Trendwise Analytics

Introduction to AI And Deep Learning

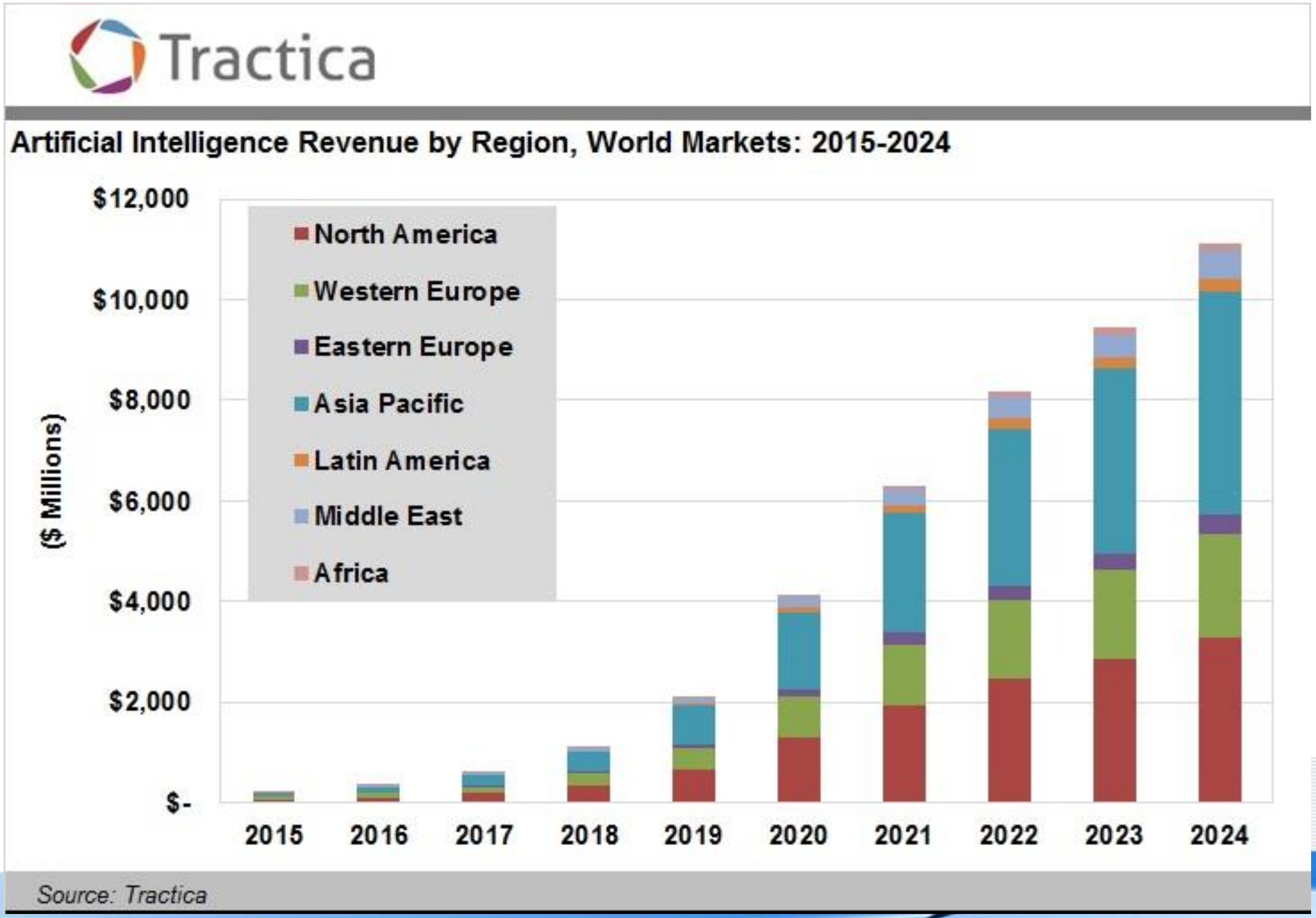
Mohan Kumar

Market potential

- The AI market is expected to be worth USD 16.06 Billion
- by 2022 growing at a CAGR of 62.9%.

-IBM CEO claims a potential $2 trillion dollar market for "cognitive computing"

AI Market by region



What is AI?

Wikipdeia: Artificial intelligence (AI) is intelligence exhibited by machines. In computer science, the field of AI research defines itself as the study of "intelligent agents".

- The field of AI research was founded at a conference at Dartmouth College in 1956
- John McCarthy is generally regarded as the father of AI.
- Could not make much progress due to low computational power (“AI Winter”)
- It was revived in late 90's

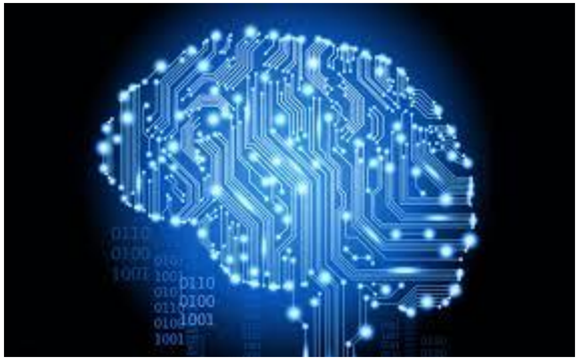
Deep learning

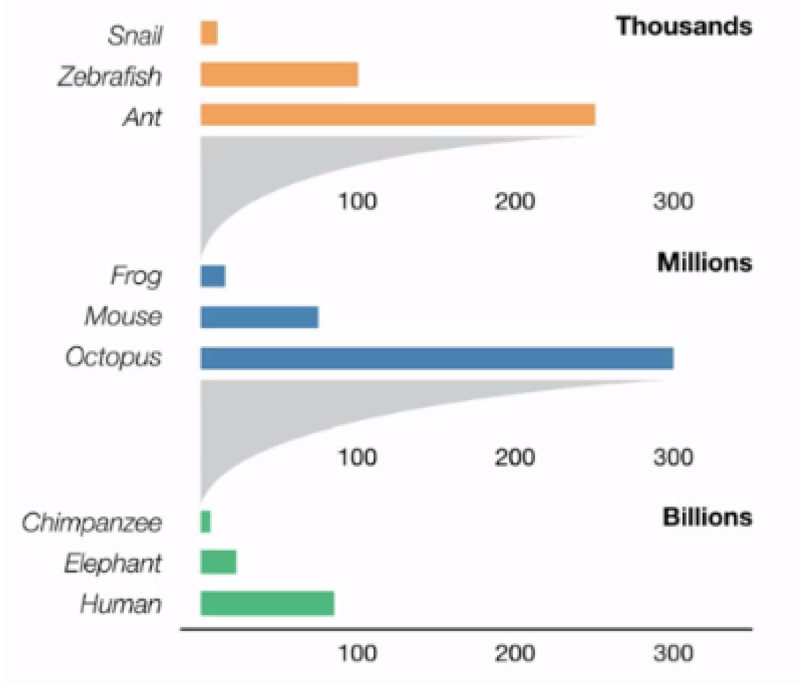
Part of the machine learning field of learning representations of data. Exceptional effective at learning patterns.

Utilizes learning algorithms that derive meaning out of data by using a hierarchy of multiple layers that mimic the neural networks of our brain.

If you provide the system tons of information, it begins to understand it and respond in useful ways.

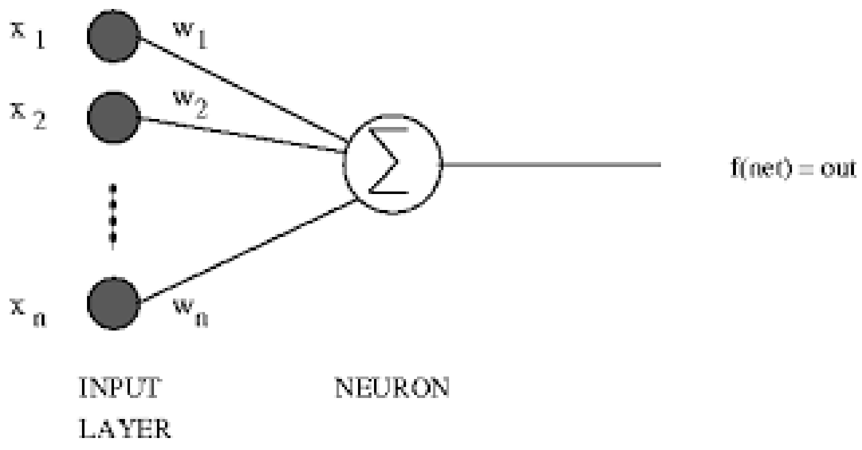
Artificial Neural Networks

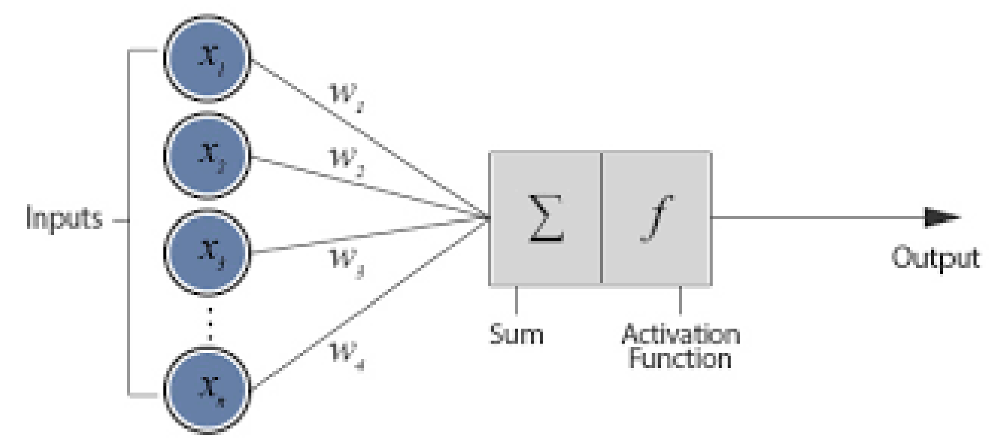


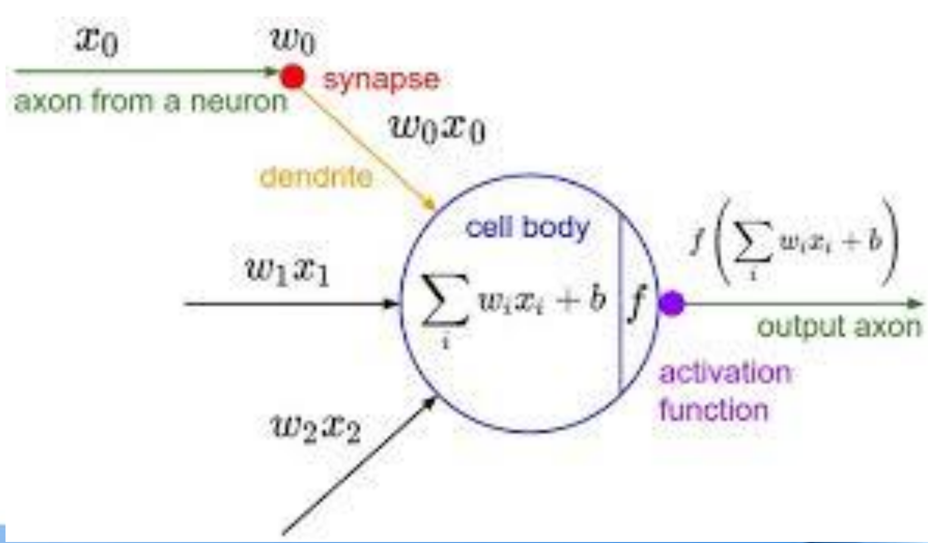


Number of Neurons

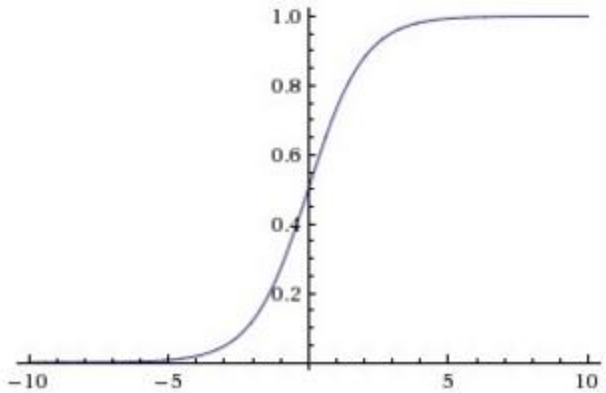
Artificial Neuron



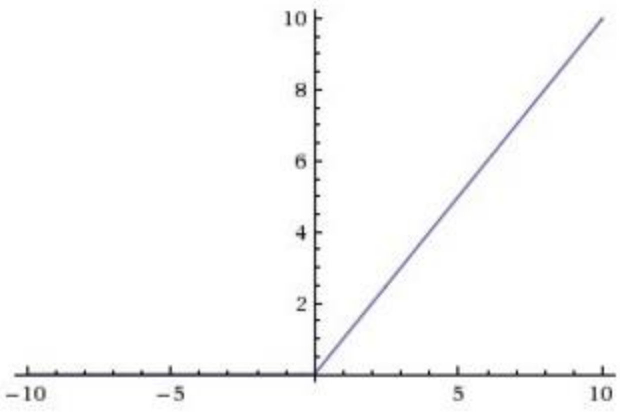




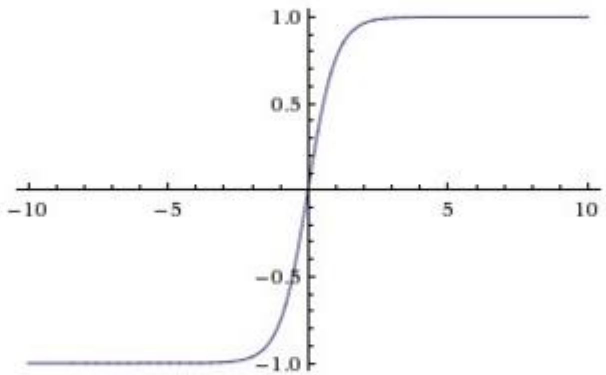
Activation functions



Sigmoid

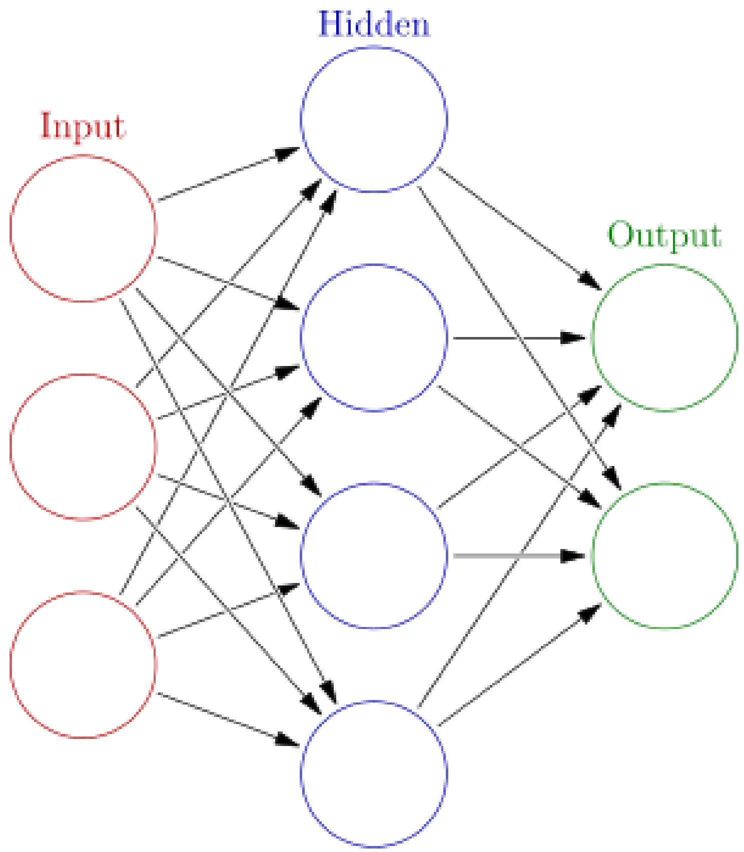


Relu

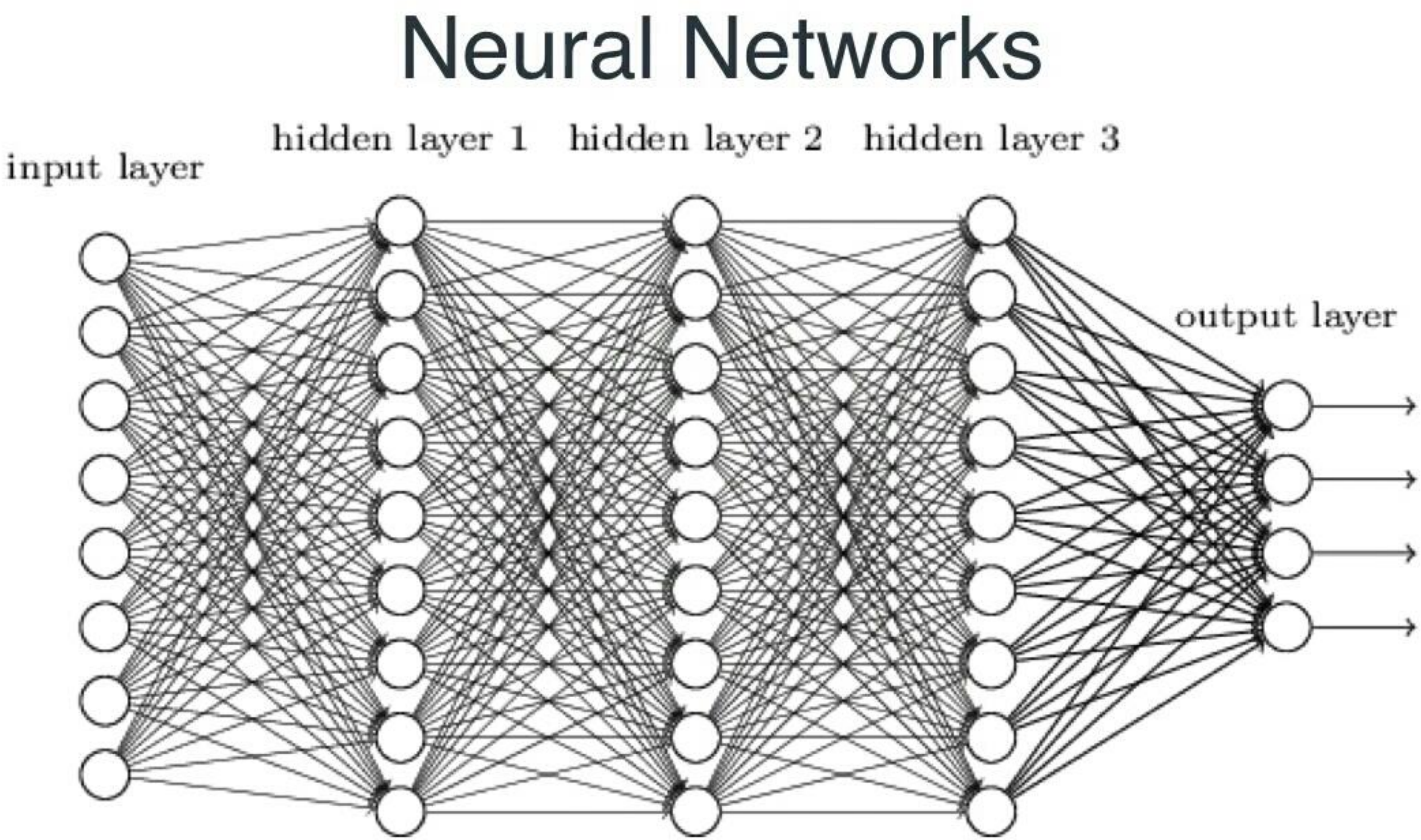


tanh

Neural Networks

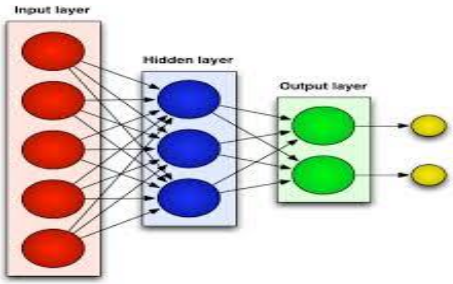


Deep Neural Networks



Types of Deep Neural network

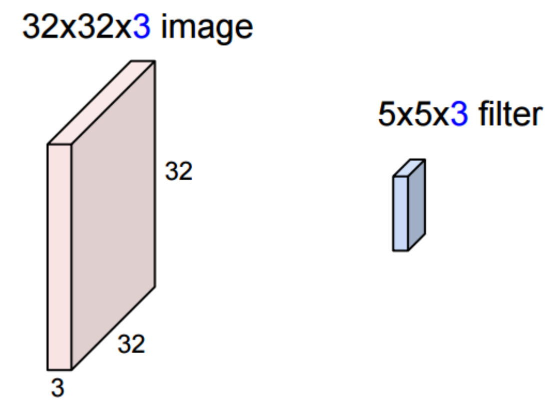
Feed forward Neural network:



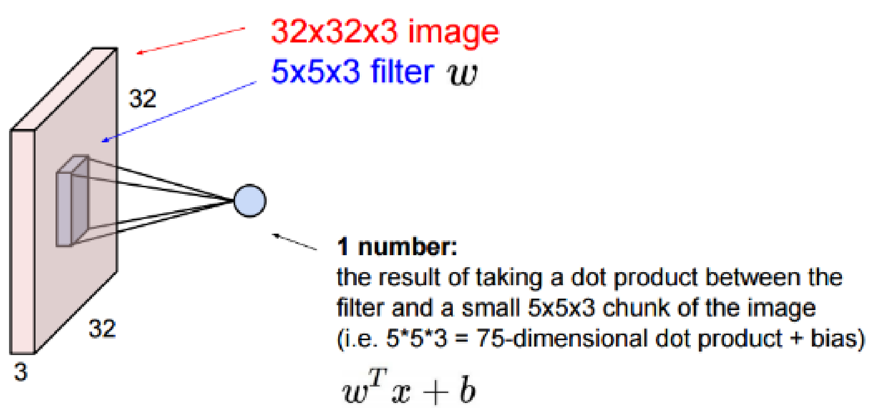
Convolutional neural network (CNN)

- Input layer/picture consists of 32 x 32 pixels with 3 colors (Red, Green & Blue) (32 x 32 x 3)
- Convolution layer is formed by running a filter (5 x 5 x 3) over Input layer which will result in (28 x 28 x 1)

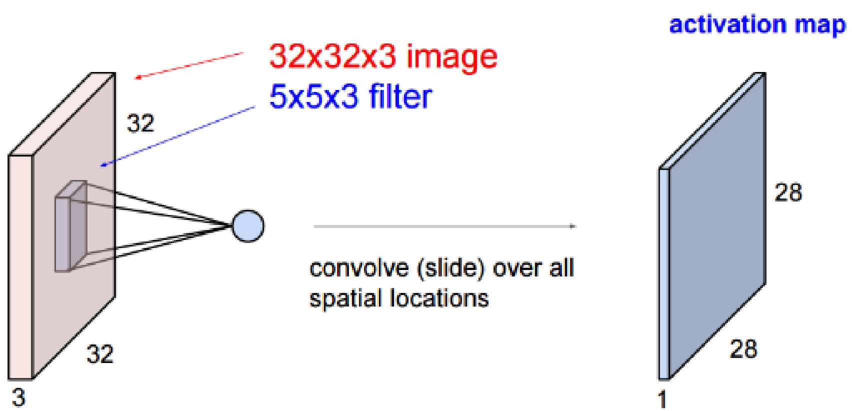
Input Layer & Filter



Running filter over Input Layer to form Convolution layer



Complete Convolution Layer from filter

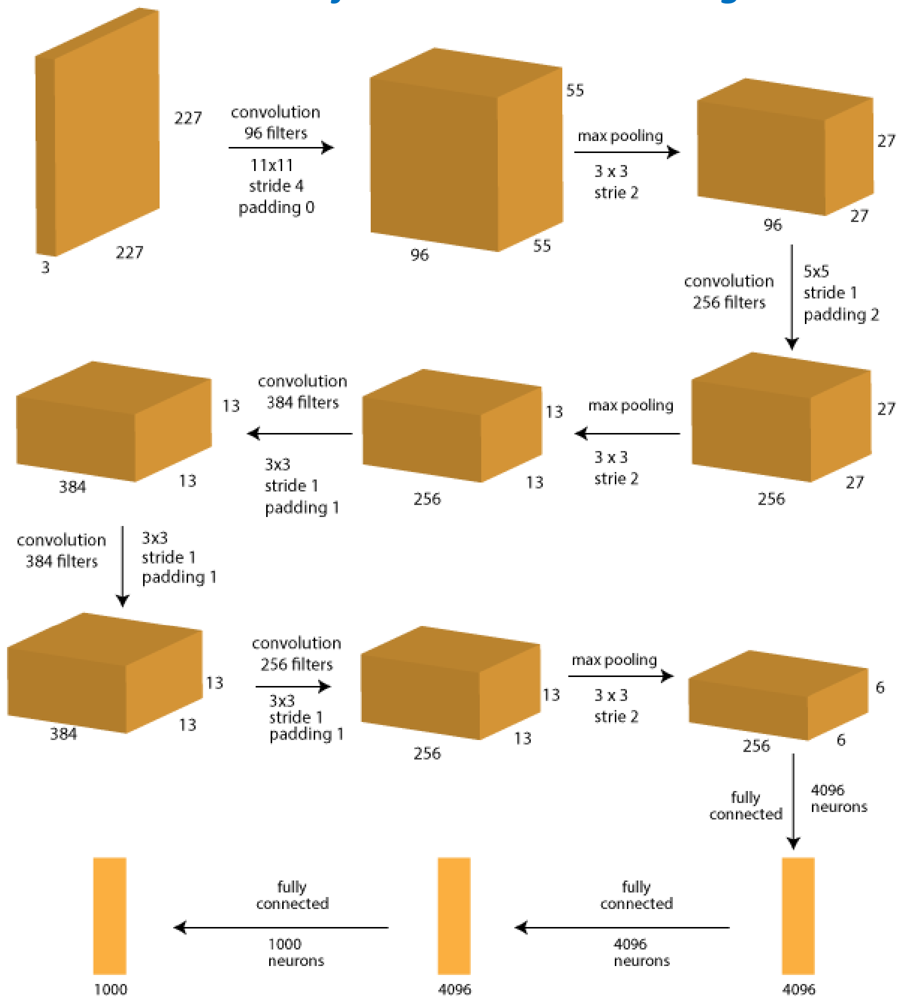


- **Alex net Architecture:** Alex Net won the IMAGENET challenge competition during 2012

- Layer 0: Input image **(227 * 227 * 3 ~= 150k)**
- Layer 1: Convolution with 96 filters, size 11×11, stride 4, padding 0
- Layer 2: Max-Pooling with 3×3 filter, stride 2
- Layer 3: Convolution with 256 filters, size 5×5, stride 1, padding 2
- Layer 4: Max-Pooling with 3×3 filter, stride 2
- Layer 5: Convolution with 384 filters, size 3×3, stride 1, padding 1
- Layer 6: Convolution with 384 filters, size 3×3, stride 1, padding 1
- Layer 7: Convolution with 256 filters, size 3×3, stride 1, padding 1
- Layer 8: Max-Pooling with 3×3 filter, stride 2
- Layer 9: Fully Connected with 4096 neuron
- Layer 10: Fully Connected with 4096 neuron
- Layer 11: Fully Connected with 1000 neurons (classes to predict)

Total memory required 24M * 4 bytes ~= 93 MB/image (only forward !~ *2 for bwd)

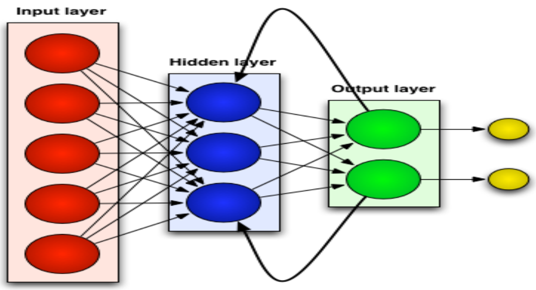
Alex Net for IMAGENET Challenge 2012



Types of Deep Neural network

Backward propagation

Recurrent neural network (RNN):



- Recurrent neural networks are very much useful in sequence remembering, time series forecasting, Image captioning, machine translation etc.
- RNNs are useful in building A.I. Chabot in which sequence of words with all syntaxes & semantics would be remembered and subsequently provide answers to given questions

Recurrent Neural Networks

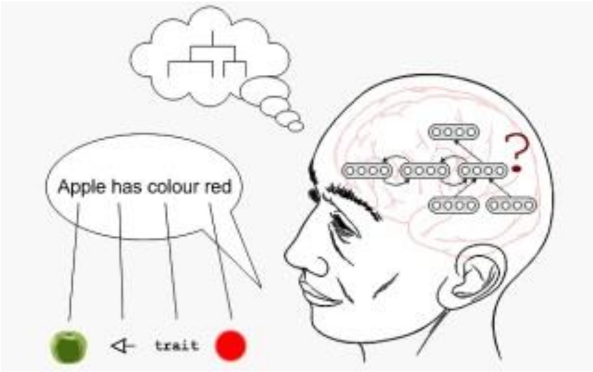
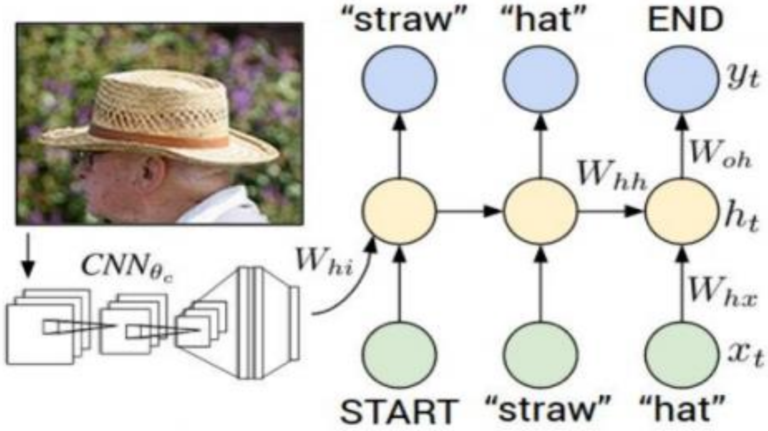


Image Captioning using Convolutional and Recurrent Neural Network



Application of RNN in A.I. Chatbot

- Recurrent neural network is used for processing sequence of vectors x by applying a recurrence formula at every time step

$$h_t = f_W(h_{t-1}, x_t)$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$

$$\boxed{h_t} = \boxed{f_W}(\boxed{h_{t-1}}, \boxed{x_t})$$

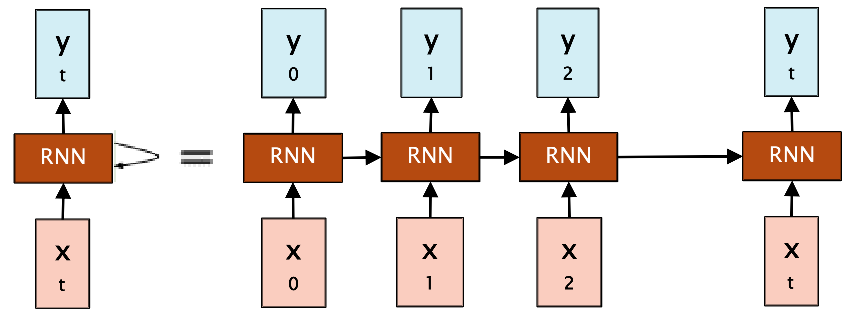
new state

old state

input vector at some time step

some function with parameters W

Recurrent Neural Network



Vanilla Network

one to one

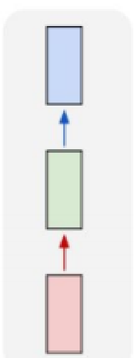
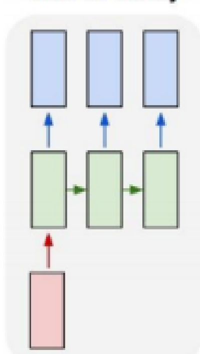


Image Captioning

(image -> Seq. of words)

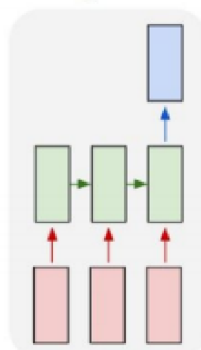
one to many



Sentiment Classification

(Seq. of words -> Sentiment)

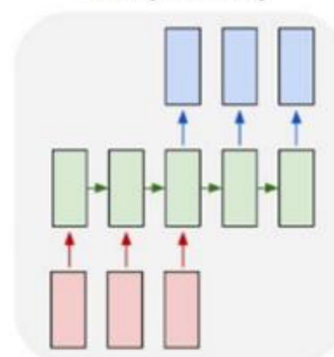
many to one



Machine Translation

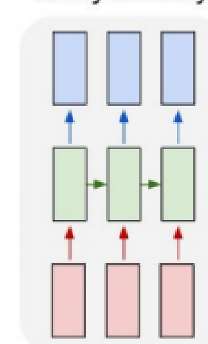
(Seq. of words -> Seq. of words)

many to many



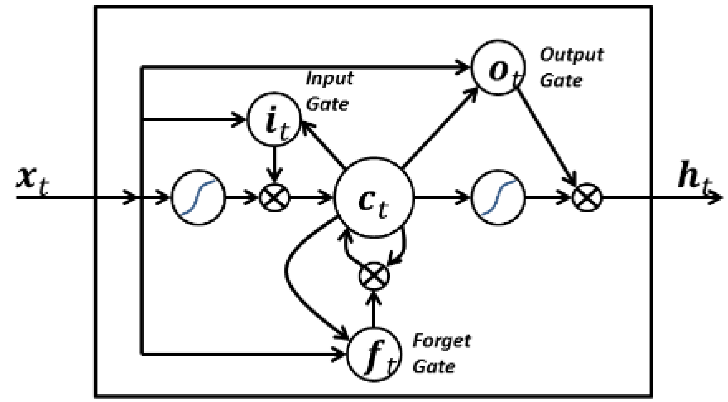
Video Classification on frame level

many to many



- **LSTM (Long Short Term Memory):** LSTM is an artificial neural network contains LSTM blocks in addition to regular network units. LSTM block contains gates that determine when the input is significant enough to remember, when it should continue to remember or when it should forget the value and when it should output the value

LSTM Cell



RNN & LSTM formula

RNN:

$$\boxed{h_t^l = \tanh W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}}$$

$h \in \mathbb{R}^n$. $\quad W^l\ [n \times 2n]$

LSTM:

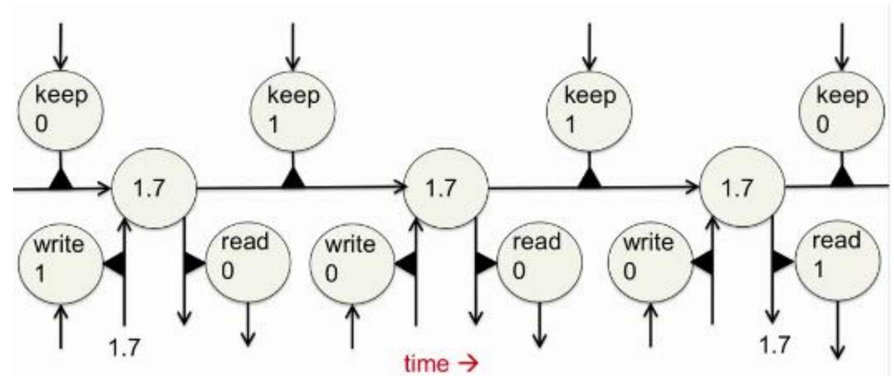
$W^l\ [4n \times 2n]$

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g$$

$$h_t^l = o \odot \tanh(c_t^l)$$

LSTM Working Principle (Backpropagation through a memory cell)



- **Case Study:** NIFTY prediction

```
tsteps = 1; batch_size = 1 ; epochs = 50

model = Sequential()
model.add(LSTM(1000,
               batch_input_shape=(batch_size, tsteps, 1),
               return_sequences=True,
               stateful=True))

model.add(LSTM(1000,
               batch_input_shape=(batch_size, tsteps, 1),
               return_sequences=True,
               stateful=True))

model.add(LSTM(1000,
               batch_input_shape=(batch_size, tsteps, 1),
               return_sequences=True,
               stateful=True))

model.add(LSTM(1000,
               batch_input_shape=(batch_size, tsteps, 1),
               return_sequences=False,
               stateful=True))

model.add(Dense(1))
model.add(Activation("linear"))
model.compile(loss='mse', optimizer='rmsprop')

print('Training')
for i in range(epochs):
    print('Epoch', i, '/', epochs)
    model.fit(X_train,
              y_train,
              batch_size=batch_size,
              verbose=1,
              nb_epoch=1,
              shuffle=False)
    model.reset_states()

print('Predicting')
predicted_output = model.predict(X_test, batch_size=batch_size)
```

Layer 1 consists of 1000 Recurrent LSTM neurons

Layer 2 consists of 1000 Recurrent LSTM neurons

Layer 3 consists of 1000 Recurrent LSTM neurons

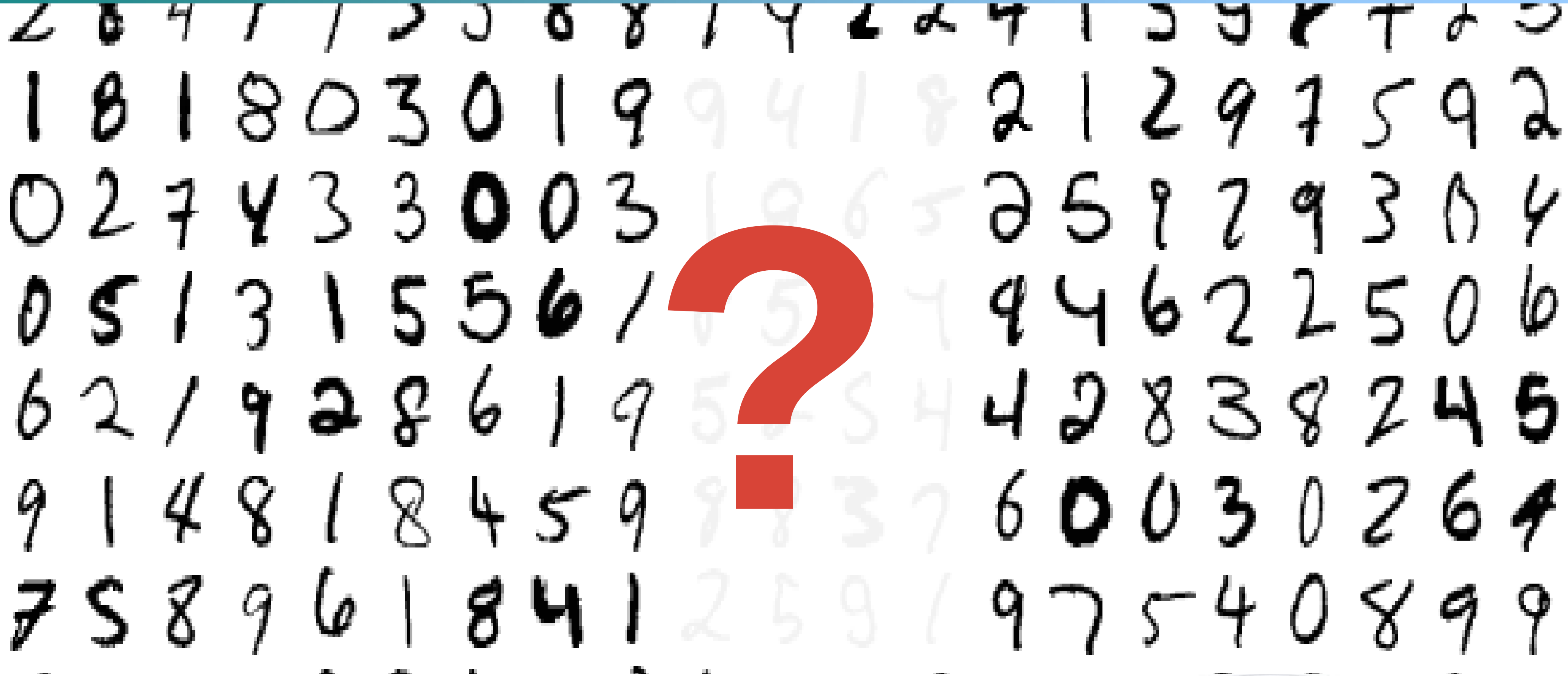
Layer 4 consists of 1000 Recurrent LSTM neurons with return sequence False

Output Layer consists of 1 neuron with linear activation function

NIFTY 1 Year EOD data

“Hello World” of Deep Neural Network?

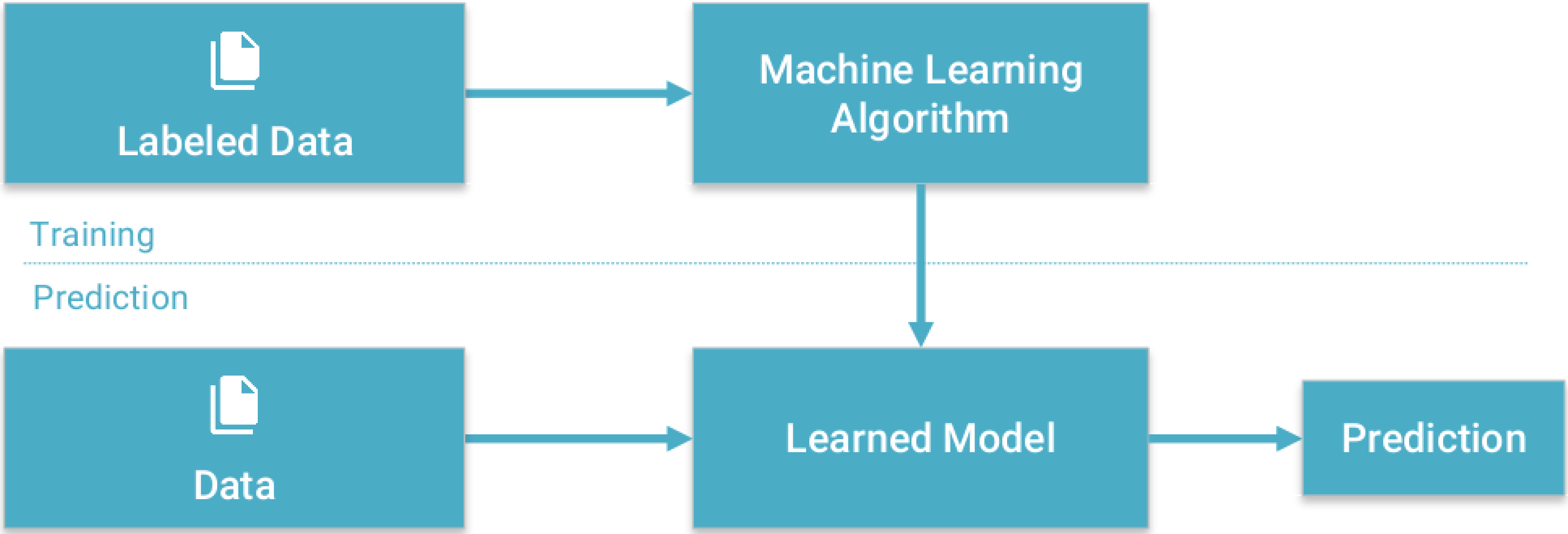
Handwritten digits classification - MNIST



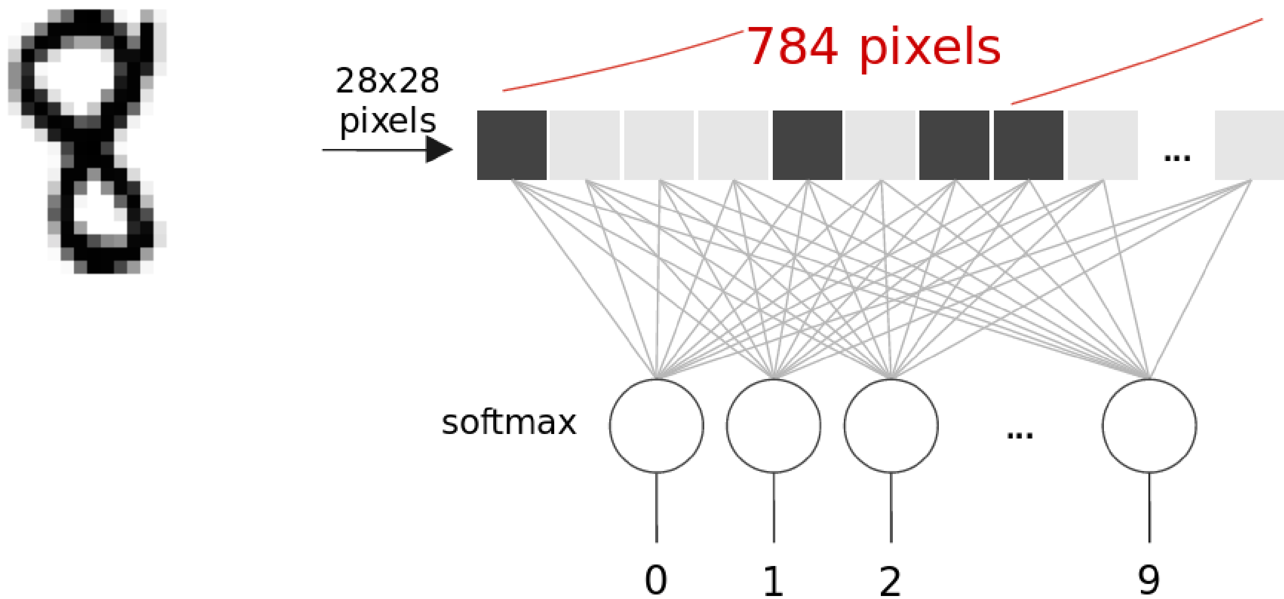
Demo

MNIST = Mixed National Institute of Standards and Technology - Download the dataset at http://yann.lecun.com/exdb/mnist/

Machine learning



Simple Softmax



weighted sum of
all pixels + bias

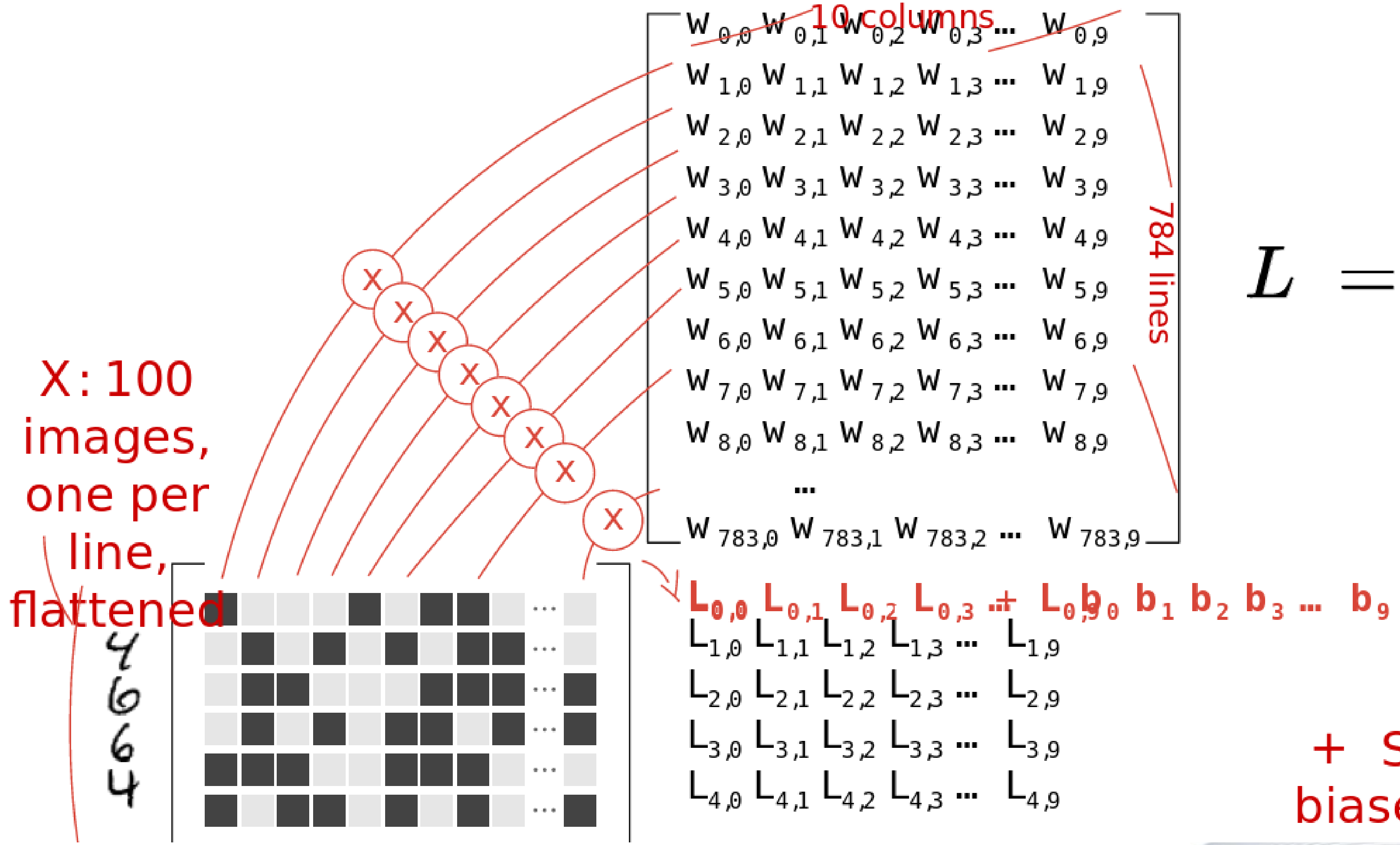
$$softmax(L_n) = \frac{e^{L_n}}{||e^{L}||}$$

neuron
outputs

Softmax output equation

$$Y = softmax(X.W + b)$$

Training – batch of 100



broadcast

$$L = X.W + b$$

\+ Same 10 biases on all

Predictions

Y[100, 10]

Images

X[100, 784]

Weights

W[784,10]

Biases

b[10]

$$Y = softmax(X.W + b)$$

applied line by line

matrix multiply

broadcast on all lines

Gradient Descent/Cross Entropy/Learning Rate

Sample Tensor Flow python code:

Learning rate

```
Optimizer =tf.train.GradientDescentOptimizer(0.003)
train_step = optimizer.minmize (cross_entropy)
```

A measure of the error between actual and predicted values

Gradient Descent/Cross Entropy/Learning Rate

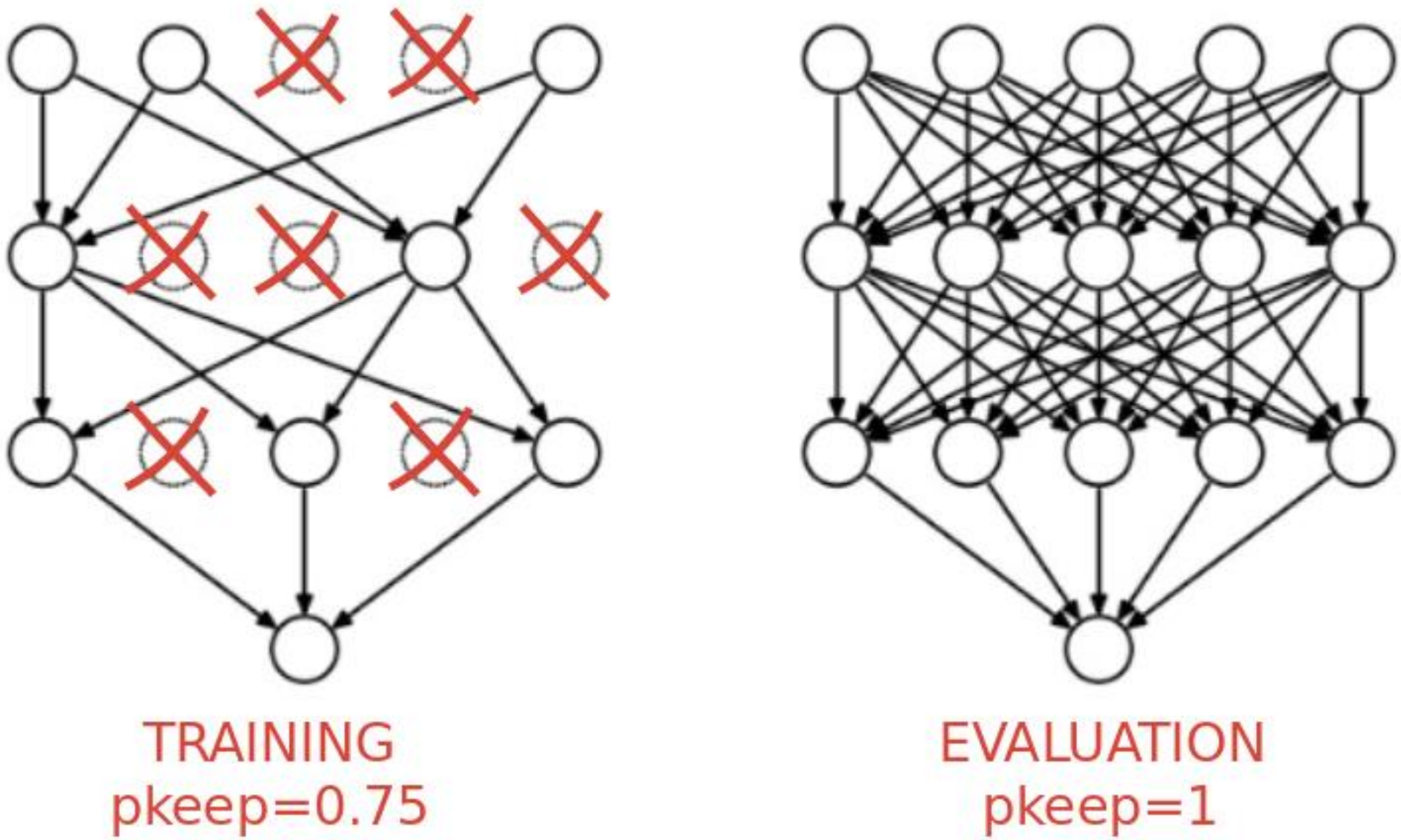
Sample Tensor Flow python code:

Learning rate

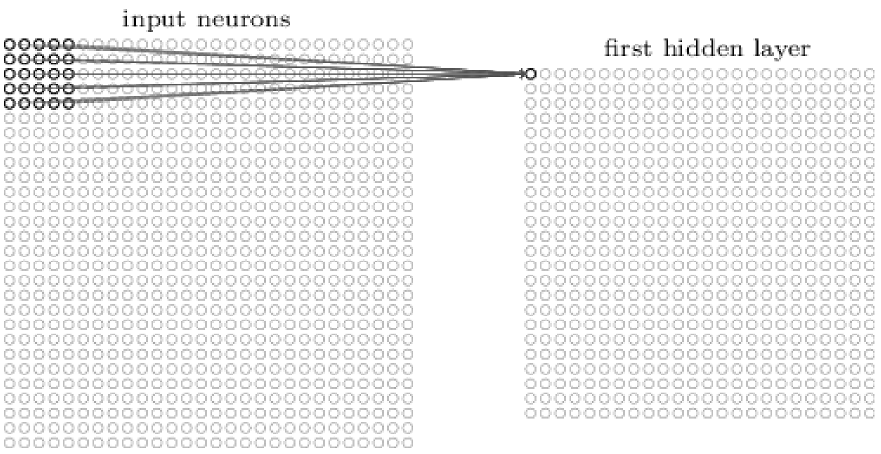
```
Optimizer =tf.train.GradientDescentOptimizer(0.003)
train_step = optimizer.minmize (cross_entropy)
```

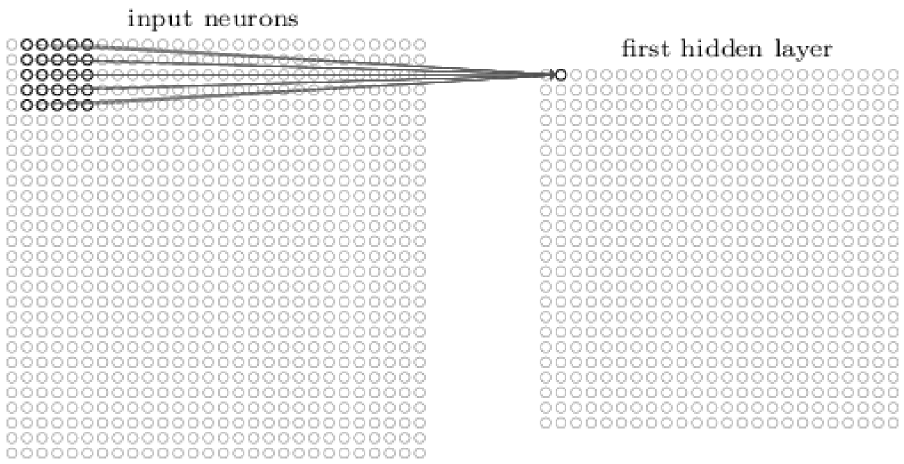
A measure of the error between actual and predicted values

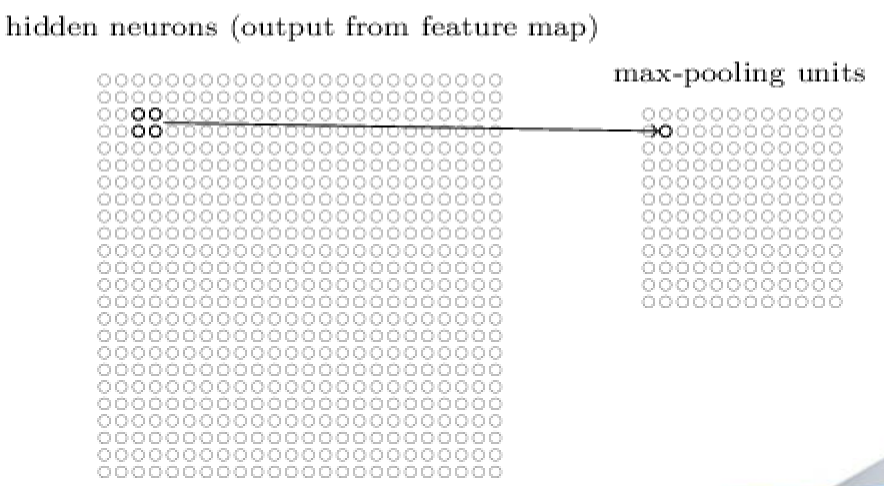
Regularization - Dropout



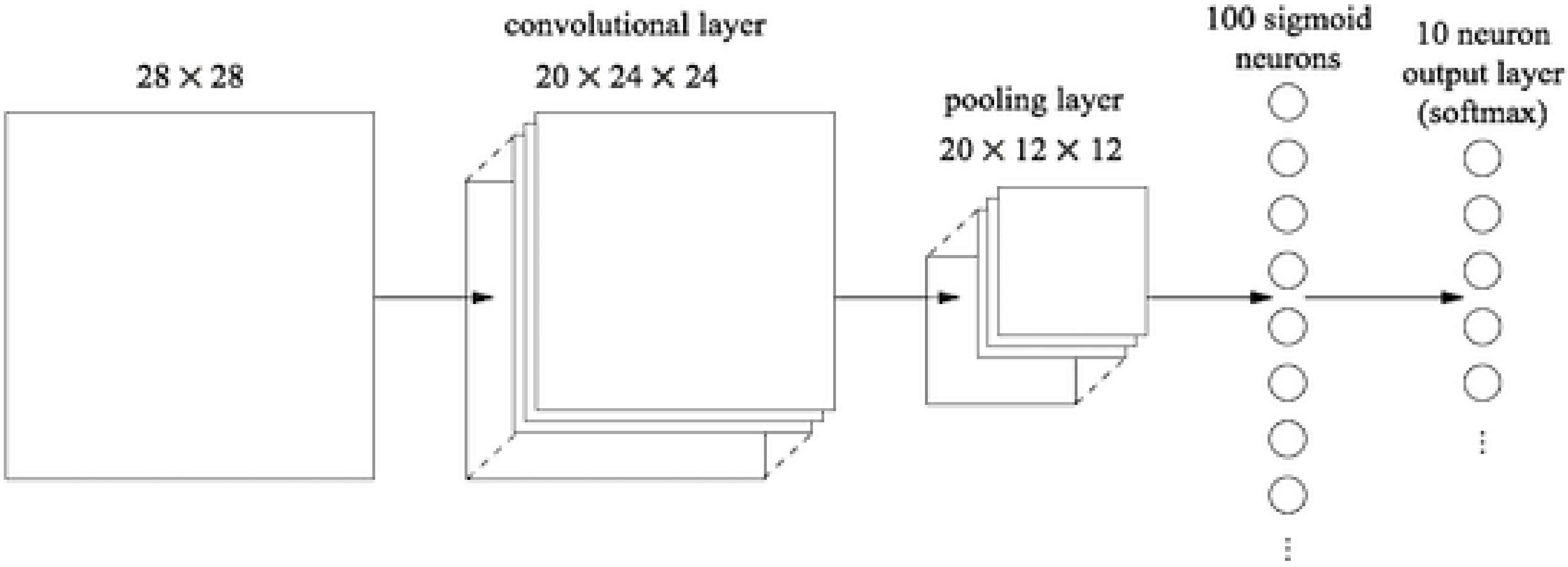
CNN Traditional Architecture



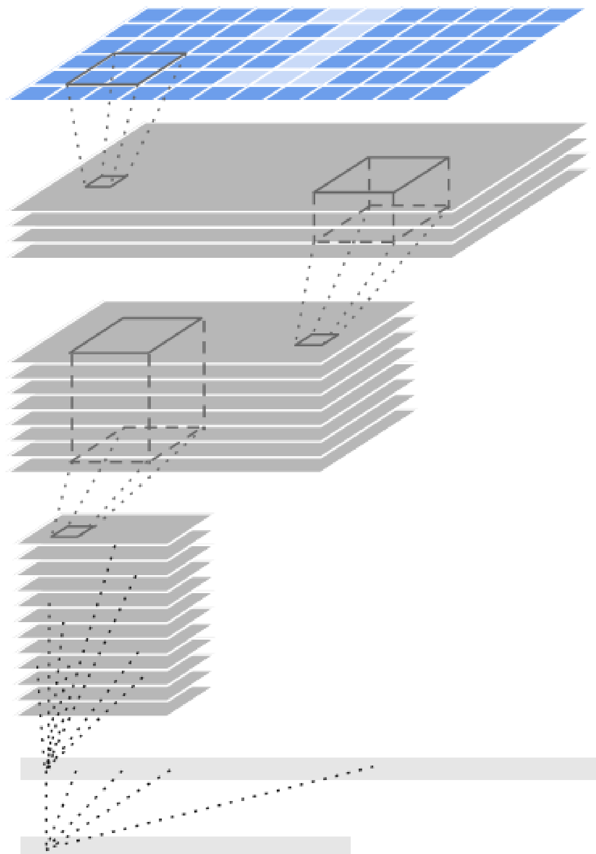




CNN Traditional Architecture



CNN Architecture



28x28x1

28x28x4

convolutional layer, 4 channels
W1[5, 5, 1, 4] stride 1

14x14x8

convolutional layer, 8 channels
W2[4, 4, 4, 8] stride 2

7x7x12

convolutional layer, 12 channels
W3[4, 4, 8, 12] stride 2

200

fully connected layer W4[7x7x12, 200]

10

softmax readout layer W5[200, 10]

Face detection/recognition