Derin Yapay Sinir Ağları ve Derin Öğrenme'ye Kısa Bir Giriş

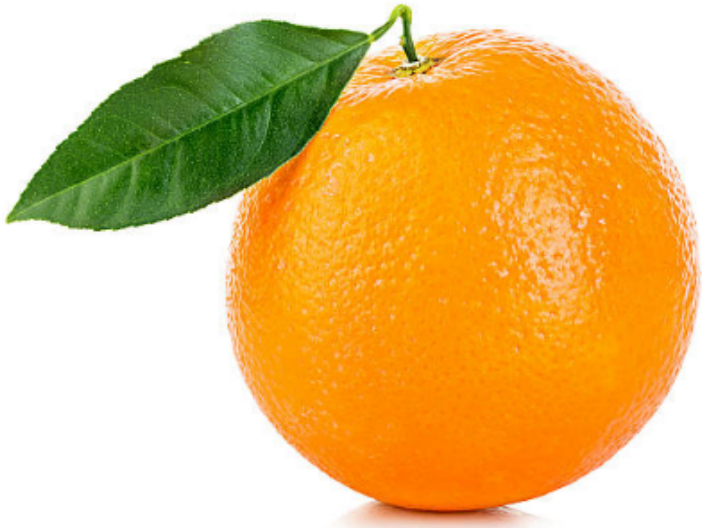
Dr. Öğr. Üyesi Emre Akbaş
Bilgisayar Mühendisliği
Orta Doğu Teknik Üniversitesi

BMO Semineri – 24 Kasım 2018

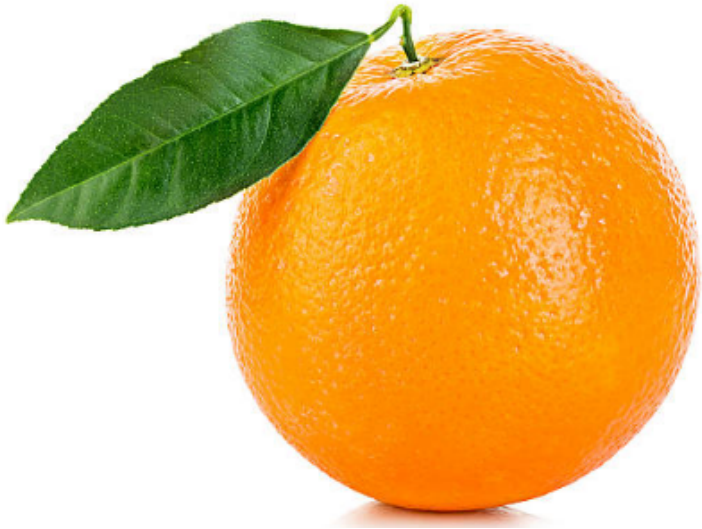
Yanıtlamaya çalışacağımız sorular

- Nedir bu “derin öğrenme”?
- Neden her yerde “derin öğrenme” görüyoruz?
- Derin öğrenme modellerinin klasik yapay sinir ağlarından ne farkı var?
- Bu modellerin uygulamaları nelerdir?
- Derin öğrenme modelleri nasıl eğitilir?
- Nereden başlayabilirim?

Makine öğrenmesi (*İng.* machine learning)



Makine öğrenmesi (*İng.* machine learning)



Verilen bir problem için veri üzerinden bir model/fonksiyon üretme

“Derin öğrenme” nedir?

- Makine öğrenmesinin (*İng. machine learning*) bir alt dalı.

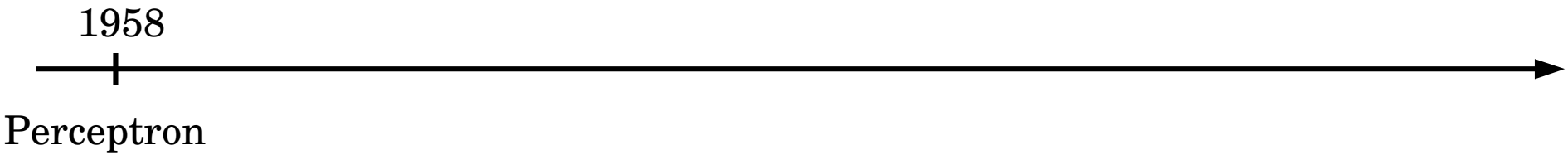
“Derin öğrenme” nedir?

- Beynin yapısal ve işlevsel özelliklerinden esinlenilerek tasarlanmış, çok katmanlı ağ yapıları (*İng. graph*) olan “yapay sinir ağları” üzerinde çalışan algoritmalar ve modeller kümesi.
- “Derin” **bir**den fazla saklı katmanı (*İng. hidden layer*) ifade eder.

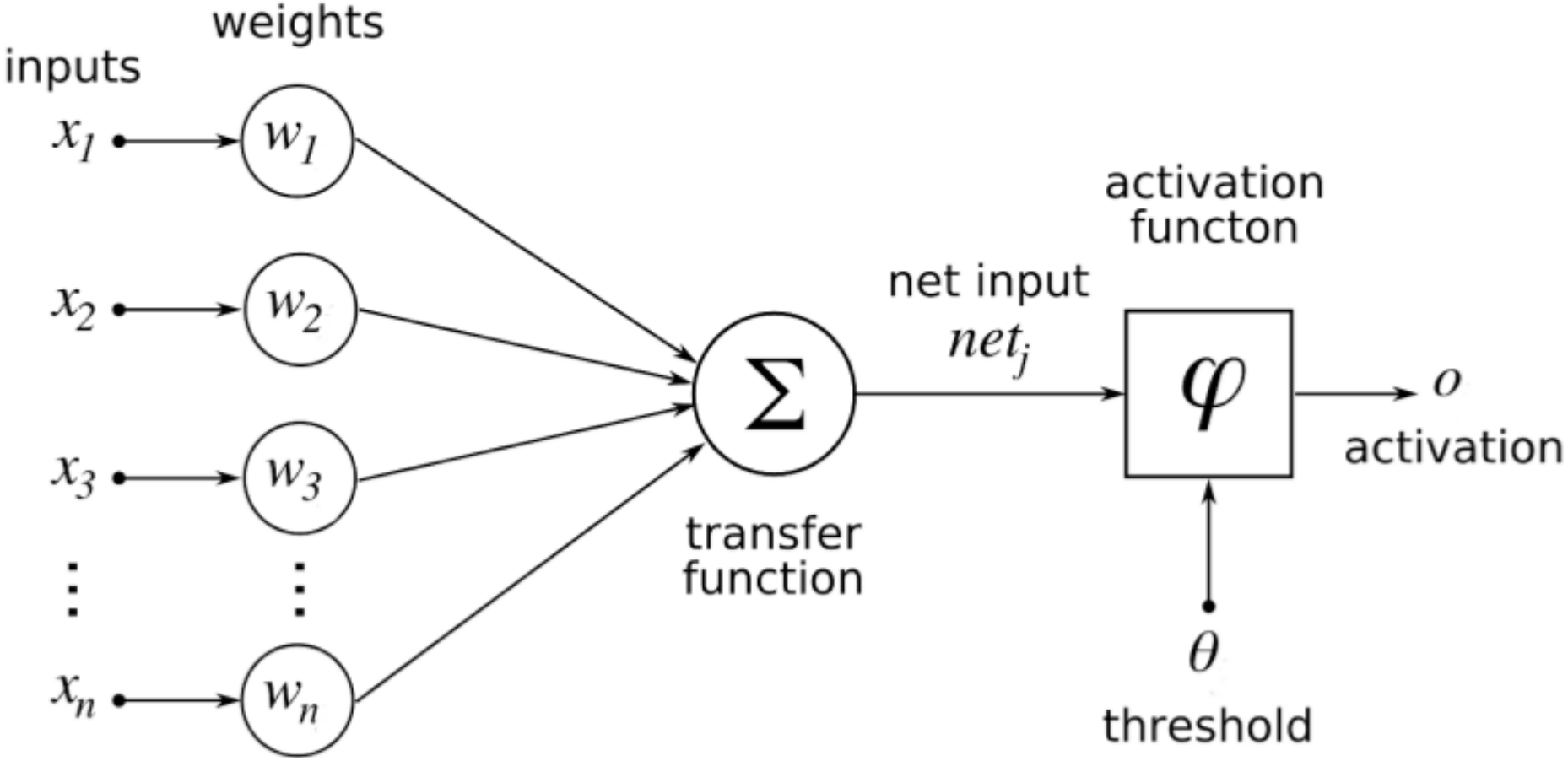
İlk önce kısa bir tarih

Yapay sinir ağları yeni değil.

Rosenblat'ın Perceptron'u, 1958



Perceptron: bir yapay nöron modeli



$$o = \varphi\left(\sum_{i=1}^{D} w_i x_i ; \theta\right) \text{ where } x \in \mathbb{R}^D, o \in \mathbb{R}$$

Yapay sinir ağlarının “gösterim gücü” (İng. *representation power*):

- 1957’de Sovyet matematikçi Kolmogorov, üç katmanlı bir sinir ağının, yeterli sayıda saklı nöron ve uygun non-lineer aktivasyon fonksiyonları verildiğinde herhangi bir sürekli (*İng. continuous*) fonksiyonu gösterebildiğini ispatladı.

Kolmogorov'un ispatı başkaları tarafından geliştirildi. En bilineni Cybenko, 1989.

Approximation by Superpositions of a Sigmoidal Function*

G. Cybenko†

Abstract. In this paper we demonstrate that finite linear combinations of compositions of a fixed, univariate function and a set of affine functionals can uniformly approximate any continuous function of n real variables with support in the unit hypercube; only mild conditions are imposed on the univariate function. Our results settle an open question about representability in the class of single hidden layer neural networks. In particular, we show that arbitrary decision regions can be arbitrarily well approximated by continuous feedforward neural networks with only a single internal, hidden layer and any continuous sigmoidal nonlinearity. The paper discusses approximation properties of other possible types of nonlinearities that might be implemented by artificial neural networks.

Key words. Neural networks, Approximation, Completeness.

1. Introduction

A number of diverse application areas are concerned with the representation of general functions of an n-dimensional real variable, $x \in \mathbb{R}^n$, by finite linear combinations of the form

$$\sum_{j=1}^{N} \alpha_j \sigma(y_j^{\mathrm{T}} x + \theta_j), \tag{1}$$

where $y_j \in \mathbb{R}^n$ and $\alpha_j, \theta \in \mathbb{R}$ are fixed. (y^{T} is the transpose of y so that $y^{\mathrm{T}}x$ is the inner product of y and x.) Here the univariate function σ depends heavily on the context of the application. Our major concern is with so-called sigmoidal σ's:

$$\sigma(t) \to \begin{cases} 1 & \text{as} \quad t \to +\infty, \\ 0 & \text{as} \quad t \to -\infty. \end{cases}$$

Kolmogorov'un ispatı
başkaları t
geliştirildi.
Cybenko, 1

Bu teoremler yapay sinir ağlarının "gösterim gücü"nü açıklıyor, onların nasıl öğrenilebileceğini veya öğrenmenin mümkün olup olmadığını değil.

Approximation by Superpositions of a Sigmoidal Function*

G. Cybenko†

Abstract. In this paper we demonstrate that finite linear combinations of compositions of a fixed, univariate function and a set of affine functionals can uniformly approximate any continuous function of n real variables with support in the unit hypercube; only mild conditions are imposed on the univariate function. Our
bility in the class of single hidden
hat arbitrary decision regions can
feedforward neural networks with
nuous sigmoidal nonlinearity. The
er possible types of nonlinearities
etworks.

Completeness.

on

ncerned with the representation of
ble, $x \in \mathbb{R}^n$, by finite linear combina-

$\theta_j)$, (1)

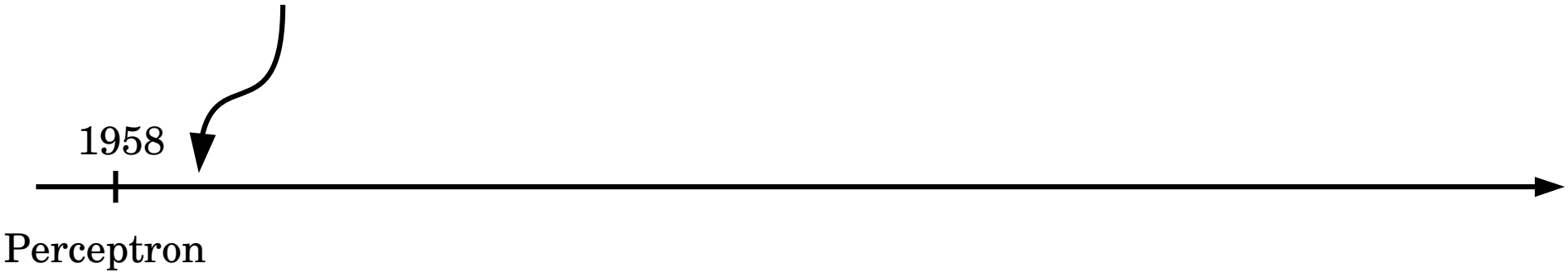
ranspose of y so that $y^T x$ is the inner
on σ depends heavily on the context
of the application. Our major concern is with so-called sigmoidal σ's:

$$\sigma(t) \to \begin{cases} 1 & \text{as} \quad t \to +\infty, \\ 0 & \text{as} \quad t \to -\infty. \end{cases}$$

Öğrenilebilirlik cephesinde...

Perceptron ve Adaline gibi algoritmalar 1960'larda geliştirildi

Tek katmanlı perceptron'un birçok Bool fonksiyonunu öğrenebildiği gösterildi.



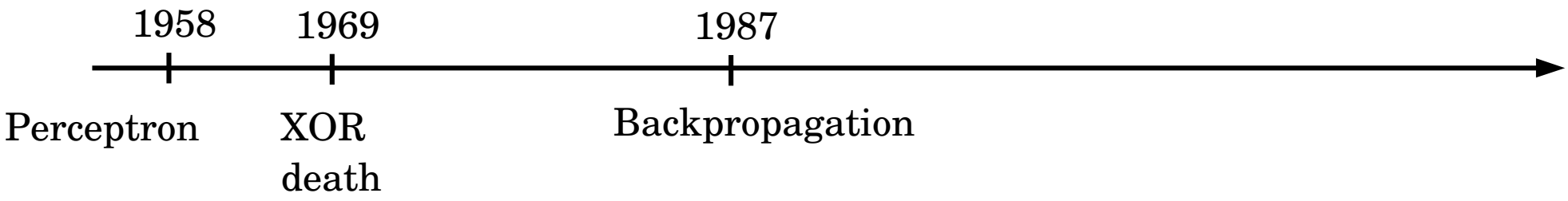
Öğrenilebilirlik cephesinde...

1969'da Marvin Minsky tek katmanlı perceptron'un XOR fonksiyonunu öğrenemeyeceğini gösterdi.

- (Yapay sinir ağlarının ilk "gömülüşü".)

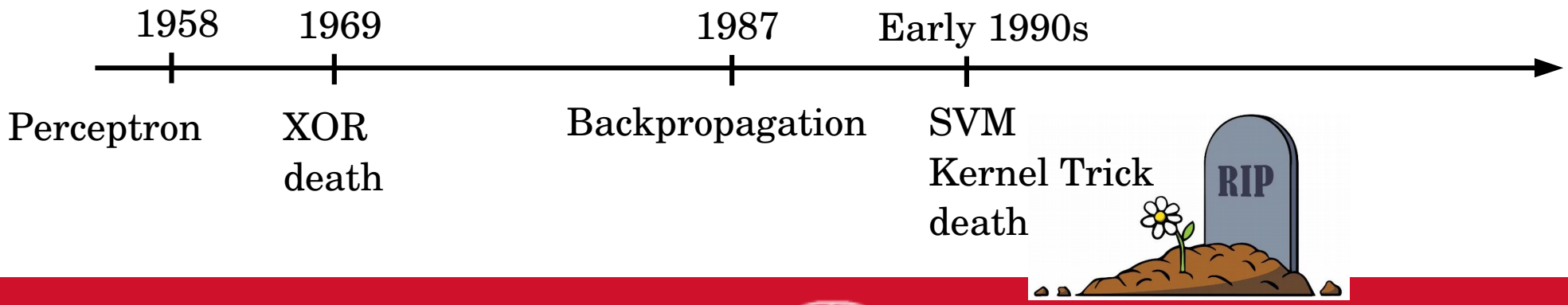
Geriyayılım (Back-propagation)

1980'lerin sonunda "geriyayılım" yönteminin bulunması ve iyi sonuçlar verdiğinin gösterilmesiyle birlikte yapay sinir ağları yeniden popüler oldu.



Geriyayılım (Back-propagation)

90'ların başında "Support Vector Machines" ve "Kernel Trick" ortaya çıkana kadar. (ikinci gömülüş)



Yapay sinir ağları 90'larda neden terk edildi?

- SVM'ler daha iyi sonuç veriyordu.
- Yapay sinir ağlarında katman sayısını arttırmak sonucu iyileştirmiyordu.
- Bazı modellerde (recurrent networks) geriyayılım hiç iyi sonuç vermedi.

Bugünden geriye baktığımızda

90'lardaki başarısızlığın kaynakları:

- Verisetlerinin aşırı küçük olması
- Bilgisayarlarımızın çok güçsüz olması
- Yanlış bir şekilde ilkleme (*İng. initialization)*
- Yanlış non-lineer aktivasyon fonksiyonları

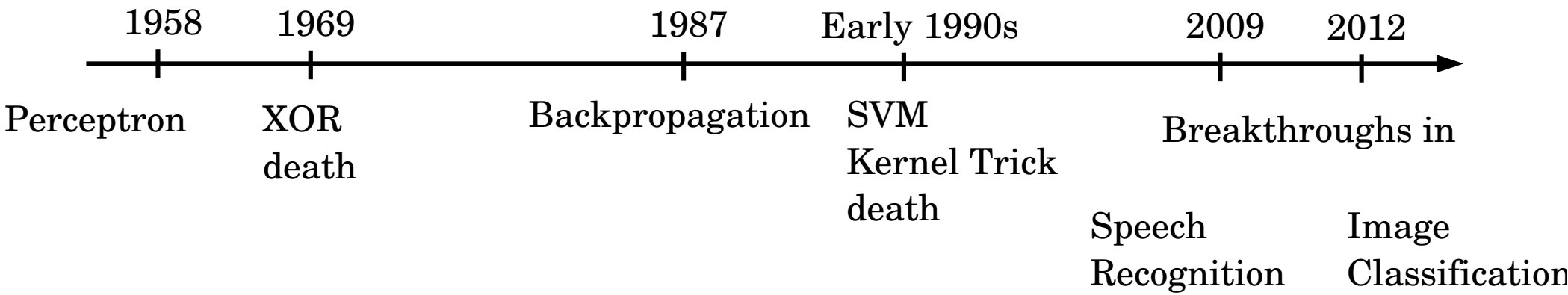
Kaynak: G. Hinton'nın Royal Society'de verdiği konuşma, 22 Mayıs 2016, https://youtu.be/izrG86jycck

Derin öğrenme modellerinin klasik yapay sinir ağlarından ne farkı var?

- "Temelde hiçbir farkları yok" demek yanlış olmaz.
- Yeni olan şeyler:
  - Daha çok veri ve daha çok işlem gücü
  - Yeni doğrusaldışı (*İng. non-lineer*) aktivasyon fonksiyonları
  - Yeni ilkleme (*İng. initialization*) yöntemleri
  - Yeni düzenlileştirme (*İng. regularization*) yöntemleri

Peki bugün?

- Yapay sinir ağları 3. baharını yaşıyor.
- Bu yeniden diriliş, 2009 ve 2012'de yapılan iki çalışmaya dayanıyor.



2009'da G. Hinton ve öğrencileri konuşma tanıma problemi (*İng. speech recognition*) için yeni bir eğitme yöntemi geliştirdi.

Eğitmensiz (İng. unsupervised) öğrenme ile ağı ilklediler.

En sona "eğitmenli" katmanı ekleyip geriyayılım kullandılar.

Bu yöntemle uzun süredir en iyi sonucu veren modeli geçtiler.

Yöntemleri Android telefonlarda 2012'den itibaren kullanılmaya başlandı.

İlgili çalışma: Mohamed, A. R., Dahl, G. E. and Hinton, G. E. "Deep belief networks for phone recognition." NIPS workshop on deep learning for speech recognition.

Yeni düzenlileştirme yöntemi: “Dropout”

Ezberlemeyi (*İng. overfitting*) önleyen yeni yöntem

2009’daki ve sonraki sistemlerin başarılı olmasında önemli rol oynadı.

İkinci başarı hikayesi (2012)

- Bilgisayar görüsü (*İng. computer vision*)
- ILSVRC 2012 yarışması
  - 1,2 milyon görüntü, 1000 sınıf
  - Problem: verilen görüntü için, o görüntüdeki baskın nesneyi tahmin etmeye çalışın. 5 tahmin üretin. Bu 5 tahminden biri doğruysa, başarılı sayılıyor.

İkinci başarı hikayesi (2012)

G. Hinton ve öğrencisi Alex Krizhevsky, 2009'daki yöntemi kullanarak 7 katmanlı bir evrişimsel sinir ağı eğitti (*İng. convolutional neural network*)

Bu ağ, günümüzde "AlexNet" olarak biliniyor.

İlgili çalışma: Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

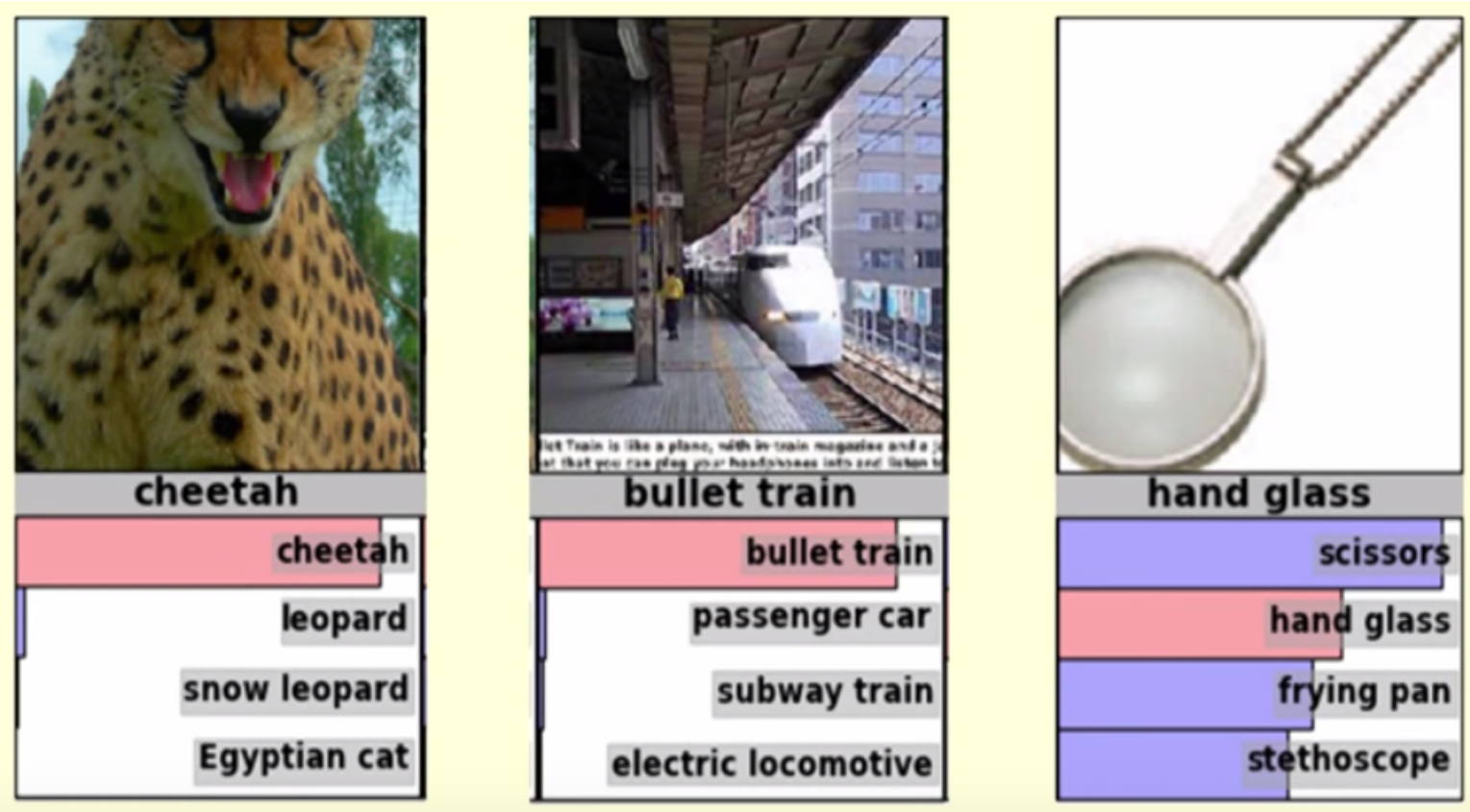
İkinci başarı hikayesi (2012)

AlexNet, %16 hata oranı elde etti.

En iyi ikinci yöntem, 2012'deki en iyi tüm bilgisayar görüsü algoritmalarının (SIFT, LBP, GIST, Fisher vector, vd.) bir kombinasyonuydu.

- Hata oranı %26.

İkinci başarı hikayesi (2012)



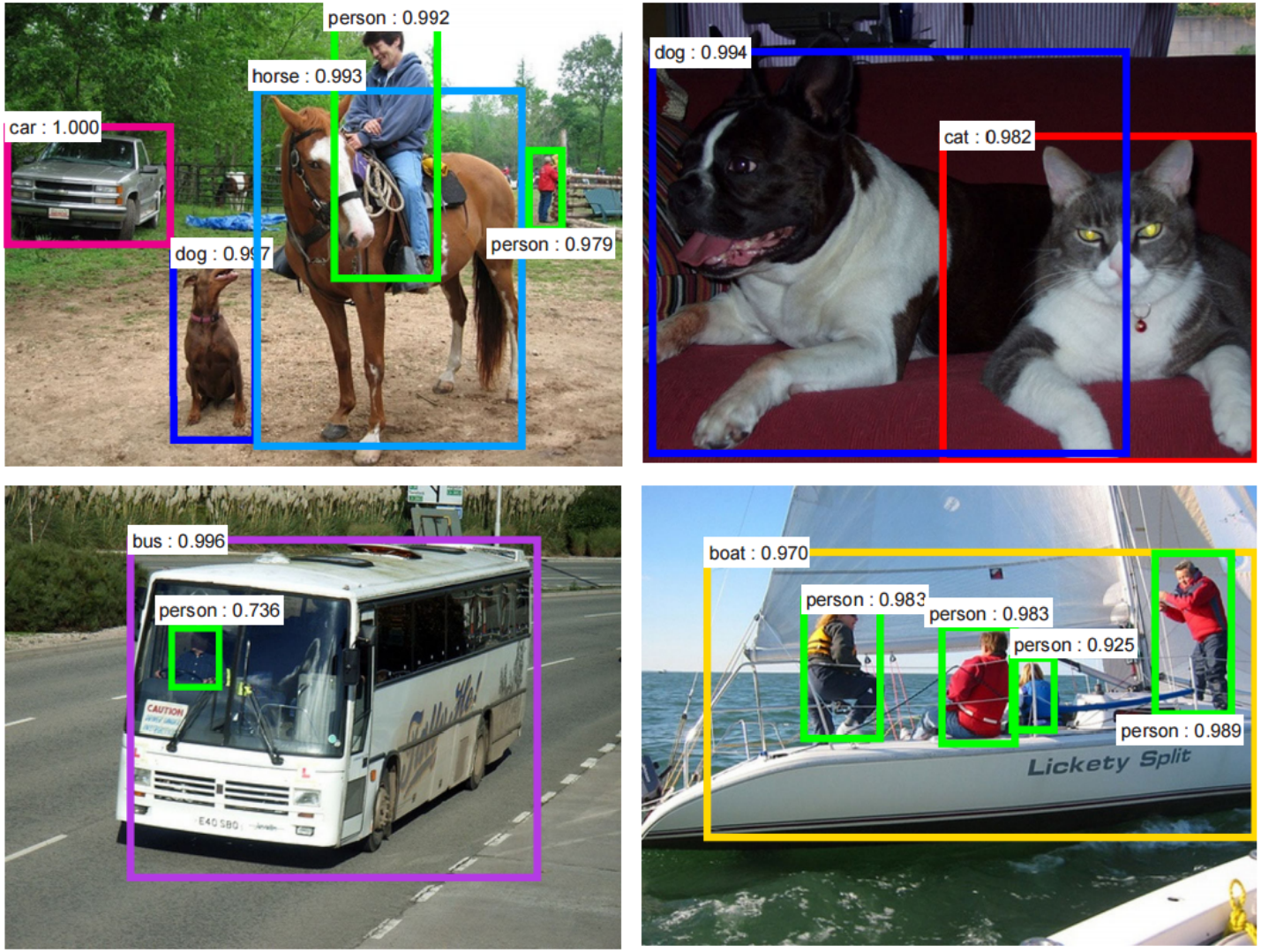
Kaynak: G. Hinton'nın Royal Society'de verdiği konuşma, 22 Mayıs 2016, https://youtu.be/izrG86jycck

2012'den bugüne

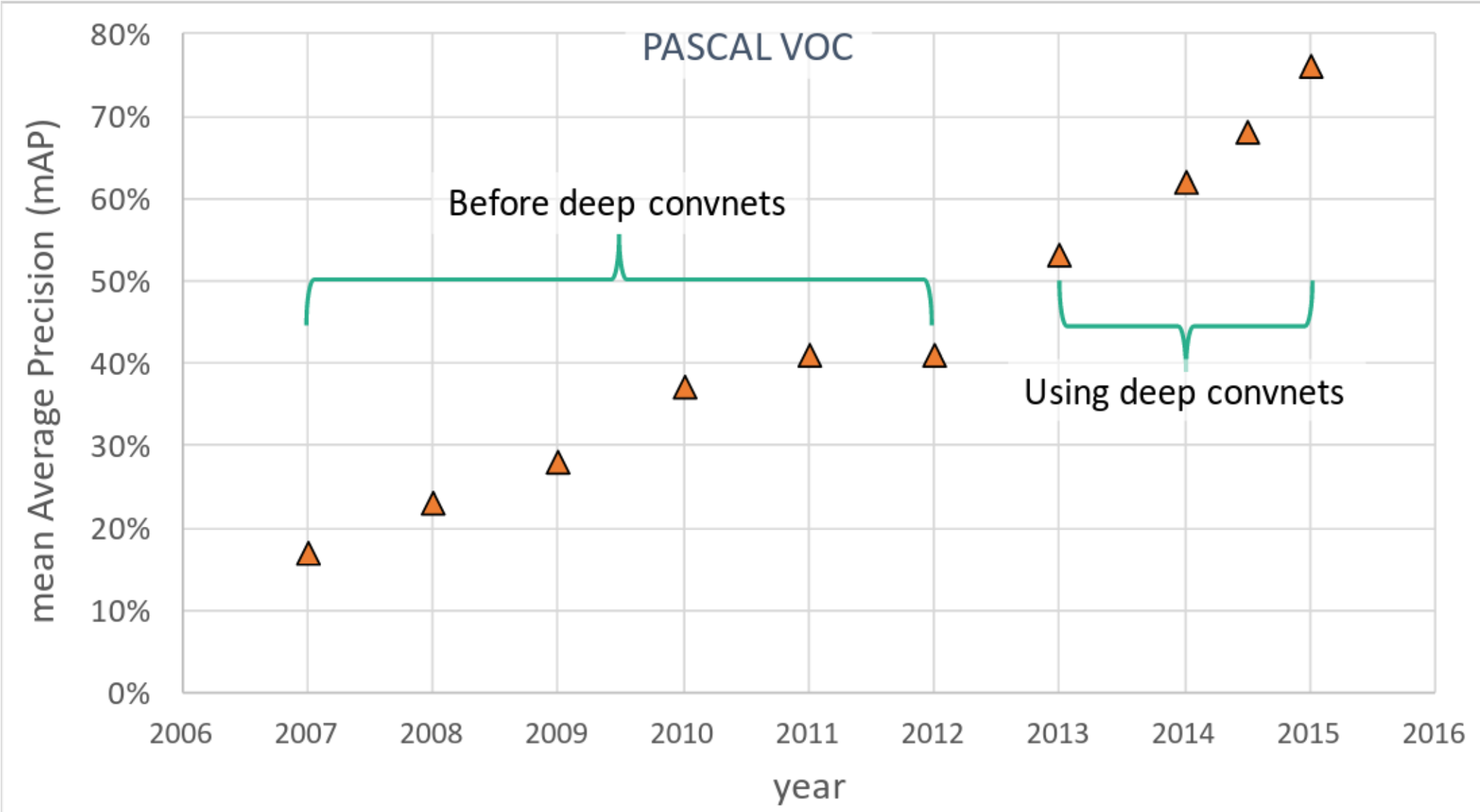
Aynı yarışmada derin öğrenme yöntemleri bugün hata oranını %5'in altına indirdi (Bu verisetinde insanın hata oranı ~%5)

… ve daha birçok başarı hikayesi

Nesne algılama (*İng. object detection*)



(Figure from Ren, He, Girschik, Sun, NIPS 2015)



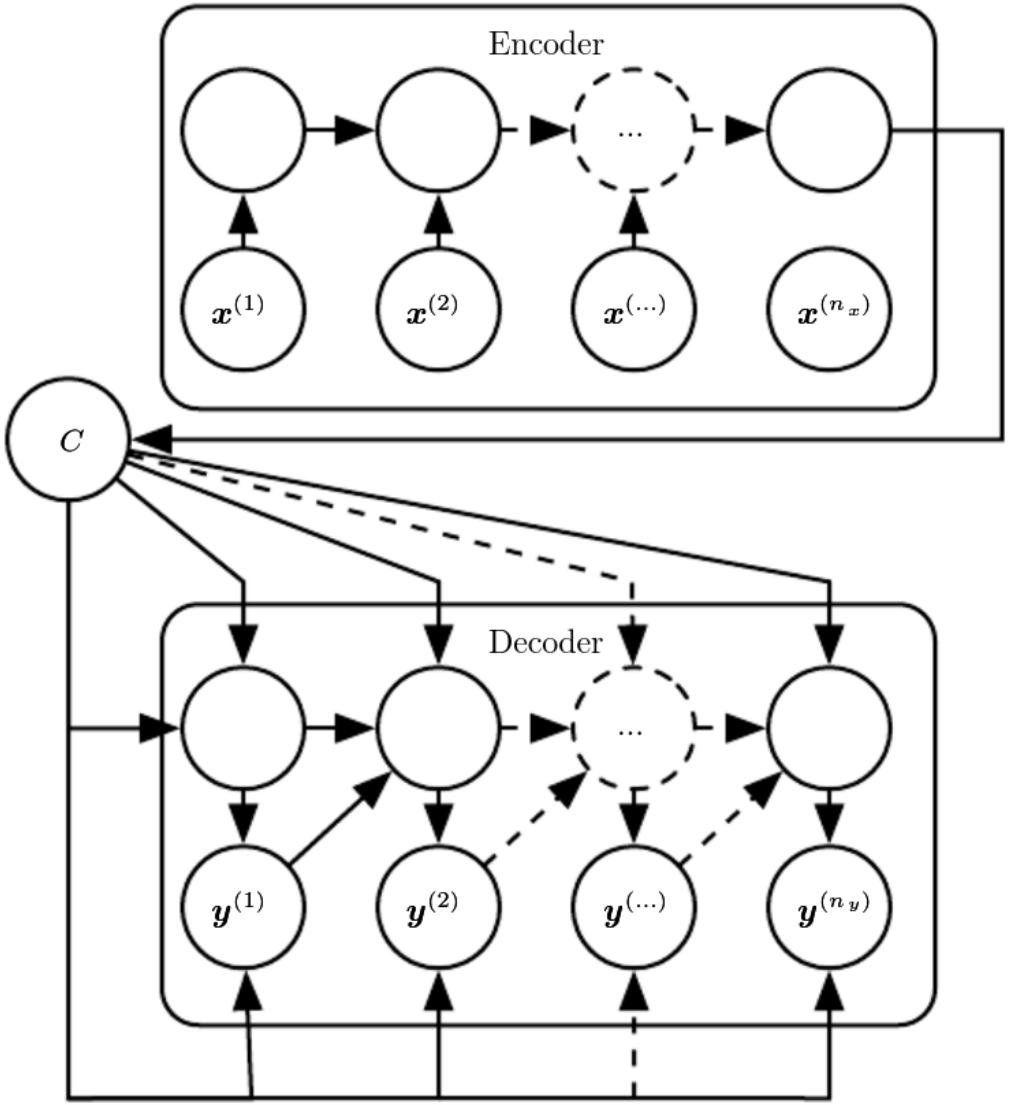
Kaynak: "Faster RCNN slides" by R. Girschik.

Otomatik tercüme

Google translate

Yinelgeli sinir ağları (*İng. Recurrent neural networks*)

[Sutskever, Vinyals and Le 2014]



Görüntü altyazılama (*İng. image captioning*)

Demo

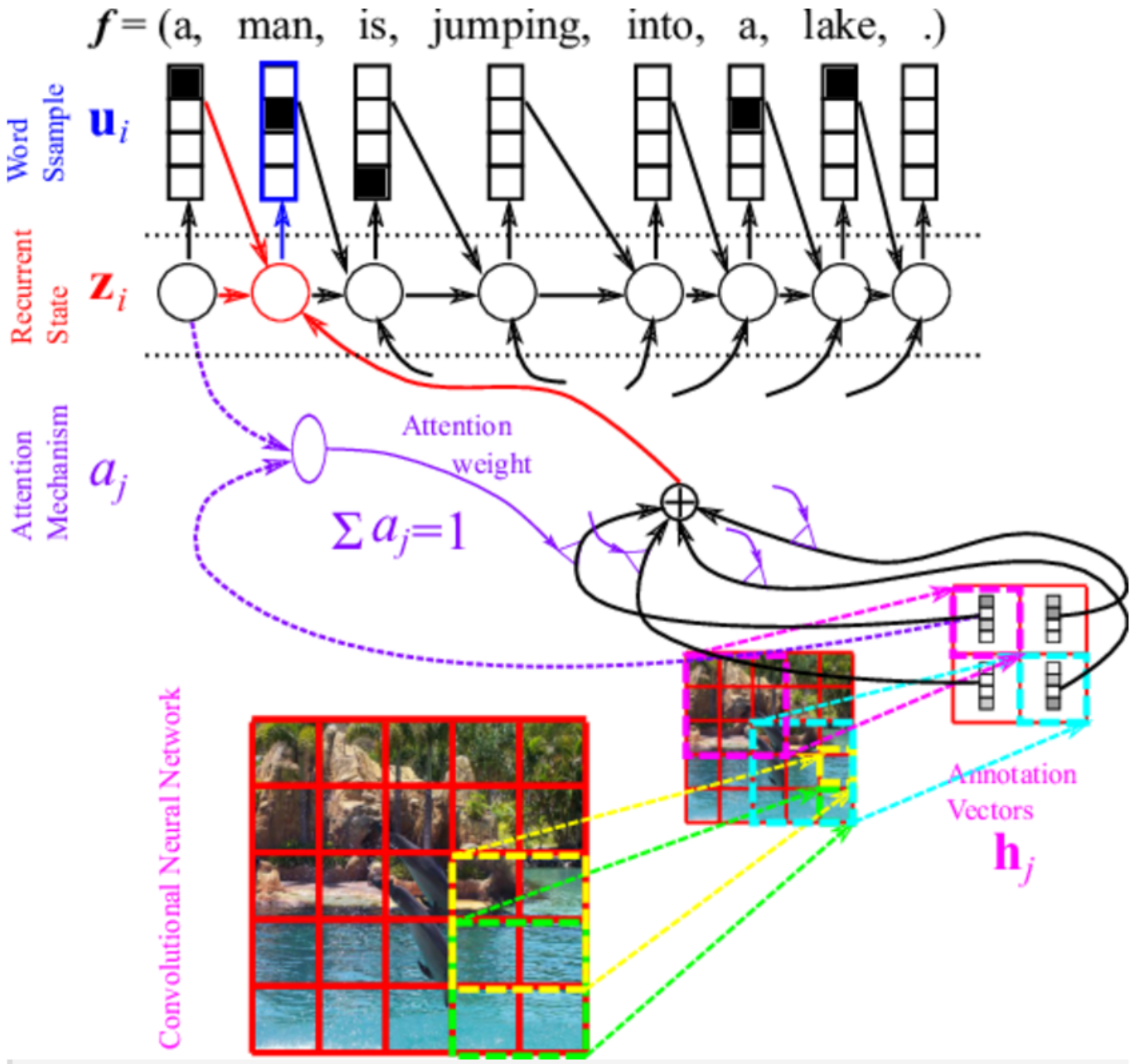
a street sign on a pole in front of a building

a plate with a sandwich and a salad

an elephant standing in a grassy field with trees in the background

a man is throwing a frisbee in a park

Görüntü altyazılama (*İng. image captioning*)



[Donahue et al. 2015]

Derin üretici (*İng. generative*) modeller

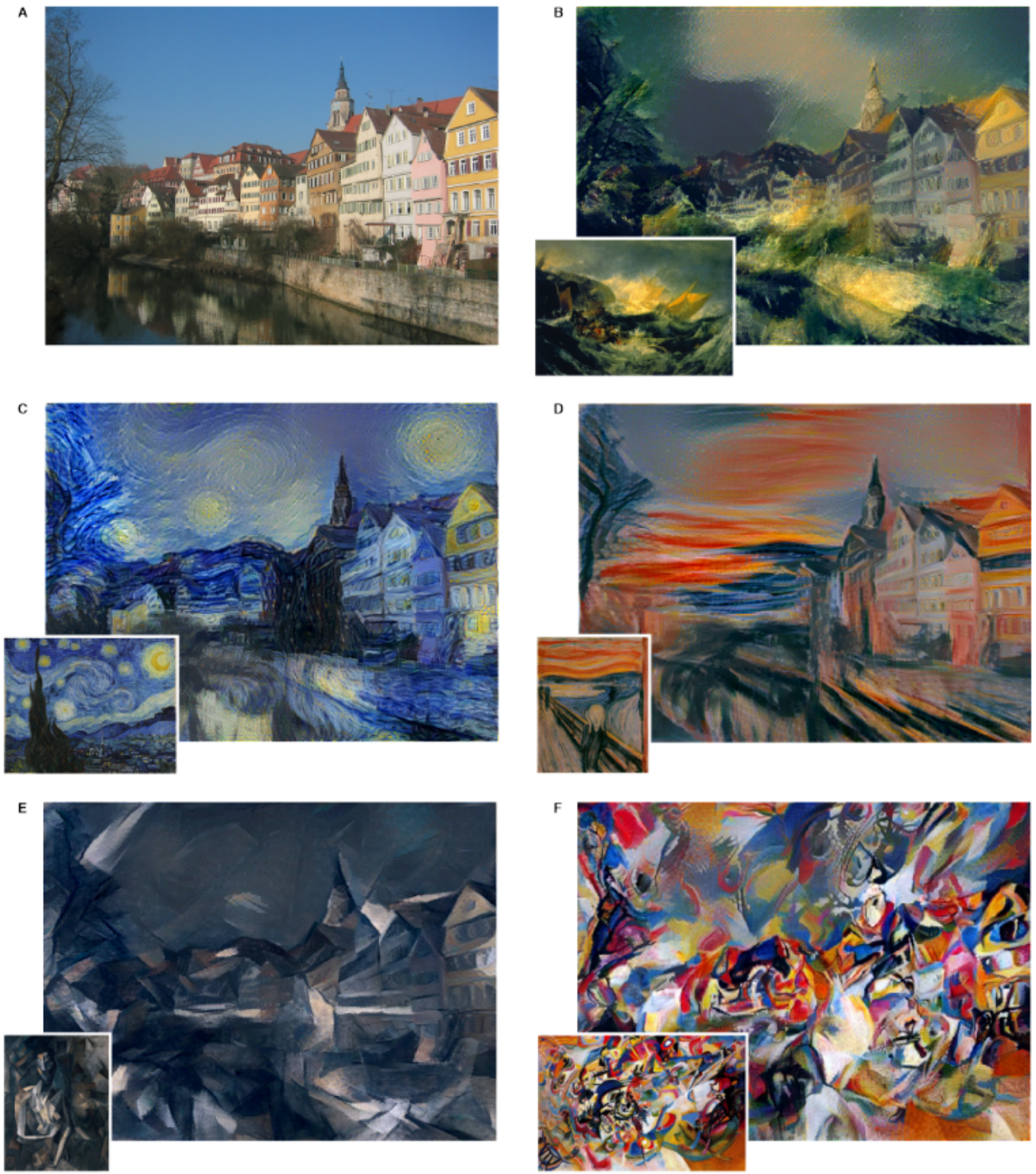
Figure 1. Image completions sampled from a PixelRNN.

(From Oord, Kalchbrenner, Kavukcuoglu 2016)

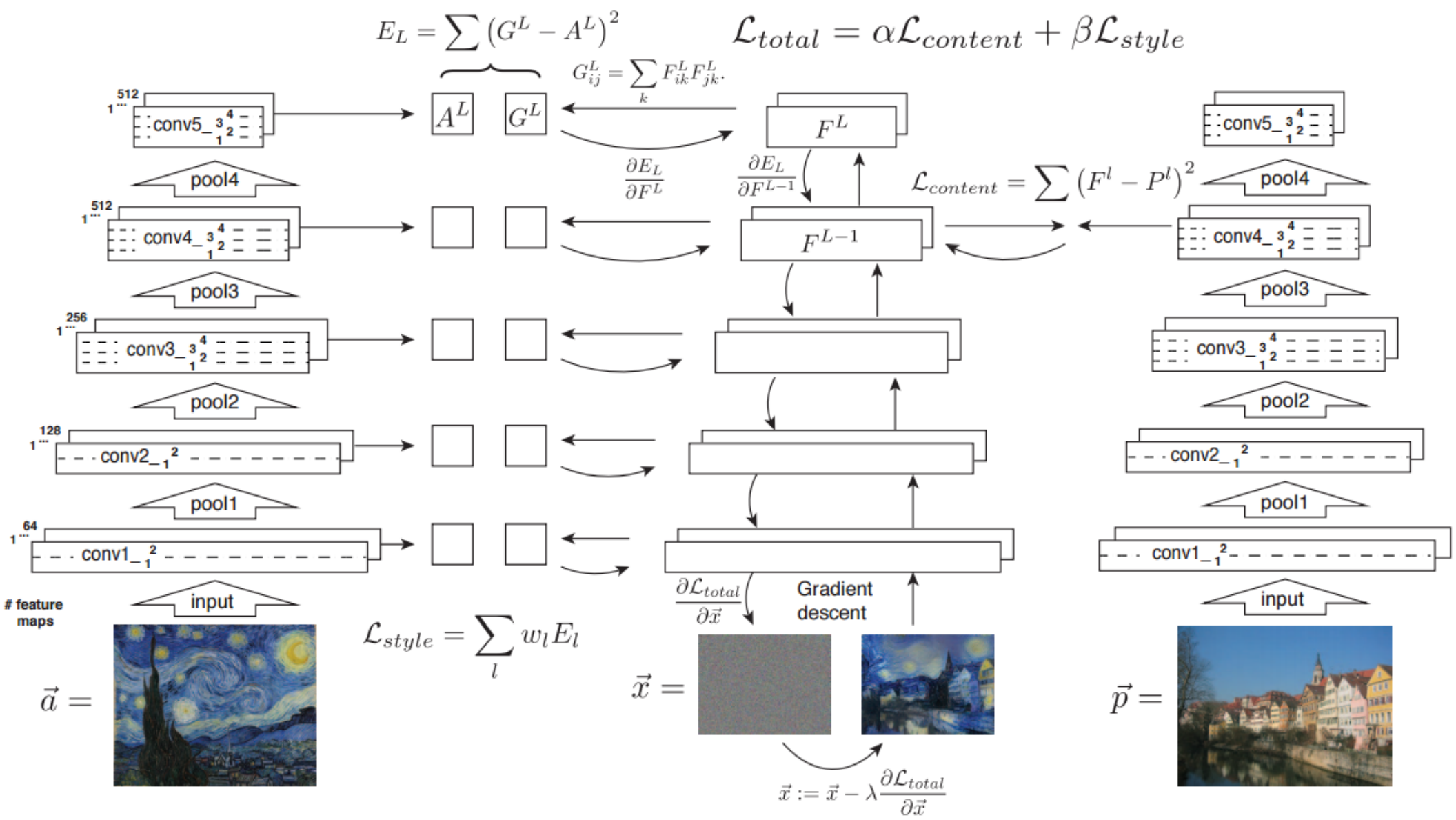
Derin üretici (*İng. generative*) modeller

Berthelot, David, Tom Schumm, and Luke Metz. "Began: Boundary equilibrium generative adversarial networks." arXiv preprint arXiv:1703.10717 (2017).

Sanatsal stil transferi

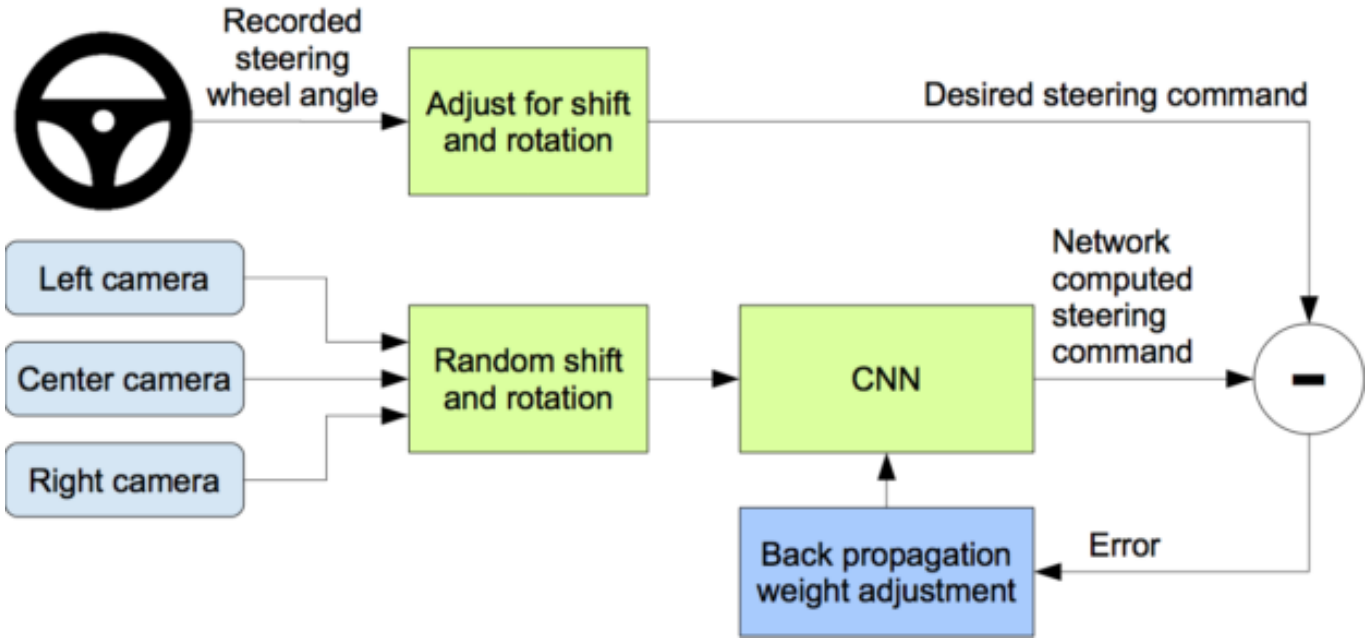


[Gatys, Ecker, Bethge 2015]


$E_L = \sum (G^L - A^L)^2$
$\mathcal{L}_{total} = \alpha \mathcal{L}_{content} + \beta \mathcal{L}_{style}$
$G^L_{ij} = \sum_k F^L_{ik} F^L_{jk}$
$\mathcal{L}_{content} = \sum (F^l - P^l)^2$
$\mathcal{L}_{style} = \sum_l w_l E_l$
$\vec{x} := \vec{x} - \lambda \frac{\partial \mathcal{L}_{total}}{\partial \vec{x}}$
conv5_1 2
conv4_1 2
conv3_1 2
conv2_1 2
conv1_1 2
pool4
pool3
pool2
pool1
input
512
256
128
64
feature maps
Gradient descent
$\vec{a} =$
$\vec{x} =$
$\vec{p} =$

https://deepdreamgenerator.com/ Kullanılarak üretildi.

Sürücüsüz arabalar



(Figürlerin kaynağı: https://devblogs.nvidia.com/parallelforall/deep-learning-self-driving-cars/)

Neden her yerde “derin öğrenme” görüyoruz?

Birçok farklı problemde konvansiyonel makine öğrenmesi yöntemlerinden (çok) daha yüksek doğruluk verdiği için,

Bu doğruluk seviyesinin ticari uygulamaları olanaklı kılmasından dolayı

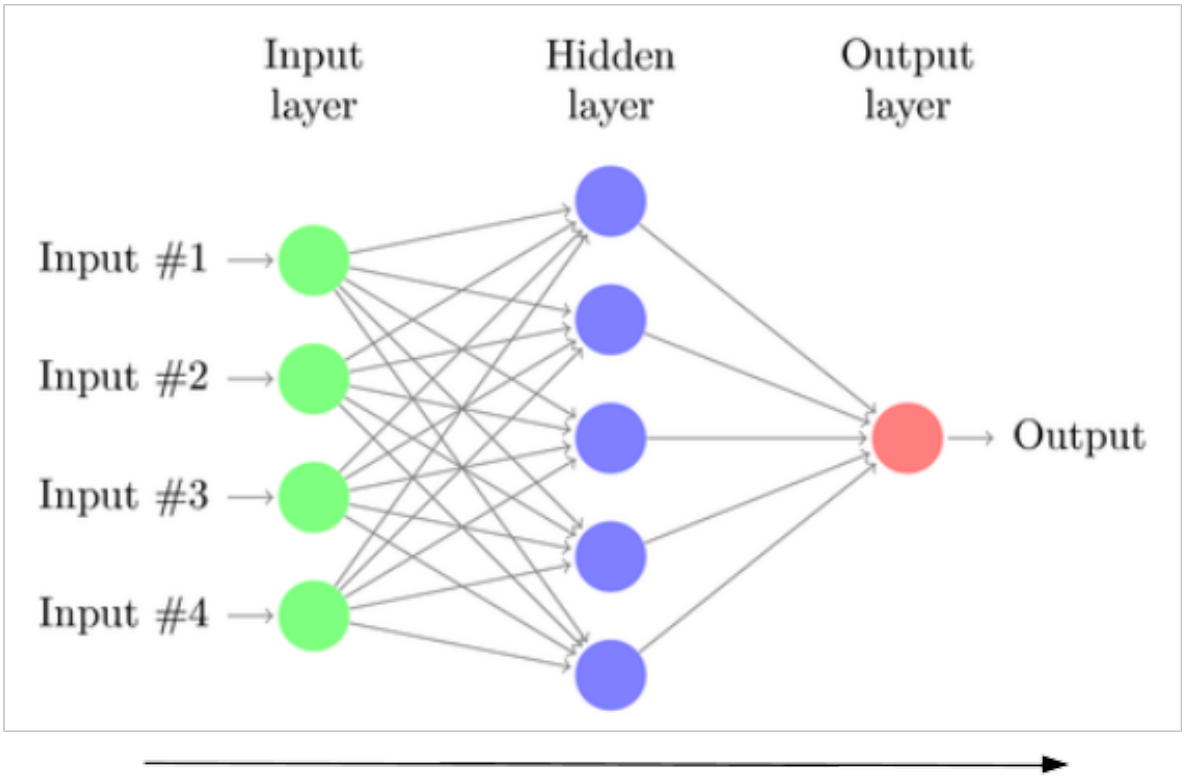
Yeni uygulamalara olanak sağladığı için.

Nasıl çalışıyor?

Üç ana derin model çeşidi:

- **Çok katmanlı Perceptron** (Multilayer Perceptrons)
- **Evrişimsel Sinir Ağı** (Convolutional Neural Networks)
- **Yinelgeli Sinir Ağı** (Recurrent Neural Networks)

Çok katmanlı perceptron

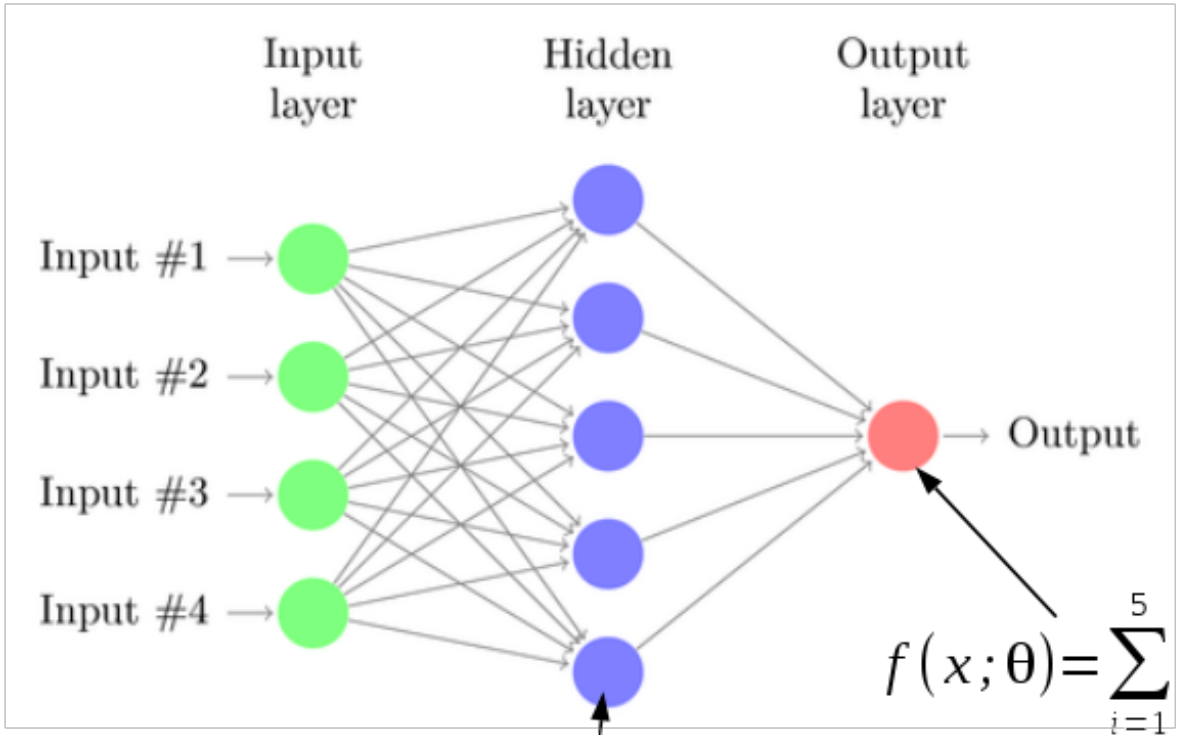


Fully connected, feedforward network

Input is a 4-dim vector.

There are 5 hidden nodes each with a 4-dim weight vector.

Çok katmanlı perceptron



$$f(x;\theta)=\sum_{i=1}^{5}\theta_i\varphi(w_i^T x)$$

Input $x\in\mathbb{R}^4$

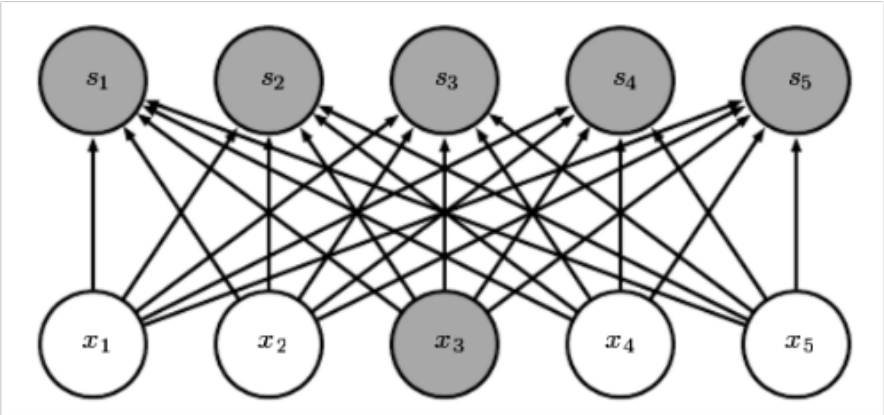
$$g(x;w_5)=\varphi(w_5^T x)$$

Activation function

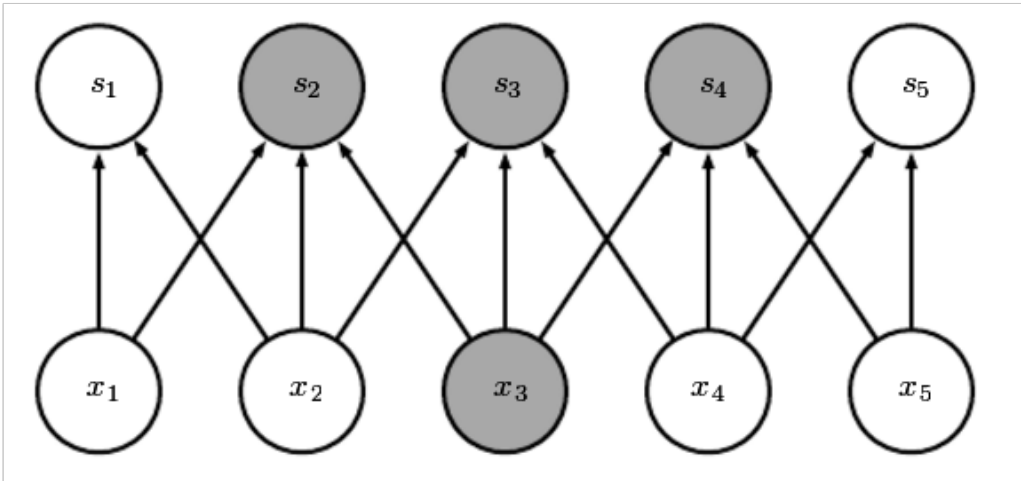
Evrişimsel Sinir Ağları (Convolutional Neural Networks (CNNs))

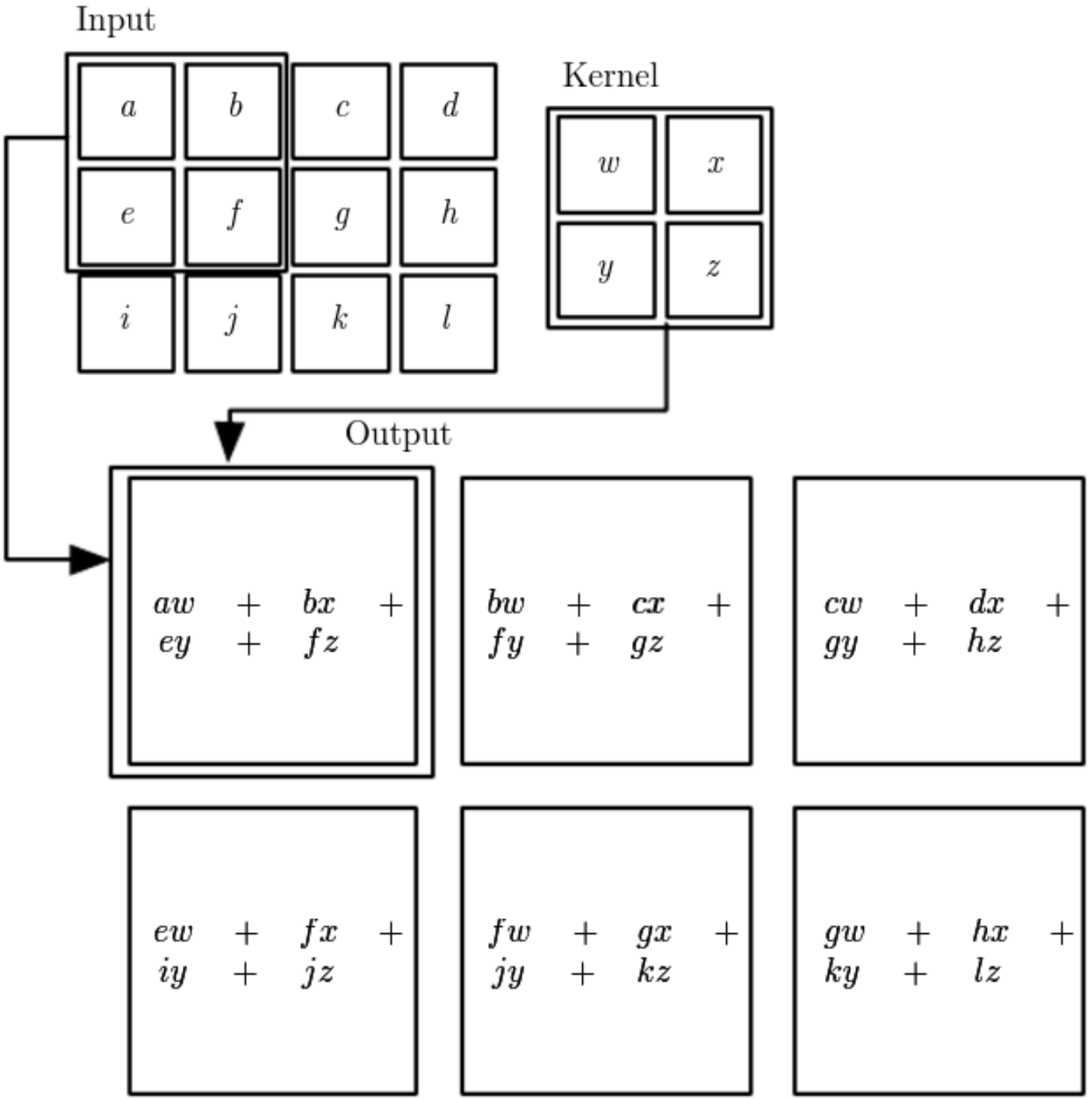
Sparse interactions and parameter sharing

In a usual ANN, nodes are fully-connected

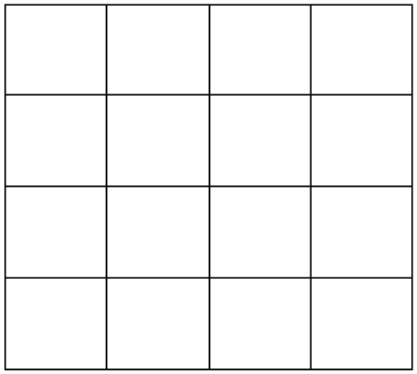


In CNN, sparse connections:

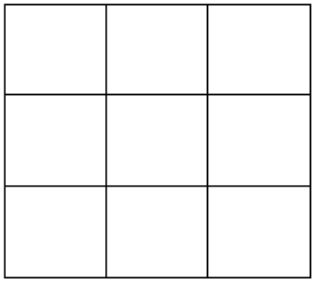



Input
a
b
c
d
e
f
g
h
i
j
k
l
Kernel
w
x
y
z
Output
aw + bx + ey + fz
bw + cx + fy + gz
cw + dx + gy + hz
ew + fx + iy + jz
fw + gx + jy + kz
gw + hx + ky + lz

Sparse interactions



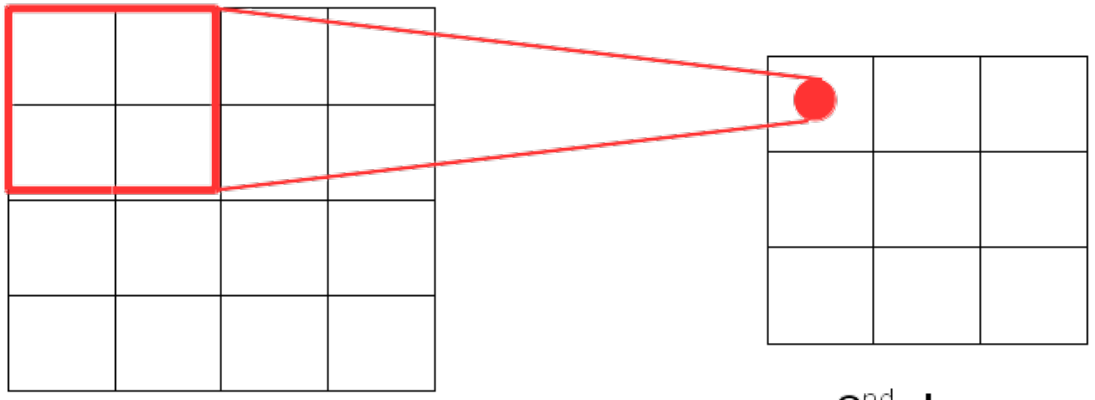
1st (input) layer: 4x4 image



2nd layer

2x2 filter

Sparse interactions

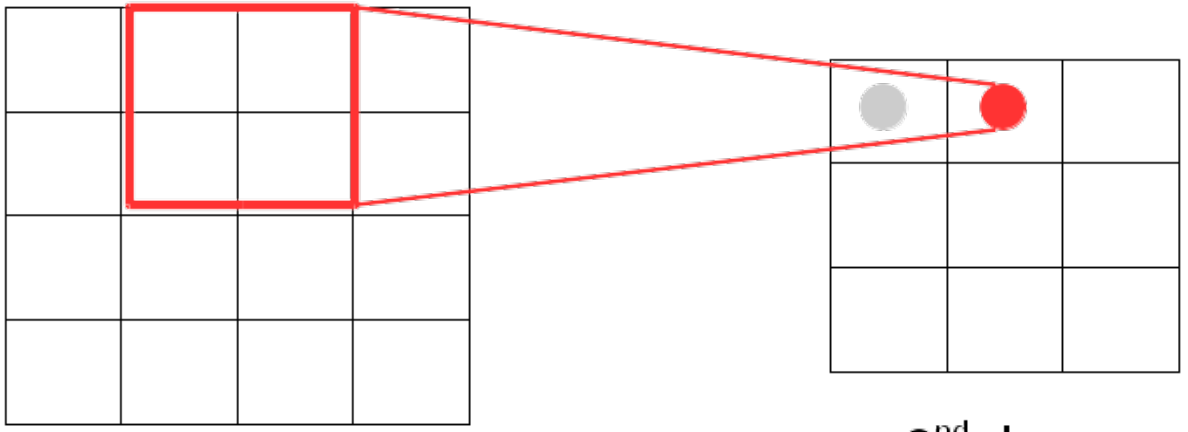


2^{nd} layer

1^{st} (input) layer: 4x4 image

Node in the 2^{nd} layer is not fully-connected to the nodes in the 1^{st} layer.

Sparse interactions

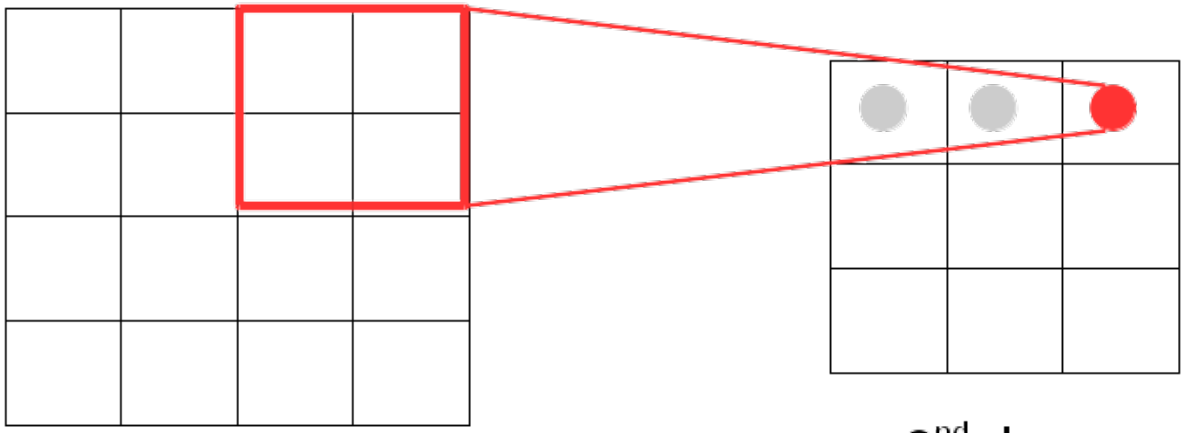


2^{nd} layer

1^{st} (input) layer: 4x4 image

: computed

Sparse interactions

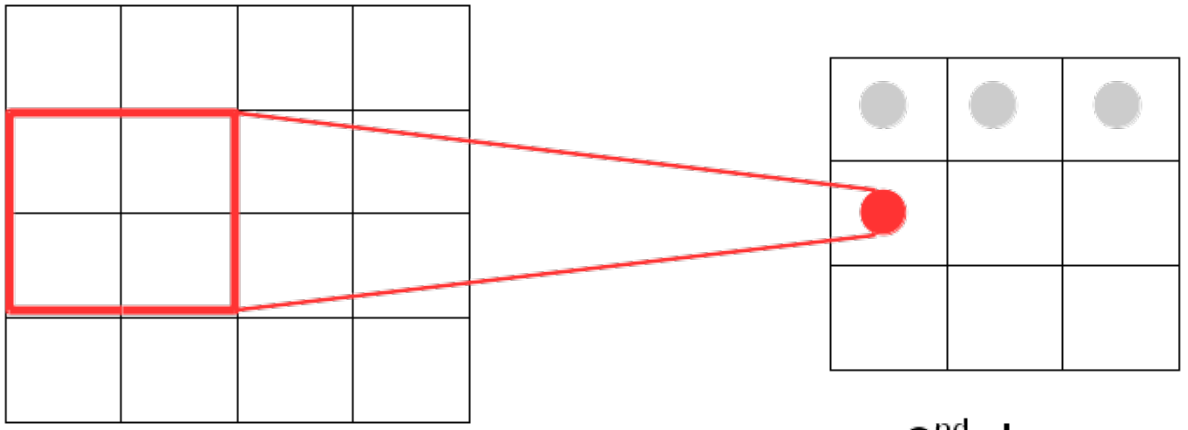


1st (input) layer: 4x4 image

2nd layer

: computed

Sparse interactions

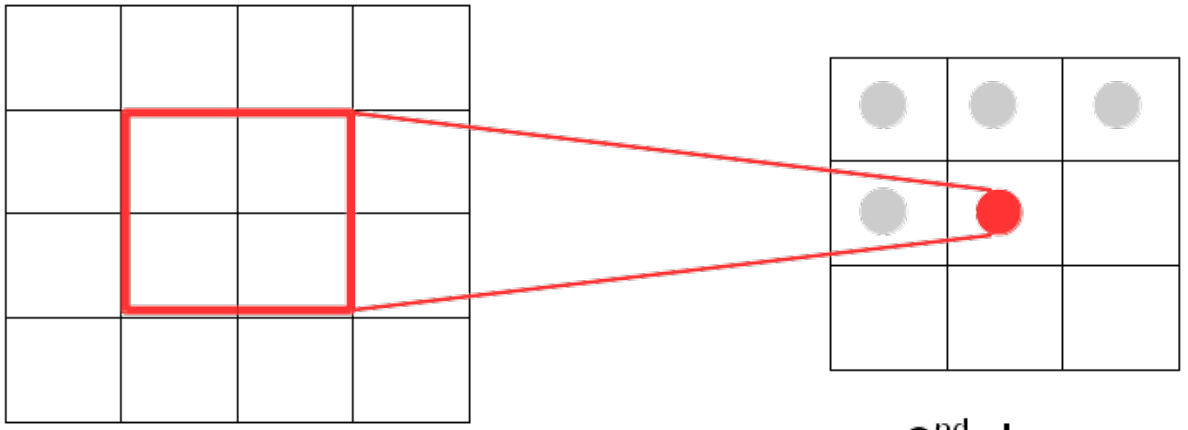


2^{nd} layer

1^{st} (input) layer: 4x4 image

: computed

Sparse interactions



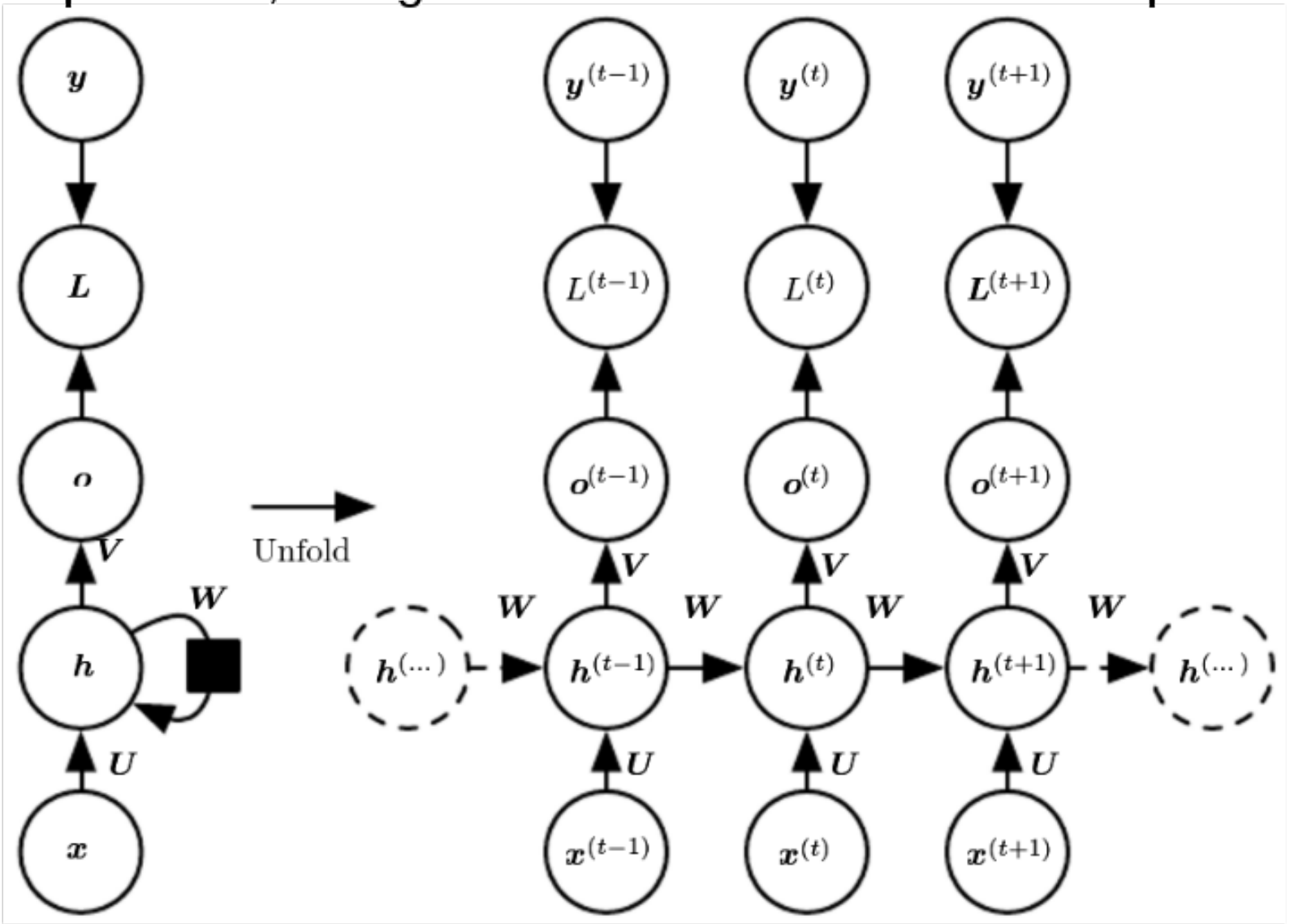
2^{nd} layer

1^{st} (input) layer: 4x4 image

: computed

Yinelgeli Sinir Ağları (Recurrent Neural Networks (RNNs))

A recurrent network that maps input sequence ***x*** to output sequence ***o***, using a loss function *L* and label sequence ***y***.



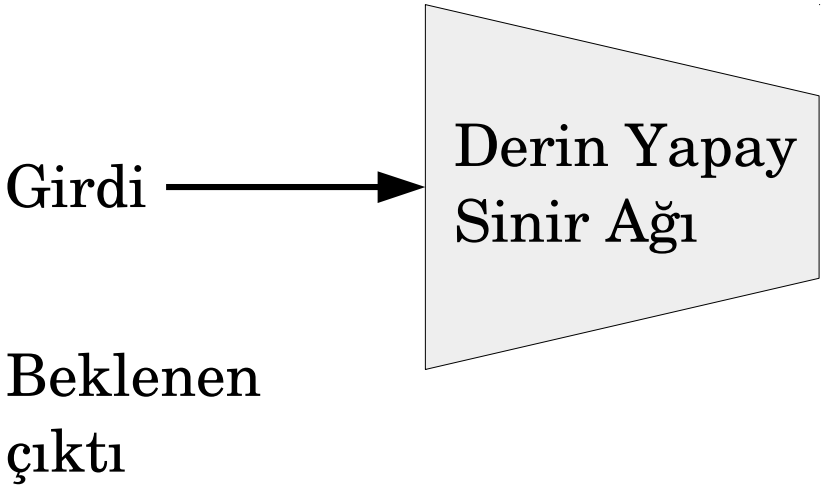
[Fig. 10.3 from Goodfellow et al. (2016)]

Ağın eğitimi nasıl gerçekleşiyor?

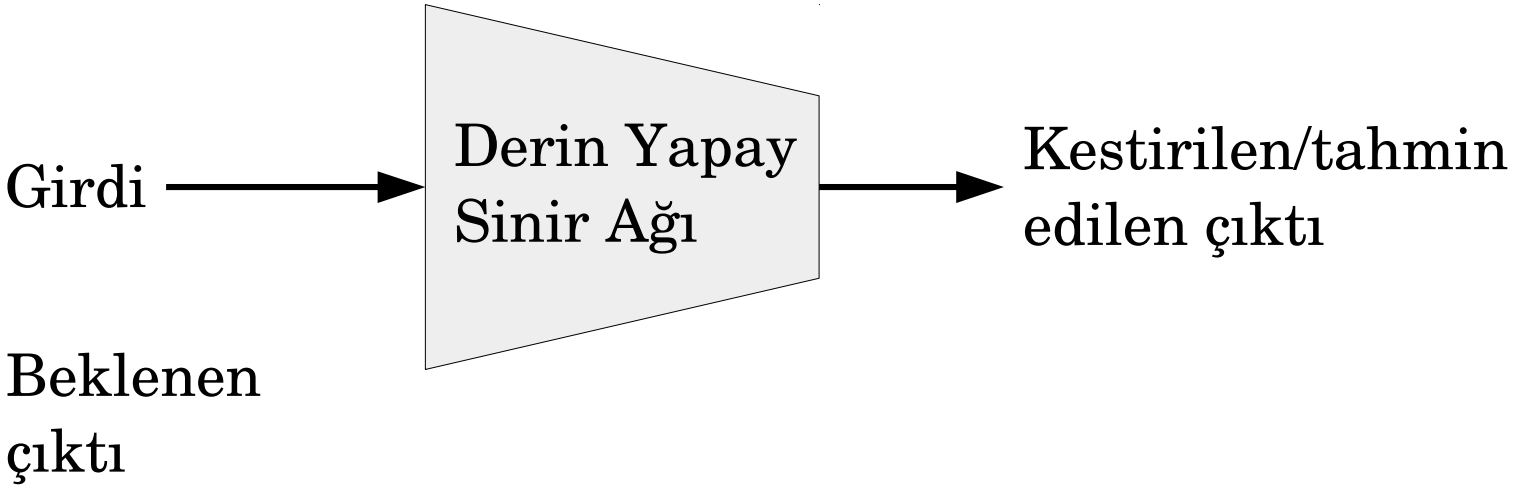
Girdi

Beklenen
çıktı

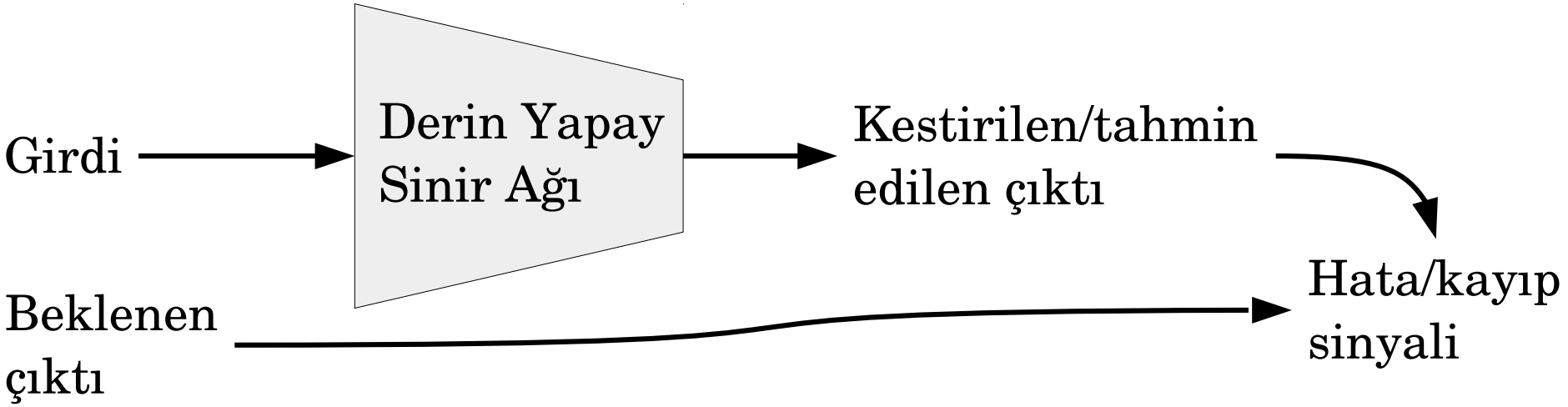
Ağın eğitimi nasıl gerçekleşiyor?



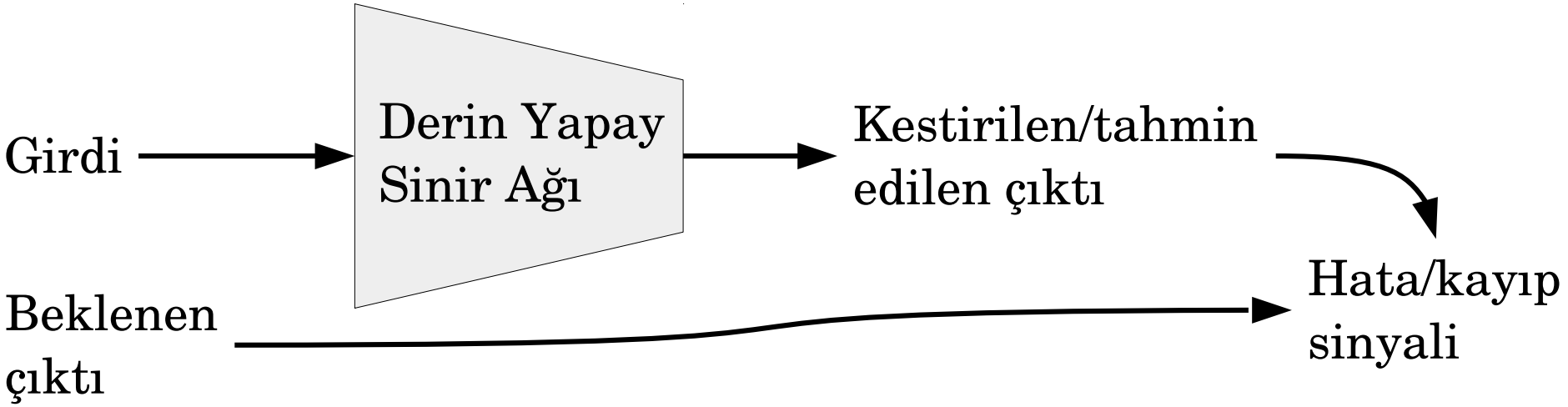
Ağın eğitimi nasıl gerçekleşiyor?



Ağın eğitimi nasıl gerçekleşiyor?

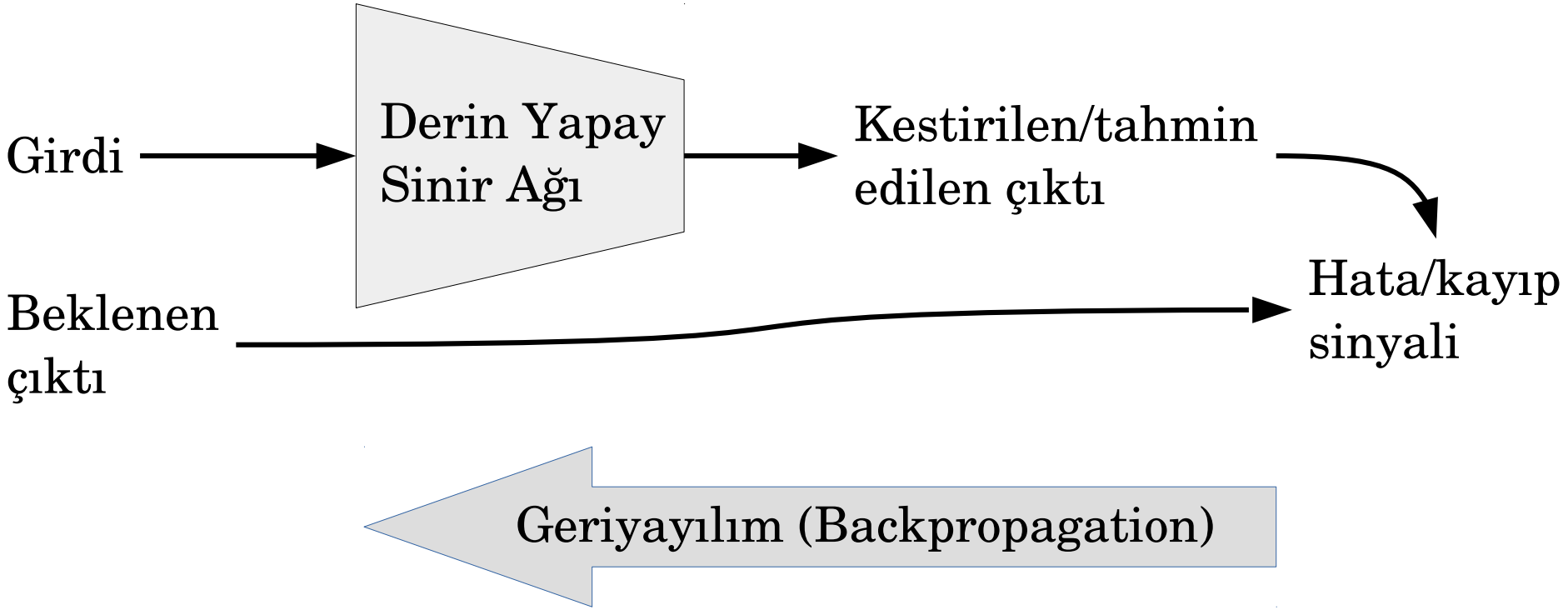


Ağın eğitimi nasıl gerçekleşiyor?



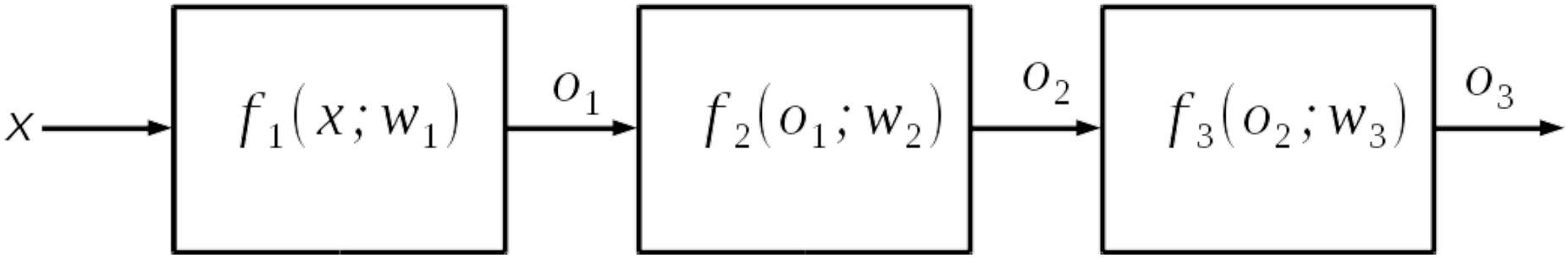
Hata sinyalinin türevini al ve Derin Yapay Sinir Ağı'ndaki ağırlıkları türevin negatif yönünde güncelle.

Ağın eğitimi nasıl gerçekleşiyor?



Hata sinyalinin türevini al ve Derin Yapay Sinir Ağı'ndaki ağırlıkları türevin negatif yönünde güncelle.

Geriyayılım (Backpropagation)



$$\frac{\partial o_3}{\partial w_1} = \frac{\partial o_3}{\partial o_2} \frac{\partial o_2}{\partial o_1} \frac{\partial o_1}{\partial w_1}$$

Türevde zincir kuralı (chain rule)

Nasıl ve nereden başlamalı?

Başlamak için en uygun kütüphaneler:

PyTorch https://pytorch.org/

Keras https://keras.io/

En iyi online ders: http://cs231n.stanford.edu/

Bölümümüzde de ders açılıyor: CENG 783 ve 793

Örnek Keras kodu

```
from keras.layers import Dense, Activation

model.add(Dense(units=64, input_dim=100))
model.add(Activation('relu'))
model.add(Dense(units=10))
model.add(Activation('softmax'))
```

```
# x_train and y_train are Numpy arrays --just like in the Scikit-Learn API.
model.fit(x_train, y_train, epochs=5, batch_size=32)
```

Teşekkürler!

İletişim:
emre@ceng.metu.edu.tr
http://user.ceng.metu.edu.tr/~emre/