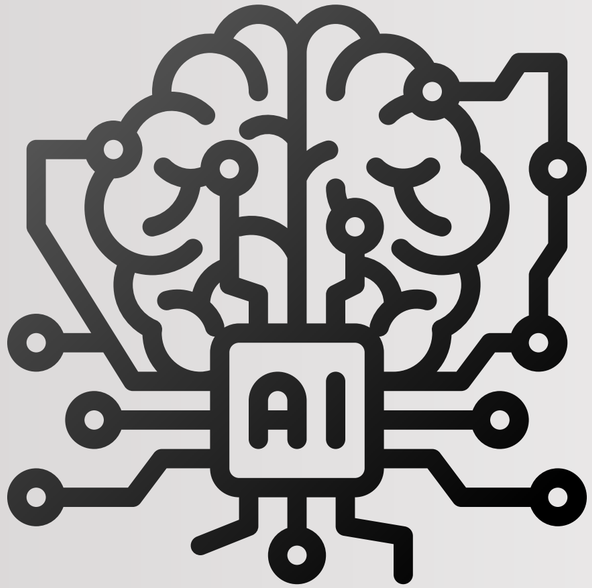
10.HAFTA

YAPAY SİNİR AĞLARI

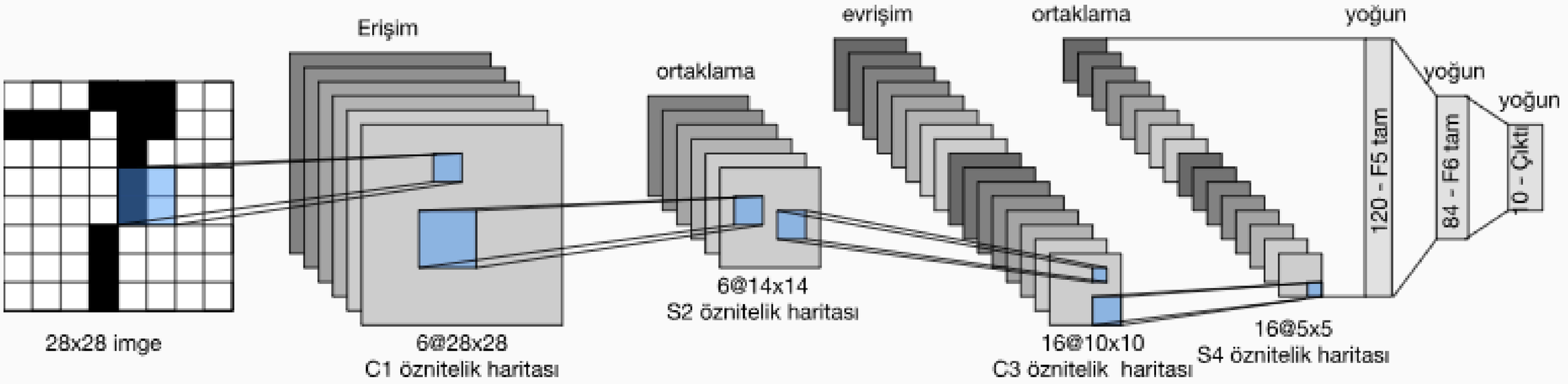


SAMSUN ÜNİVERSİTESİ

DR. ÖĞR. ÜYESİ ALPER TALHA KARADENİZ

Evrişimli Sinir Ağları (Convolutional Neural Networks - CNN)

Evrişimli Sinir Ağları (CNN), yapay sinir ağlarının özel bir türüdür ve özellikle görüntü, video ve hatta zaman serisi gibi **uzamsal ve zamansal ilişkiler** içeren verilerde yüksek başarı gösterir.
Klasik yapay sinir ağlarında her nöron, önceki katmandaki tüm nöronlarla bağlantılıdır. Bu durum görüntü gibi çok büyük boyutlu verilerde **çok fazla parametre** ortaya çıkarır. CNN ise bu sorunu **evrişim (convolution)** adı verilen özel bir matematiksel işlemle çözer.


Erişim
ortaklama
evrişim
ortaklama
yoğun
yoğun
yoğun
120 - F5 tam
84 - F6 tam
10 - Çıktı
28x28 imge
6@28x28
C1 öznitelik haritası
6@14x14
S2 öznitelik haritası
16@10x10
C3 öznitelik haritası
16@5x5
S4 öznitelik haritası

CNN’ler, verinin tümünü bir anda değil, küçük parçalarını işleyerek özellikler çıkarır. Bu sayede:

- Parametre sayısı azalır,
- Eğitim daha verimli olur,
- Model, görüntülerdeki yerel desenleri öğrenebilir.

Evrişim (Convolution) İşlemi

Evrişim, CNN'in kalbidir. Matematikte, evrişim işlemi şu şekilde tanımlanır:

$$S(i,j) = \sum_m \sum_n X(i+m, j+n) \cdot K(m,n)$$

Burada:

- X: giriş görüntüsü,
- K: filtre (kernel),
- S: çıktı özelliği (feature map).

Mantık: Küçük bir filtre matrisi, girişin üzerinde kaydırılır. Her konumda çarpım-toplam işlemi yapılır ve bu değer, o konumdaki yeni özellik haritasına yazılır.

Stride

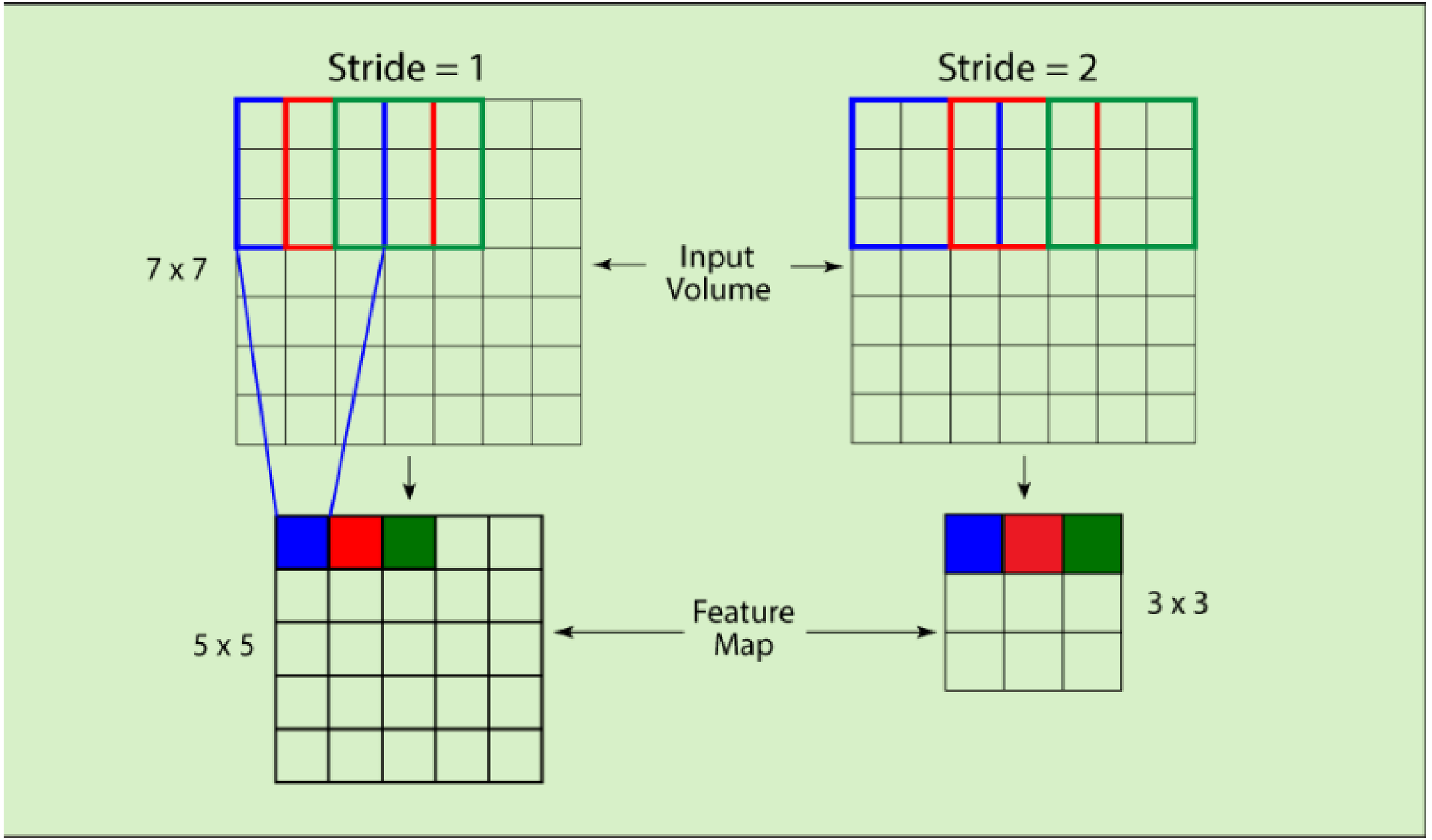
Filtrenin kaydırılma adımıdır. Stride büyük olursa çıktı boyutu küçülür.

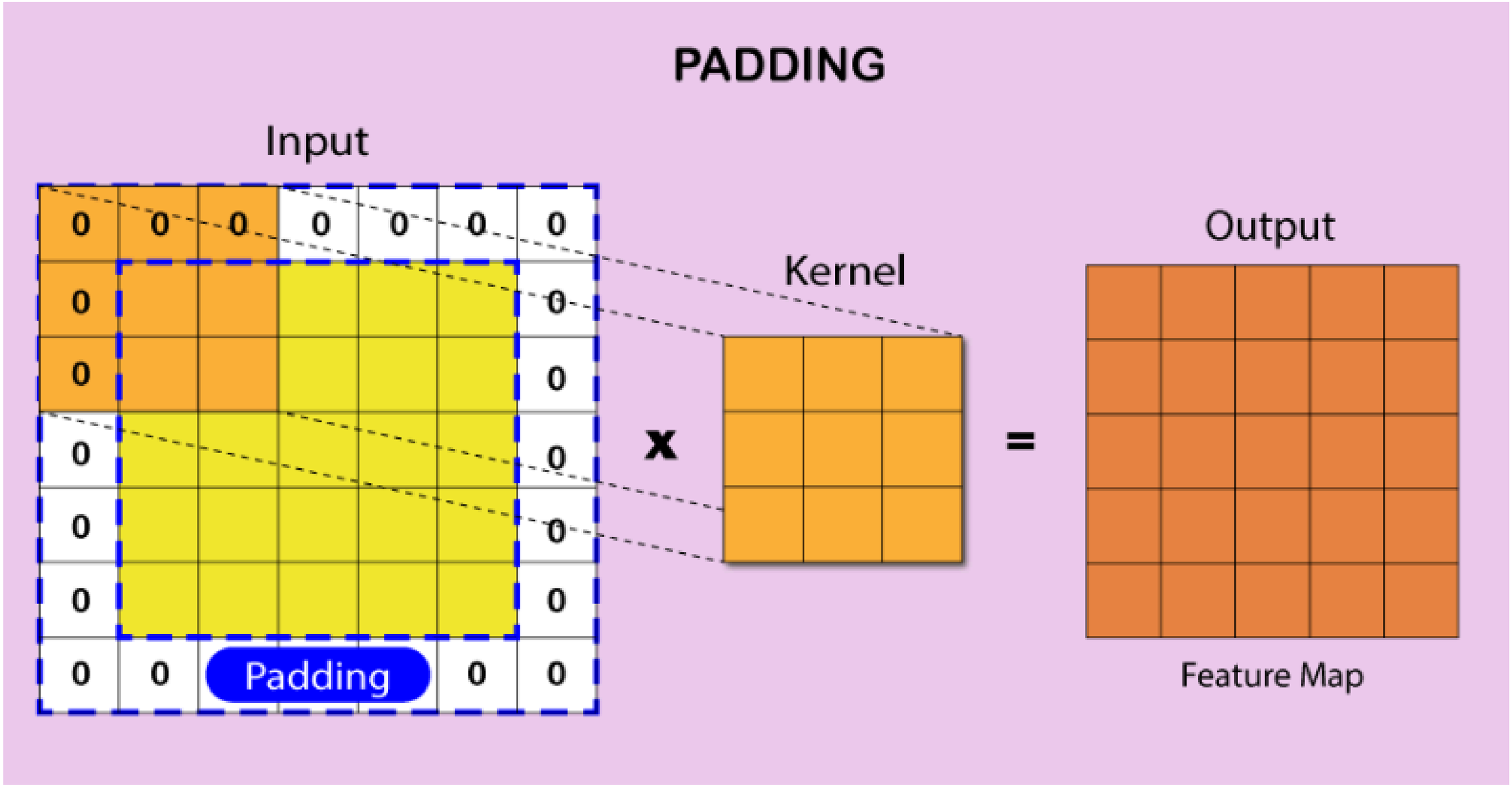
Padding

Filtrenin girişin kenarlarına da uygulanabilmesi için görüntünün etrafına sıfır değerleri eklenmesidir. Bu sayede boyut korunabilir.

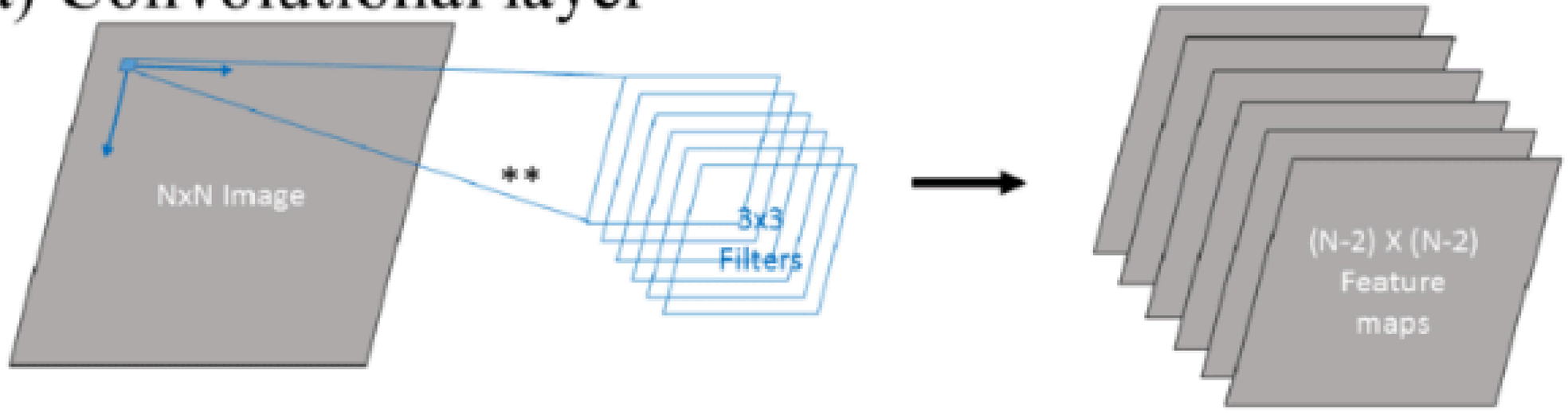
Örnek:

- Girdi: 32×32 boyutlu bir görüntü,
- Filtre: 5×5, stride = 1, padding = 0.
  Çıktı boyutu: $(32 - 5)/1 + 1 = 28$.
  Yani çıktı 28×28 olur.


Stride = 1
Stride = 2
7 x 7
Input
Volume
5 x 5
Feature
Map
3 x 3


PADDING
Input
0 0 0 0 0 0 0
0
0
0
0
0
0
0
0 0 0 0 0
Padding
x
Kernel
=
Output
Feature Map

(a) Convolutional layer



(b) Convolution operation with 3x3 filter

129	132	127	148	89	101
78	172	87	56	87	104
55	67	202	228	19	34
134	159	77	87	89	108
14	15	4	157	102	99
202	255	199	33	67	103

**

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

=

129	132	127	148	89	101
78	-1	0	+1	87	104
55	-2	0	+2	19	34
134	-1	0	+1	89	108
14	15	4	157	102	99
202	255	199	33	67	103

-1 * 172 -2 *67 -1 * 159
+ 0 * 87 + 0 * 202 + 0 * 77
+1 * 56 +2 * 228 + 1 *87 = 134

Aktivasyon Fonksiyonu

Evrişimden sonra elde edilen değerler doğrusaldır. Sinir ağlarının karmaşık ilişkileri öğrenebilmesi için doğrusal olmayan fonksiyonlar gerekir.

- En sık kullanılan aktivasyon: **ReLU (Rectified Linear Unit)**

$$f(x) = \max(0, x)$$

- ReLU, negatif değerleri sıfırlar, pozitifleri olduğu gibi geçirir.
- Bu sayede model daha hızlı öğrenir ve gradyan kaybolması (vanishing gradient) sorununu azaltır.

Havuzlama (Pooling)

CNN'in bir diğer önemli parçası **havuzlama katmanıdır**. Amaç:

1. Çıktı boyutunu küçültmek,
2. Hesaplama maliyetini azaltmak,
3. Daha genelleştirilebilir öznitelikler elde etmek.

Max Pooling

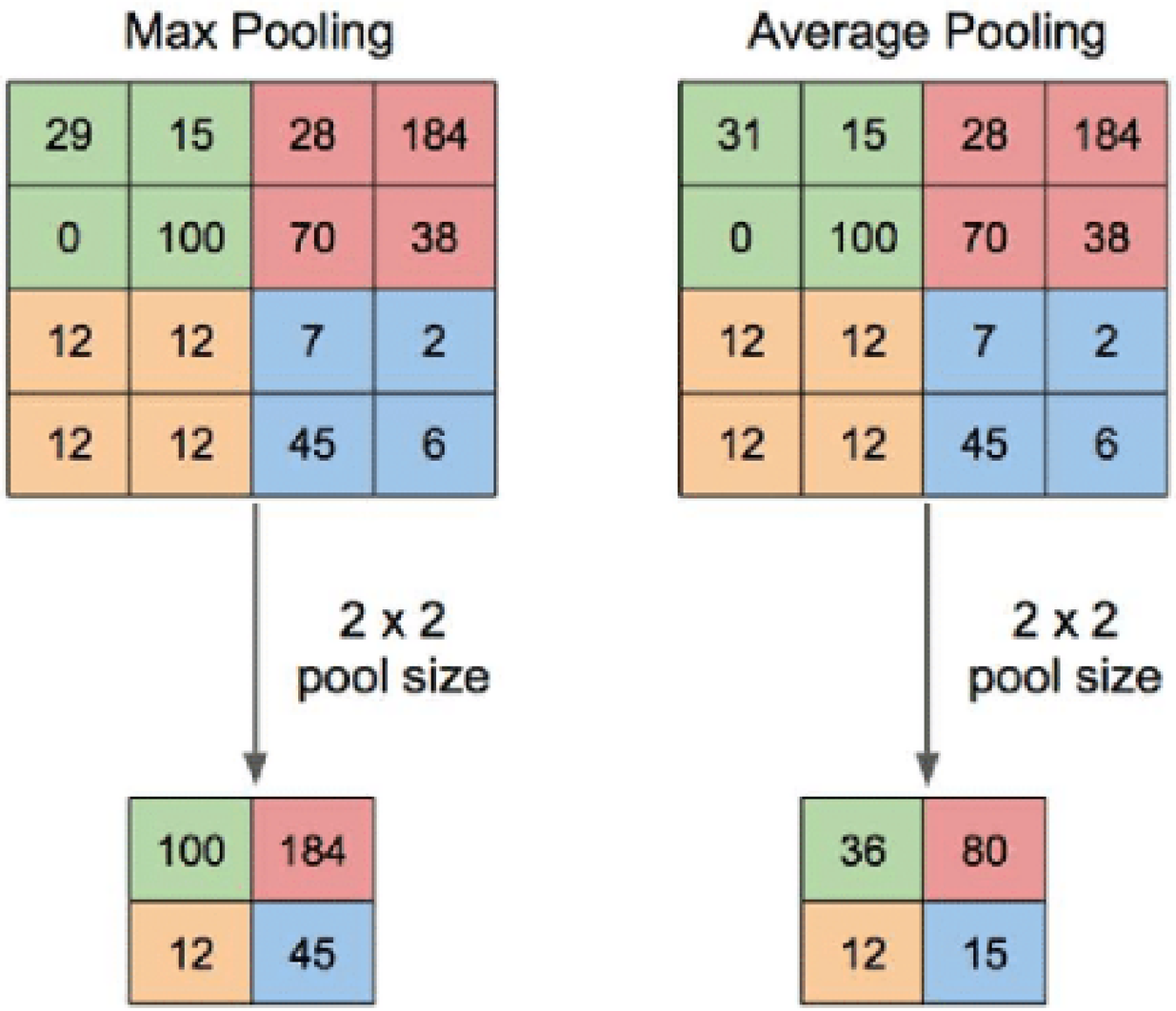
Belirli bir bölgede (örneğin 2x2) en büyük değeri alır.

Average Pooling

Belirli bir bölgedeki ortalama değeri alır.

Örnek:

Bir 4×4 özellik haritasına 2×2 max pooling uygulanırsa, sonuç 2×2 boyutunda olur.


Max Pooling
29 15 28 184
0 100 70 38
12 12 7 2
12 12 45 6
2 x 2
pool size
100 184
12 45
Average Pooling
31 15 28 184
0 100 70 38
12 12 7 2
12 12 45 6
2 x 2
pool size
36 80
12 15

Flatten Katmanı

Flatten katmanı, çok boyutlu bir girdi tensörünü tek boyutlu bir vektöre dönüştüren özel bir katmandır. Bu katman, genellikle evrişimsel sinir ağlarında (CNN), evrişim (convolution) ve havuzlama (pooling) katmanlarından sonra kullanılır. Amaç, uzamsal (görsel) bilgiyi tek boyutlu hale getirerek tam bağlı (Dense) katmanlara aktarabilmektir.

- Evrişim ve havuzlama katmanları, çıktıyı genellikle 3 boyutlu (yükseklik × genişlik × kanal sayısı) bir tensör halinde üretir.
- Ancak Dense katmanlar, sadece tek boyutlu giriş vektörleri kabul eder.
- İşte bu noktada Flatten katmanı devreye girer. Çok boyutlu çıktıyı “düzleştirerek” tek boyutlu bir vektöre çevirir.

Örnek:

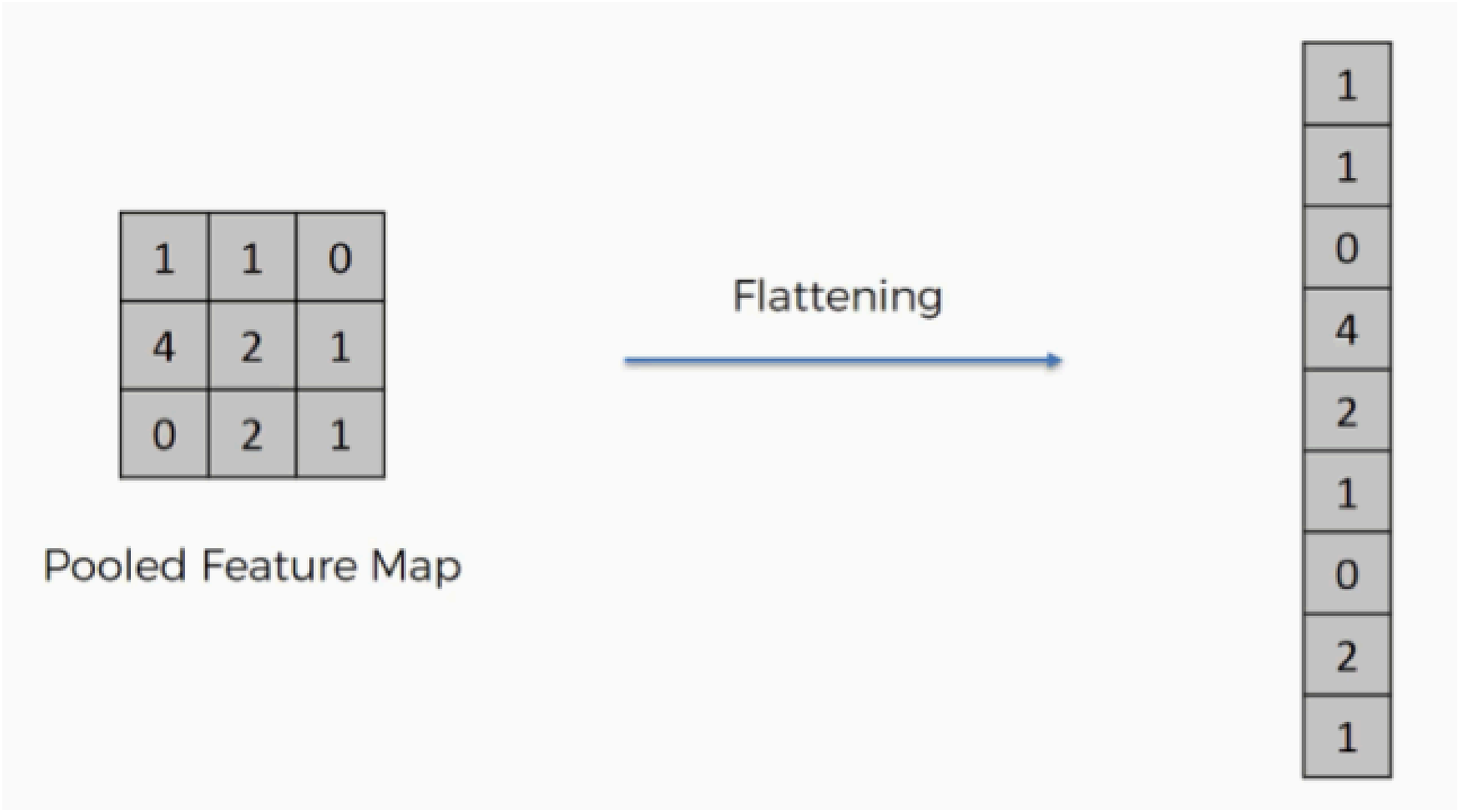
Diyelim ki bir evrişim katmanından sonra elimizde şu boyutta bir çıktı var:

$$4 \times 4 \times 3$$

- Yani: **yükseklik = 4, genişlik = 4, kanal sayısı = 3**
- Toplam eleman sayısı = $4 \times 4 \times 3 = 48$

Flatten katmanı bu çıktıyı şu hale getirir:

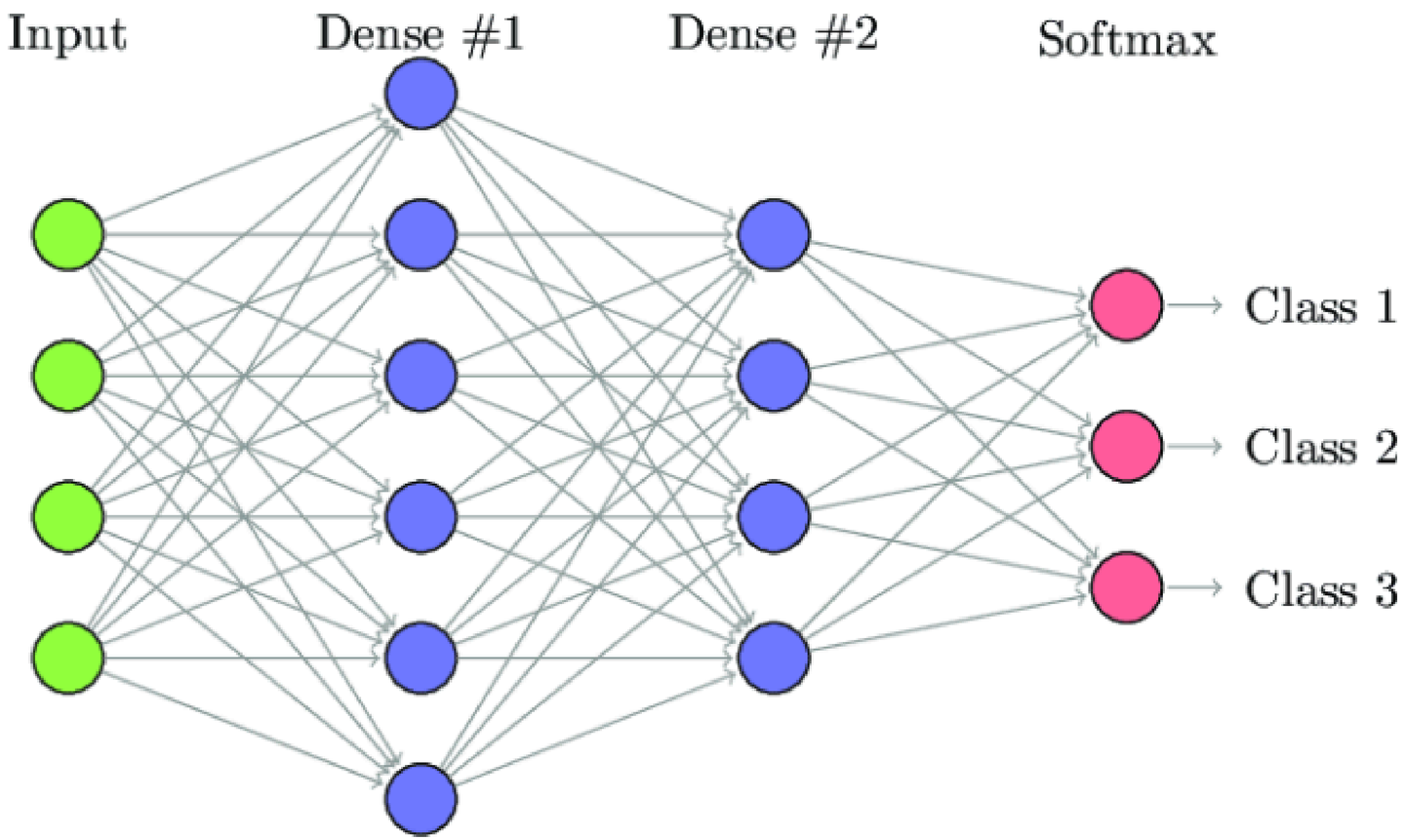
$$[48 \text{ elemanlı tek boyutlu vektör}]$$


1 1 0
4 2 1
0 2 1
Pooled Feature Map
Flattening
1
1
0
4
2
1
0
2
1

Tam Bağlantılı (Dense) Katmanlar

- Evrişim ve havuzlama işlemlerinden sonra elde edilen özellik haritaları vektör hâline getirilir.
- Bu vektör, bir veya daha fazla tam bağlantılı katmana aktarılır.
- Son katmanda genellikle softmax aktivasyonu bulunur. Bu fonksiyon her sınıf için bir olasılık değeri üretir.

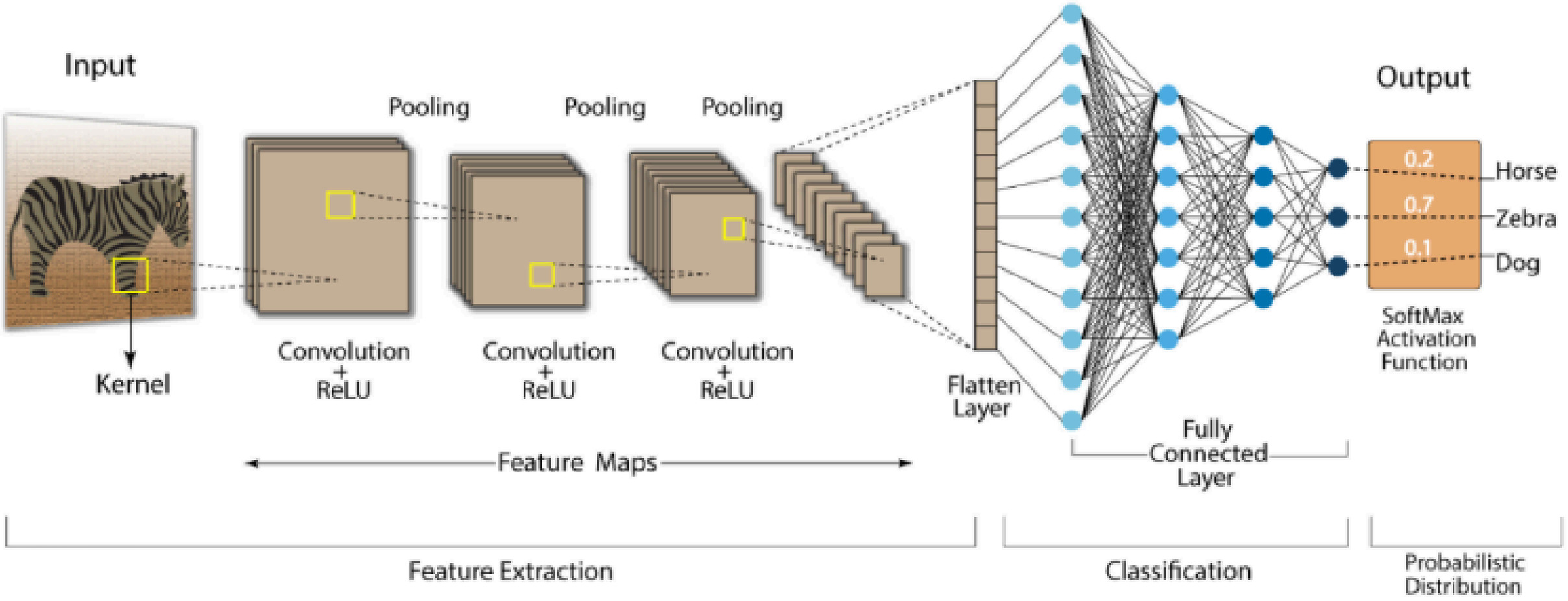
- Fully Connected katmanda birkaç kez evrişimli katmandan ve pooling katmanından geçen ve matris halinde olan görselimiz düz bir vektör haline getirilir.



Katmanların Birleşimi

Bir CNN tipik olarak şu sıralamayla çalışır:

- **Evrişim Katmanı:** Girdi görüntüden özellikler çıkarır.
- **Aktivasyon Fonksiyonu:** Doğrusal olmayanlık ekler.
- **Havuzlama Katmanı:** Boyutu küçültür.
- Bu blok birkaç kez tekrar eder.
- **Tam Bağlantılı Katmanlar:** Elde edilen öznitelikleri kullanarak sınıflandırma yapar.


Convolution Neural Network (CNN)
Input
Pooling
Pooling
Pooling
Output
0.2
Horse
0.7
Zebra
0.1
Dog
Kernel
Convolution + ReLU
Convolution + ReLU
Convolution + ReLU
SoftMax Activation Function
Flatten Layer
Feature Maps
Fully Connected Layer
Feature Extraction
Classification
Probabilistic Distribution

Görsel Veri ile Çalışmanın Avantajları

Görsel veri, yapay zekâ ve makine öğrenmesi uygulamalarında sıklıkla kullanılan veri türlerinden biridir. Görseller, gerçek dünyadaki nesnelerin, olayların ve ilişkilerin doğrudan temsilini sağladığından, çeşitli alanlarda etkili çözümler üretmek için önemli bir veri kaynağıdır.

1. İnsan Algısına Uygunluk

Görsel veri, insan algısının temel mekanizmalarına uygun bir veri türüdür. İnsanlar çevrelerini büyük ölçüde görsel bilgiler üzerinden algıladığından, görsel veriler üzerinde gerçekleştirilen analizler, insanlar tarafından daha kolay yorumlanabilir ve anlaşılabilir sonuçlar üretir.

2. Yüksek Bilgi Yoğunluğu

Görsel veriler, tek bir anda çok sayıda bilgiyi taşıyabilir; renk, şekil, boyut, konum ve bağlamsal özellikler aynı veri seti içinde bulunabilir. Bu özellik, görsel verileri metin veya sayısal verilerden ayıran en önemli avantajlardan biridir.

3. Otomasyon ve Verimlilik

Görsel veri işleme teknikleri, insan müdahalesine olan gereksinimi azaltarak süreçleri otomatik hâle getirir. Örneğin; üretim hatlarında kalite kontrol, tıbbi görüntü analizi ve güvenlik sistemlerinde yüz tanıma uygulamaları, görsel veriler kullanılarak etkin biçimde yürütülebilir.

4. Çoklu Uygulama Alanı

Görsel veri, farklı sektörlerde geniş bir uygulama yelpazesine sahiptir:

- Sağlık: Tıbbi görüntülerin analizi ile hastalık tanısı.
- Güvenlik: Yüz ve plaka tanıma sistemleri.
- Otonom Araçlar: Çevresel algılama ve nesne takibi.
- Perakende: Ürün tanıma ve müşteri davranış analizi.

5. Derin Öğrenme Yöntemleri ile Uyum

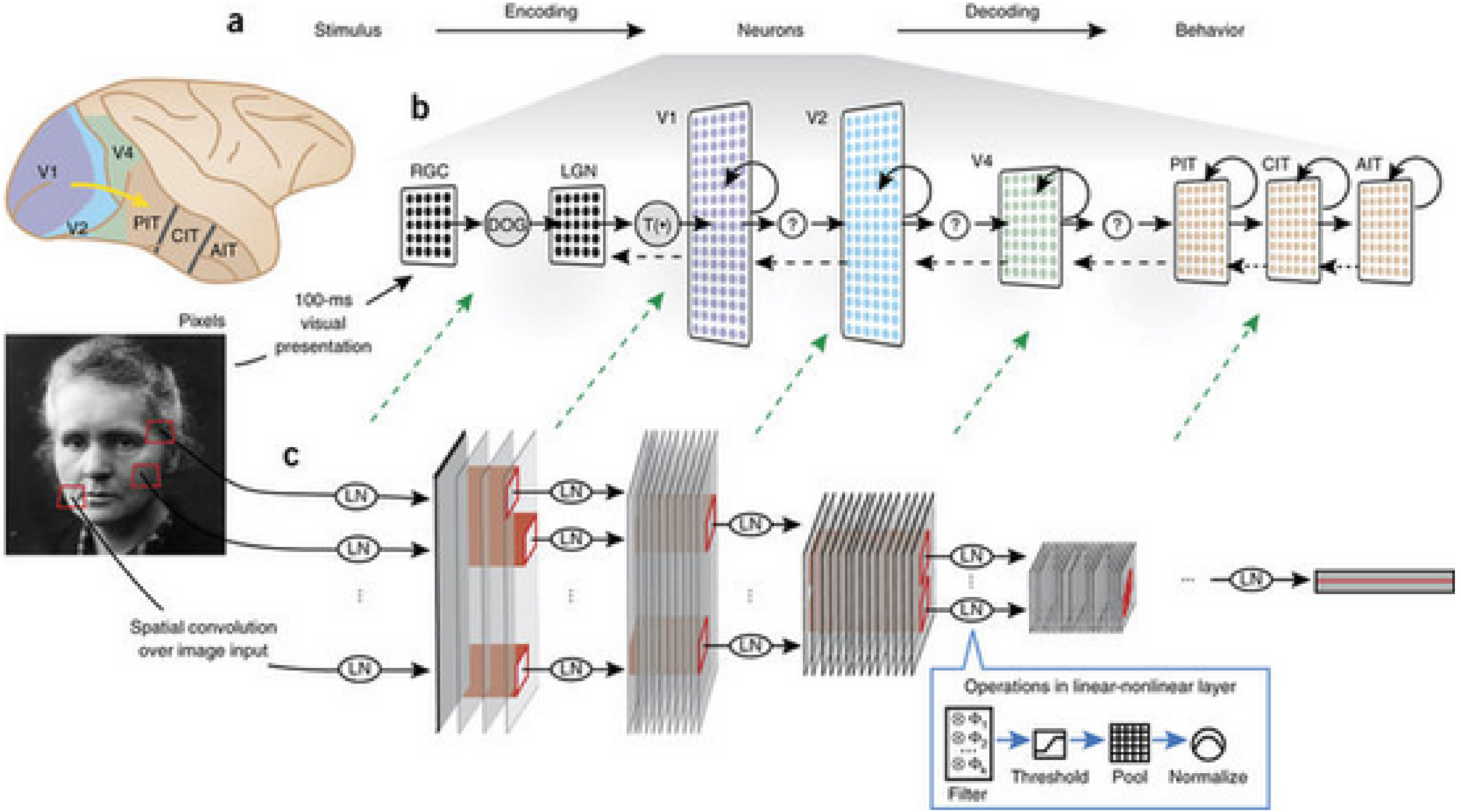
Görsel veriler, derin öğrenme algoritmalarının etkin biçimde uygulanabilmesine uygun yapıya sahiptir. Büyük hacimli görsel veri setleri, evrişimli sinir ağları ve diğer derin öğrenme modelleri ile işlenerek yüksek doğrulukta sınıflandırma ve tahmin sonuçları elde edilmesini sağlar.

6. Anlamlı ve Yorumlanabilir Çıktılar

Görsel veriler ile çalışan sistemler, çıktıları doğrudan kullanıcıya anlaşılır biçimde sunabilir. Örneğin, nesne tespiti uygulamalarında, algılanan nesneler üzerinde konum kutuları veya etiketler gösterilerek kullanıcıya açıklayıcı bilgi sağlanabilir.

7. Karmaşık Problemlerin Basitleştirilmesi

Metinsel veya sayısal verilerle ifade edilmesi zor olan durumlar, görsel veri aracılığıyla daha hızlı ve doğru biçimde analiz edilebilir. Örneğin, trafik yoğunluğunu sayısal verilerle modellemek yerine görüntü verisi üzerinden değerlendirmek, durumu daha net ve anlaşılır hâle getirir.


a
Stimulus
Encoding
Neurons
Decoding
Behavior
V1
V4
V2
PIT
CIT
AIT
b
RGC
DOG
LGN
T(+)
V1
V2
V4
PIT
CIT
AIT
?
100-ms
visual
presentation
Pixels
c
LN
Spatial convolution
over image input
Operations in linear-nonlinear layer
Filter
Threshold
Pool
Normalize

Uygulama: CIFAR-10 veri kümesi ile basit bir CNN modeli

```
# Gerekli kütüphaneleri yükleme
import tensorflow as tf
from tensorflow.keras import layers, models
import matplotlib.pyplot as plt

# 1. VERİYİ HAZIRLAMA
# CIFAR-10 veri setini yükle
(train_images, train_labels), (test_images, test_labels) = tf.keras.datasets.cifar10.load_data()

# Etiket isimleri (10 sınıf)
class_names = ['uçak', 'araba', 'kuş', 'kedi', 'geyik',
               'köpek', 'kurbağa', 'at', 'gemi', 'kamyon']

# Normalizasyon: Resim piksel değerlerini 0-255'ten 0-1 arasına getir
train_images = train_images / 255.0
test_images = test_images / 255.0
```

```python
# 2. CNN MODELİNİ OLUŞTURMA
model = models.Sequential()

# Konvolüsyon (Evrişim) Katmanları - Özellik çıkarımı için
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(layers.MaxPooling2D((2, 2)))  # Boyut küçültme

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))

# Yoğun (Dense) Katmanları - Sınıflandırma için
model.add(layers.Flatten())  # 2D'den 1D'ye düzleştirme
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))  # 10 sınıf çıktısı
```

```python
# 3. MODELİ DERLEME
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Model yapısını göster
model.summary()

# 4. MODELİ EĞİTME
history = model.fit(train_images, train_labels,
                    epochs=10,  # 10 kez eğitim
                    validation_data=(test_images, test_labels))
```

```python
# 5. MODELİ DEĞERLENDİRME
test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=2)
print(f"\nTest doğruluğu: {test_acc:.4f}")

# 6. ÖRNEK TAHMİN YAPMA
# İlk 5 test resmini tahmin et
predictions = model.predict(test_images[:5])

# Tahminleri göster
for i in range(5):
    predicted_class = np.argmax(predictions[i])
    actual_class = test_labels[i][0]
    print(f"Tahmin: {class_names[predicted_class]}, Gerçek: {class_names[actual_class]}")
```

Model özeti çıktısı

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 30, 30, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 13, 13, 64)        18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2D) (None, 6, 6, 64)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 4, 4, 64)          36928
_________________________________________________________________
flatten (Flatten)            (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 64)                65600
_________________________________________________________________
dense_1 (Dense)              (None, 10)                650
=================================================================
Total params: 122,570
Trainable params: 122,570
Non-trainable params: 0
```

Test sonuçları :

```
313/313 - 1s - loss: 0.8923 - accuracy: 0.6915

Test doğruluğu: 0.6915
```

Tahmin örnekleri :

```
1/1 [==============================] - 0s 74ms/step
Tahmin: kedi, Gerçek: kedi
Tahmin: gemi, Gerçek: gemi
Tahmin: gemi, Gerçek: gemi
Tahmin: at, Gerçek: at
Tahmin: araba, Gerçek: araba
```

ÖDEV: Modelde genişletme yaparak modelin tahmin doğruluk değerini artıracak yeni bir model tasarlayınız