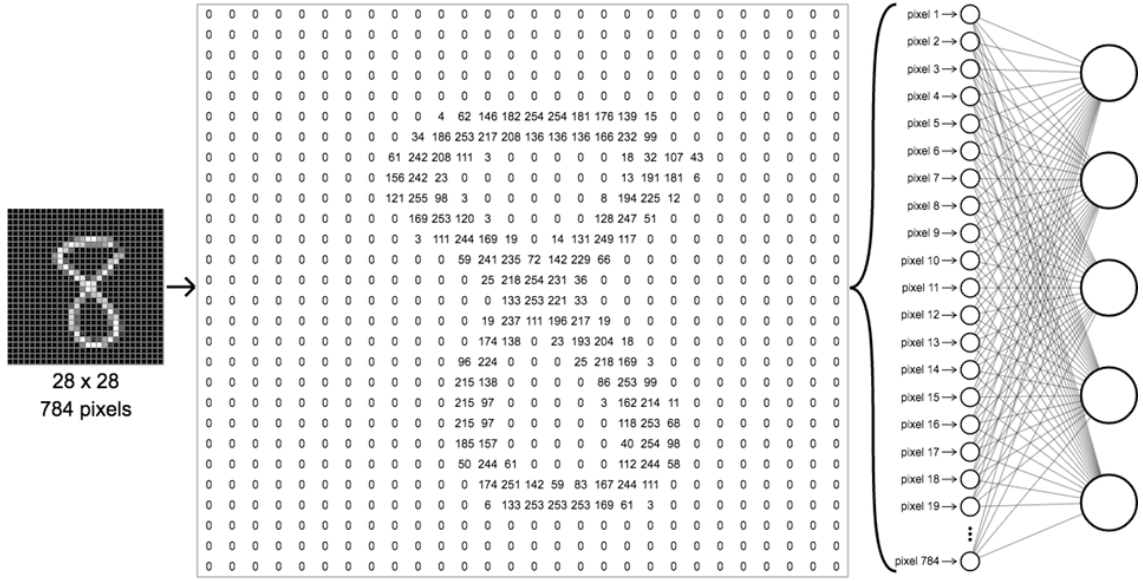

loss
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1.0
0
5
10
15
20
25
layers
train
valid


accuracy
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0
5
10
15
20
25
layers
train
valid

# شبکه‌های کانولوشنی

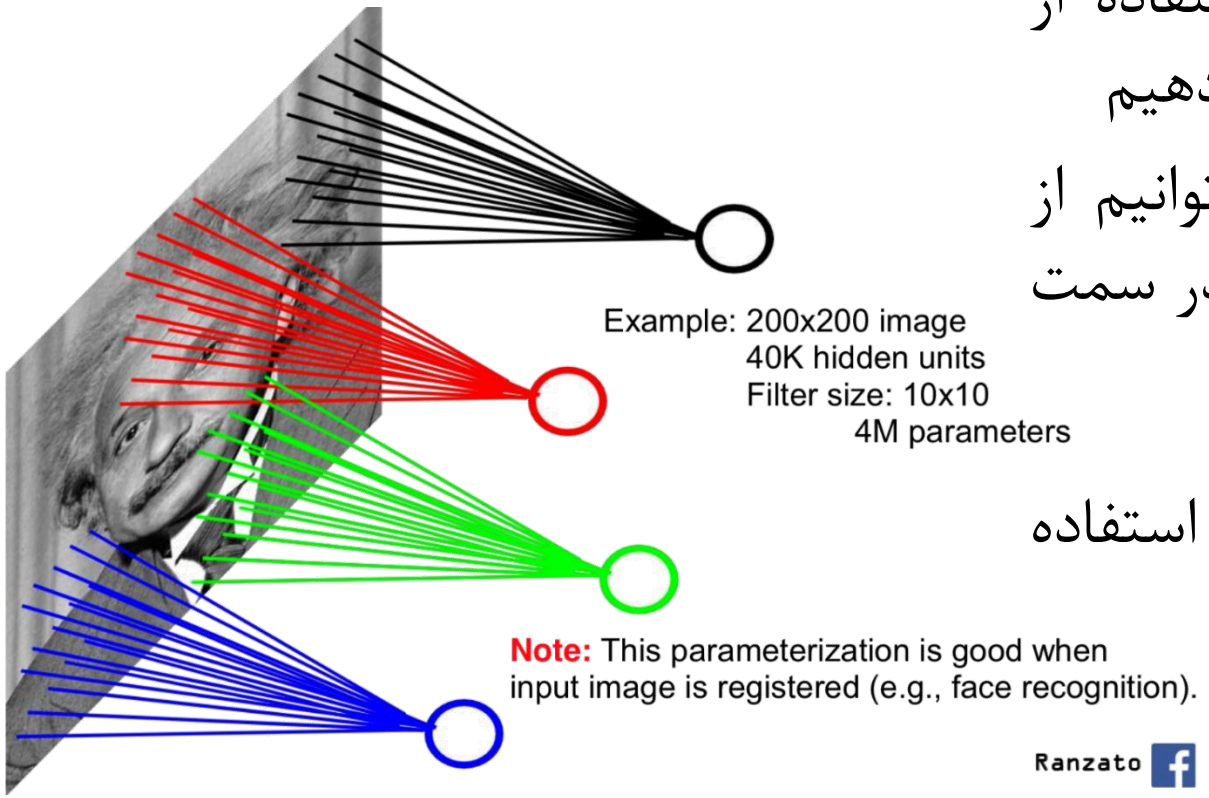
- شبکه‌های عصبی کانولوشنی (CNNs) نوع خاصی از NNها هستند که برای پردازش داده‌هایی که دارای توپولوژی شبکه‌ای شناخته‌شده‌ای هستند مناسب‌اند
- کانولوشن یک عمل خطی خاص است



# لایه‌های متصل محلّی

- بسیاری از ویژگی‌هایی که چشم انسان به راحتی می‌تواند تشخیص دهد، ویژگی‌های محلی هستند
- ما می‌توانیم لبه‌ها، بافت‌ها و حتی شکل‌ها را با استفاده از شدت پیکسل‌ها در ناحیه کوچکی از تصویر تشخیص دهیم
- اگر می‌خواهیم یک ویژگی را تشخیص بدهیم، می‌توانیم از همان آشکارساز در گوشه پایین سمت چپ تصویر و در سمت راست بالای تصویر استفاده کنیم
- ما می‌توانیم از وزن‌های یکسان در هر مکان از تصویر استفاده کنیم
  - اشتراک وزن‌ها (weight sharing)



# کانولوشن و همبستگی

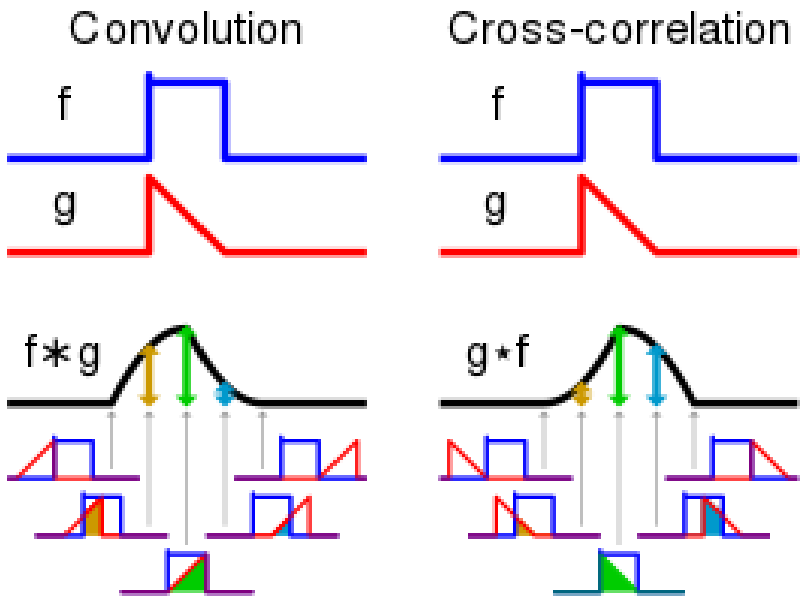
- بسیاری از کتابخانه‌های ML همبستگی متقابل را پیاده‌سازی می‌کنند اما آن را کانولوشن می‌نامند!
- الگوریتم یادگیری مقادیر مناسب هسته را در مکان مناسب یاد می‌گیرد

$$S(i) = (I * K)(i) = \sum_m I(i-m)K(m)$$

$$S(i) = (I \star K)(i) = \sum_m I(i+m)K(m)$$

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n)$$

$$S(i,j) = (I \star K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n)$$




https://en.wikipedia.org/wiki/Cross-correlation

# کانولوشن

$G_y$

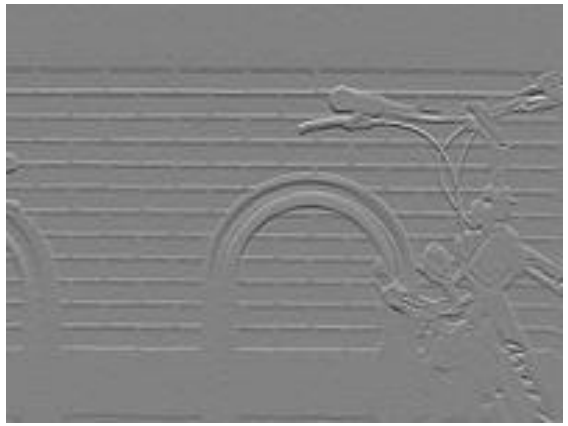
| | | |
|---|---|---|
| +1 | 0 | -1 |
| +2 | 0 | -2 |
| +1 | 0 | -1 |

| | | | | |
|---|---|---|---|---|
| $3_0$ | $3_1$ | $2_2$ | 1 | 0 |
| $0_2$ | $0_2$ | $1_0$ | 3 | 1 |
| $3_0$ | $1_1$ | $2_2$ | 2 | 3 |
| 2 | 0 | 0 | 2 | 2 |
| 2 | 0 | 0 | 0 | 1 |

| | | |
|---|---|---|
| 12.0 | 12.0 | 17.0 |
| 10.0 | 17.0 | 19.0 |
| 9.0 | 6.0 | 14.0 |

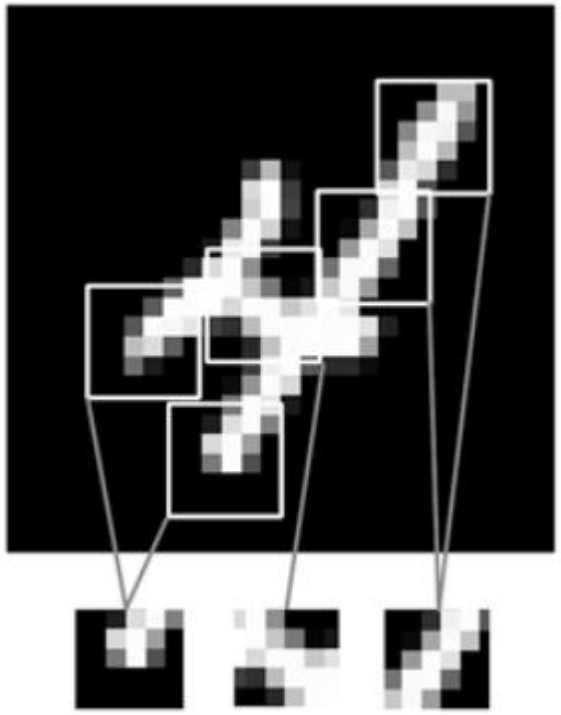
$G_x$

| | | |
|---|---|---|
| +1 | +2 | +1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |



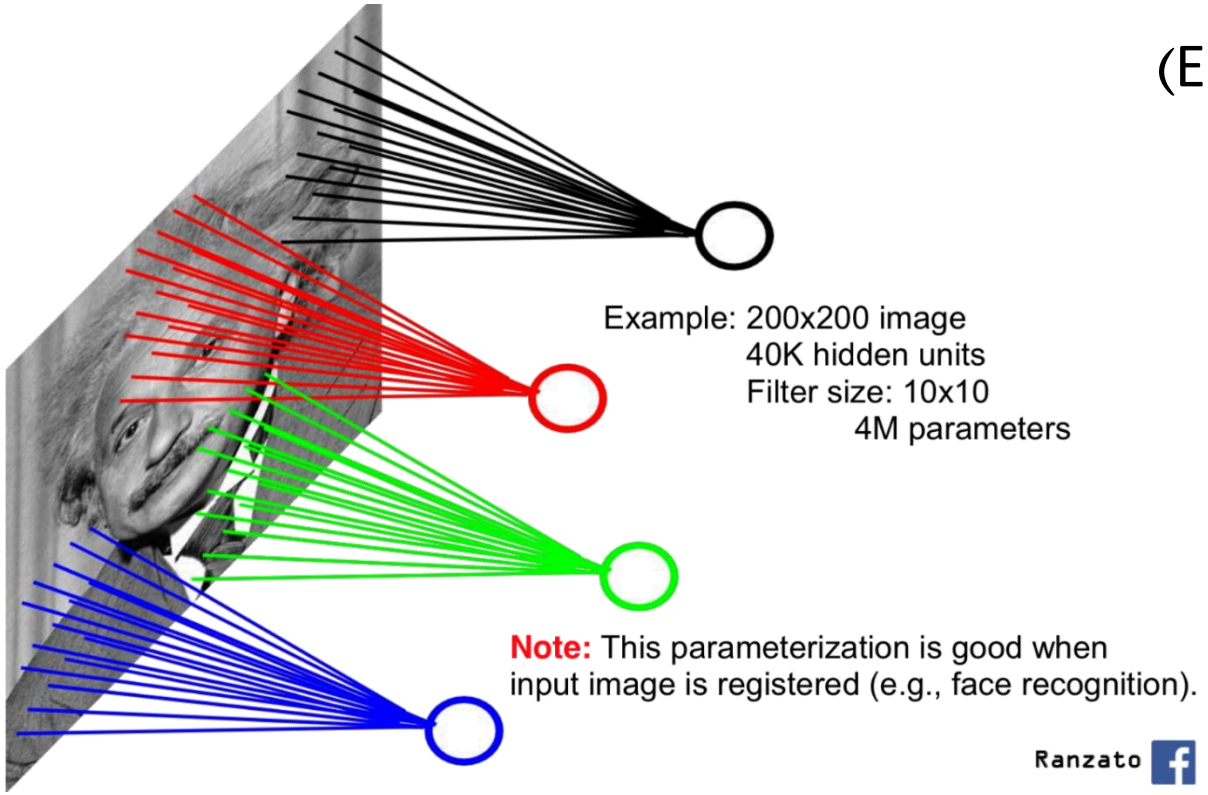
# لایه Dense در مقایسه با لایه Conv

- لایه‌های Dense الگوهای سراسری را در فضای ویژگی ورودی خود می‌آموزند
  - به عنوان مثال، برای یک رقم MNIST، الگوهایی که شامل همه پیکسل‌ها هستند
- لایه‌های Conv الگوهای محلی را یاد می‌گیرند



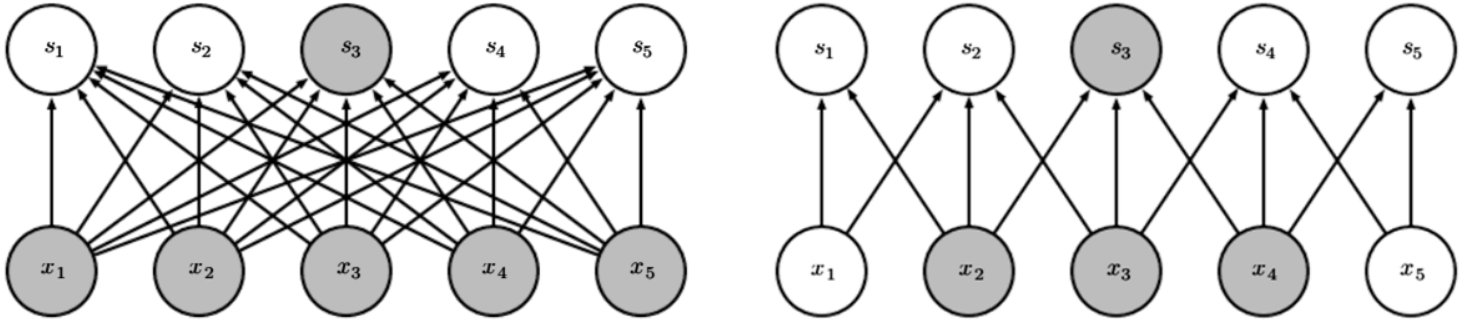
# لایه کانولوشنی

- اتصالات تنک (Sparse interactions)
- اشتراک وزن‌ها (Parameter sharing)
- بازنمایی‌های هم‌تغییر (Equivariant representations)
- توانایی کار با ورودی‌های با ابعاد مختلف



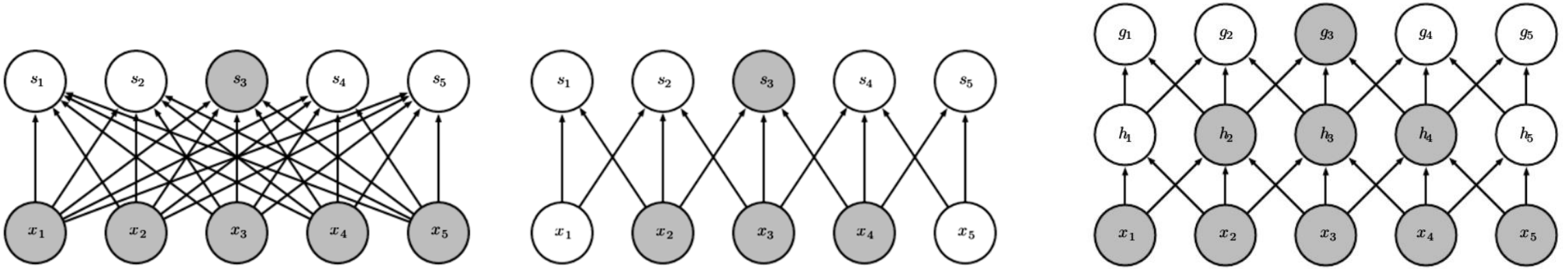
# اتصال محلّی

- در لایه‌های کاملا متصل، هر واحد خروجی به تمام واحدهای ورودی متصل است
- در لایه‌های کانولوشنی، معمولاً هر واحد تنها به برخی از واحدهای ورودی متصل است
- برای مثال، هنگام پردازش یک تصویر، ورودی ممکن است هزاران یا میلیون‌ها پیکسل داشته باشد، اما می‌توانیم ویژگی‌های کوچک و بامعنایی مانند لبه‌ها را با هسته‌هایی که فقط ده‌ها یا صدها پیکسل را استفاده می‌کنند، تشخیص بدهیم
  - باید پارامترهای بسیار کمتری را ذخیره کنیم



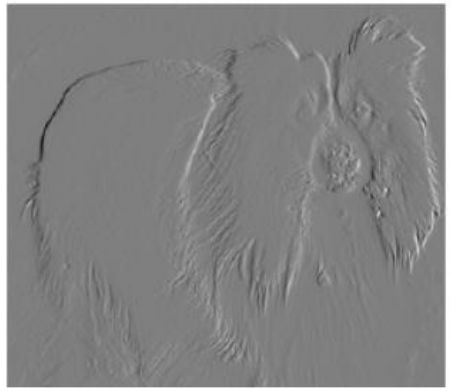
# اتصال محلّی

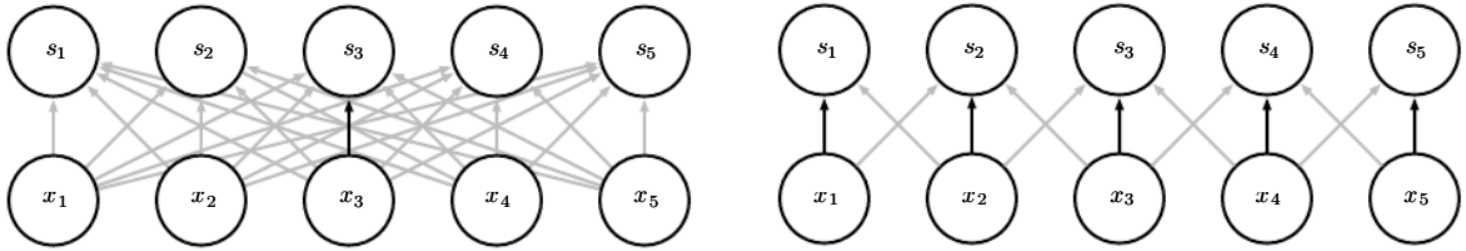
- در یک شبکه کانولوشنی عمیق، واحدها در لایه‌های عمیق‌تر ممکن است به طور غیرمستقیم به بخش بزرگ‌تری از ورودی وابستگی داشته باشند
- این کار به شبکه اجازه می‌دهد تا بتواند ویژگی‌های پیچیده را به صورت سلسله‌مراتبی و با استفاده از چنین اتصالات محلّی بیاموزد
- میدان تاثیر واحدها در لایه‌های عمیق‌تر می‌تواند بسیار بزرگ باشد



# اشتراک وزن‌ها

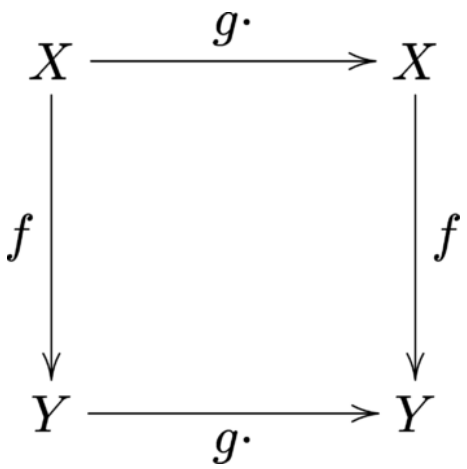
- اشتراک پارامتر به استفاده از یک پارامتر برای بیش از یک تابع در یک مدل اشاره دارد
- در یک لایه کانولوشنی، هر یک از پارامترهای هسته در هر موقعیت از ورودی استفاده می‌شود
- در برخی موارد، ممکن است ما نخواهیم پارامترها را به اشتراک بگذاریم!





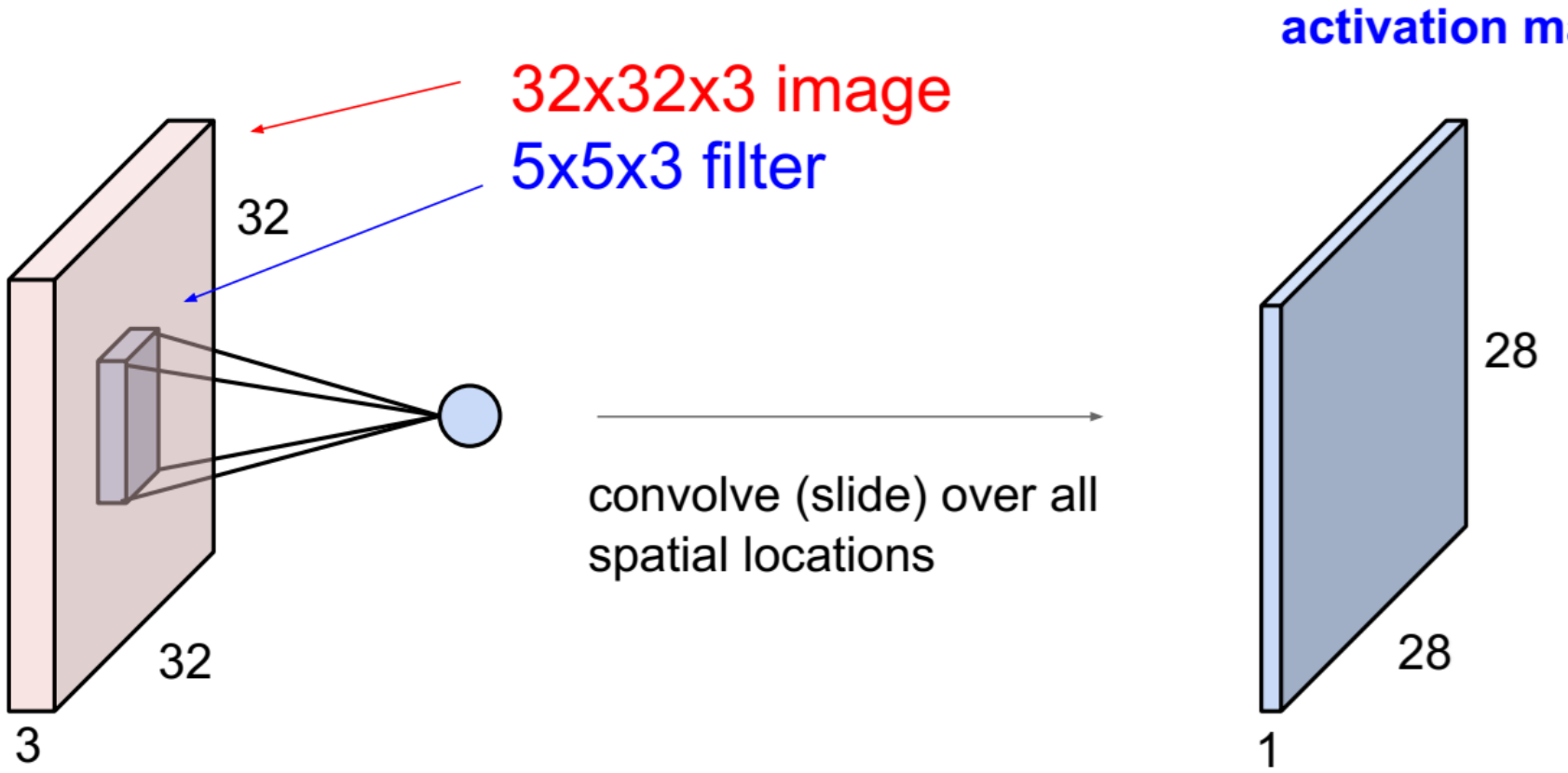
# بازنمایی‌های هم‌تغییر

- هم‌تغییر بودن یک تابع به این معنی است که اگر ورودی تغییر کند، خروجی نیز به همان صورت تغییر می‌کند
- در مورد کانولوشن، شکل خاص به اشتراک‌گذاری پارامترها باعث می‌شود که لایه نسبت به جابجایی هم‌تغییر شود
- کانولوشن به طور طبیعی نسبت به برخی از تبدیل‌های دیگر، مانند تغییر در مقیاس یا چرخش یک تصویر، هم‌تغییر نیست




$$f\big(g(\boldsymbol{x})\big) = g\big(f(\boldsymbol{x})\big)$$

# لایه کانولوشنی

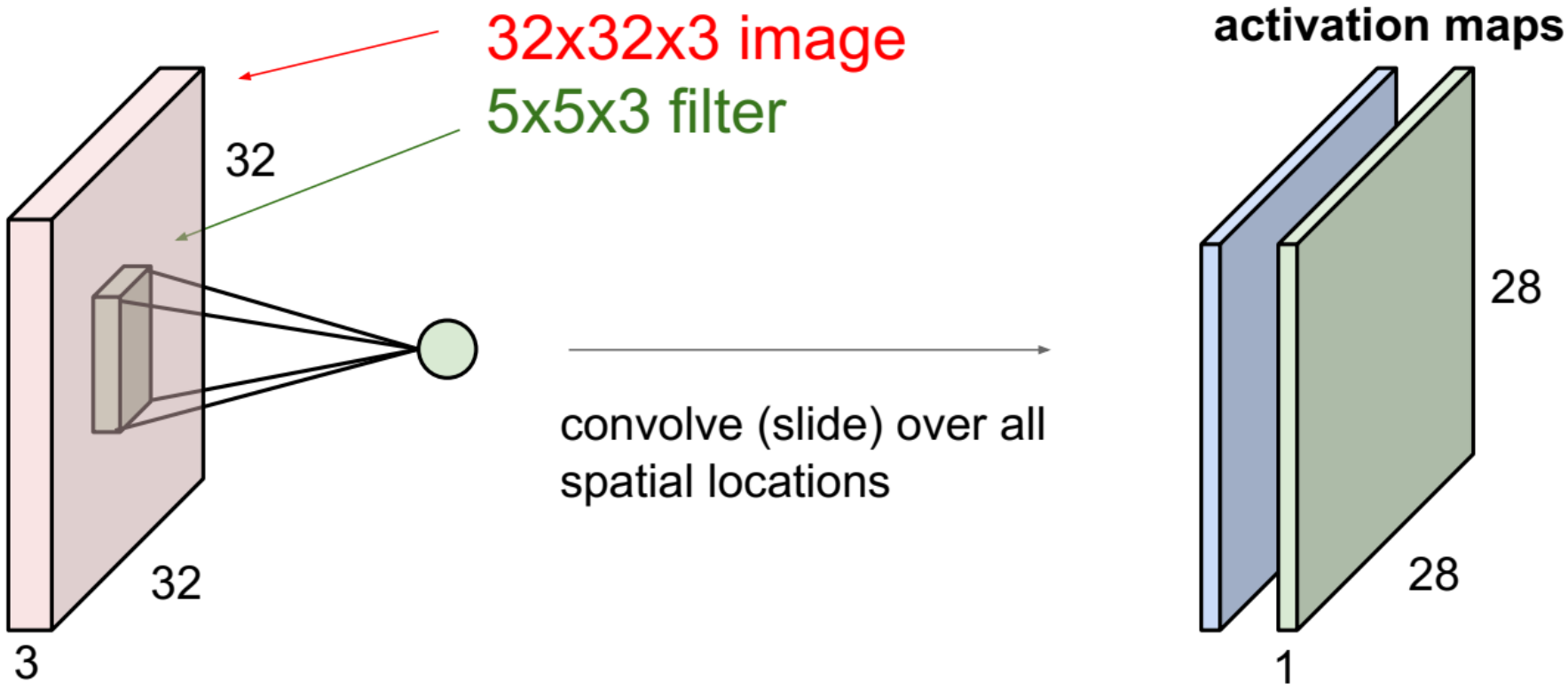





http://cs231n.stanford.edu/slides/2020/lecture_5.pdf

# لایه کانولوشنی

- یک فیلتر دوم را در نظر بگیرید (رنگ سبز)



# لایه کانولوشنی

- در لایه کانولوشن از چند فیلتر مجزا استفاده می کنیم