Introduction to Deep Learning

Dr. Arjun Jain

Agenda

- History and Evolution of Neural Networks
- Tipping Point for Deep Learning
- Why Deep Learning and the Data Driven Paradigm
- Training a classifier (without any trainable parameters) – KNN
- Training a classifier (with trainable parameters) – Linear Classifier

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History

Hubel & Wiesel,

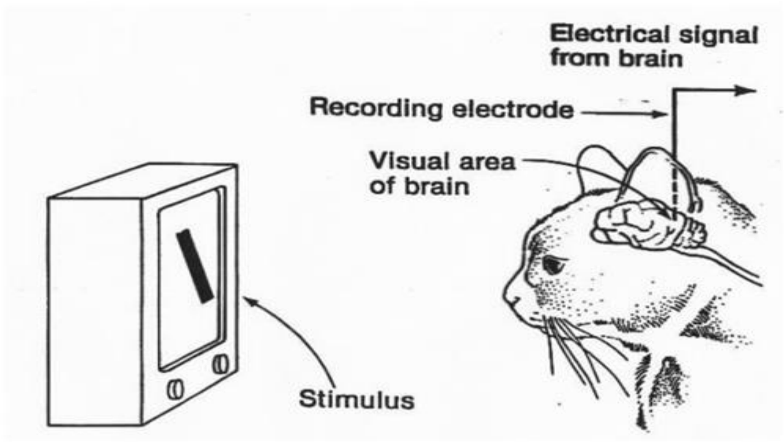
1959

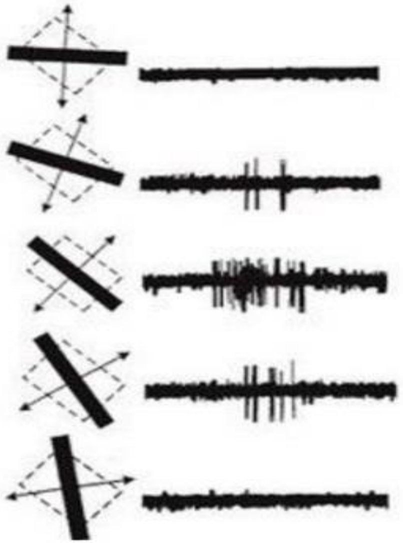
RECEPTIVE FIELDS OF SINGLE NEURONES IN
THE CAT'S STRIATE CORTEX

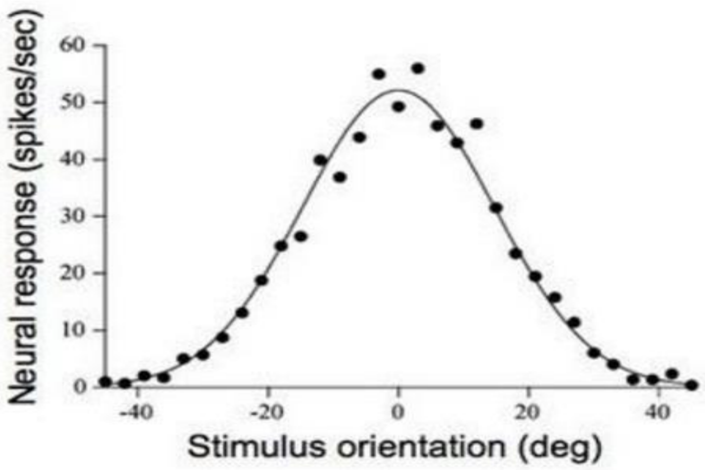
1962

RECEPTIVE FIELDS, BINOCULAR INTERACTION
AND FUNCTIONAL ARCHITECTURE IN THE CAT'S VISUAL CORTEX

1968...







Sourced from: Receptive fields of single Neurons in the Cat's Striate Cortex by D.H.Hubel and T.N.Wiesel

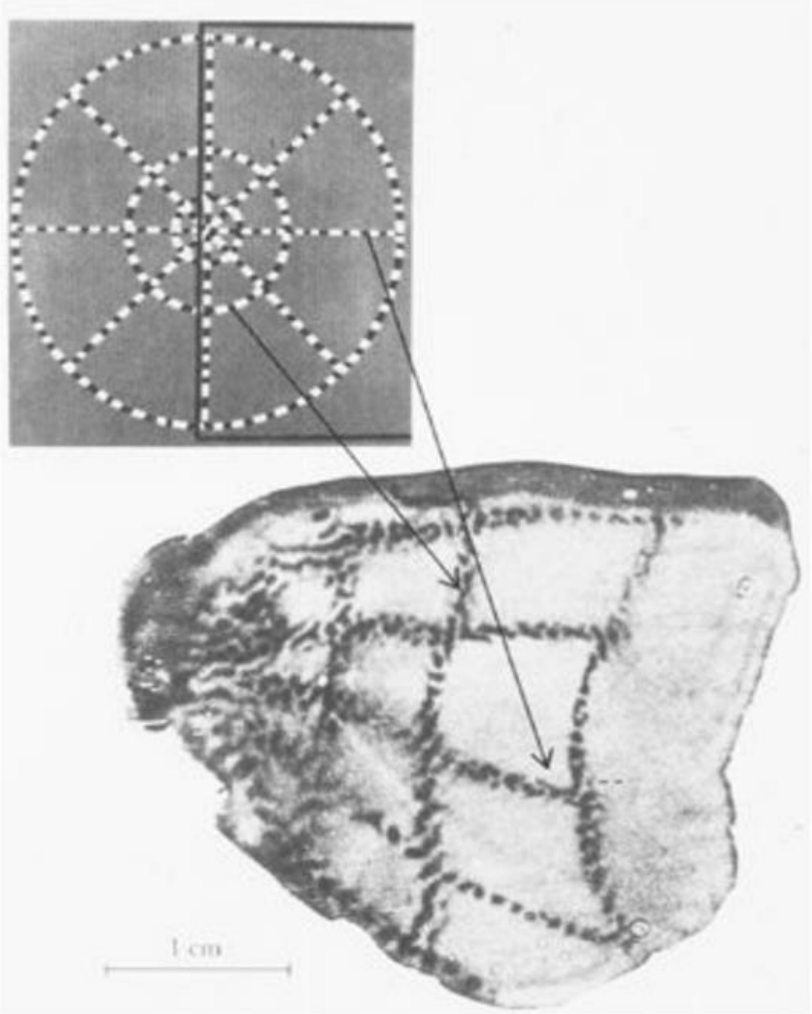
Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Video Source: Cortical Neuron, Hubel & Wiesel, Youtube (https://youtu.be/8VdFf3egwfg)

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History

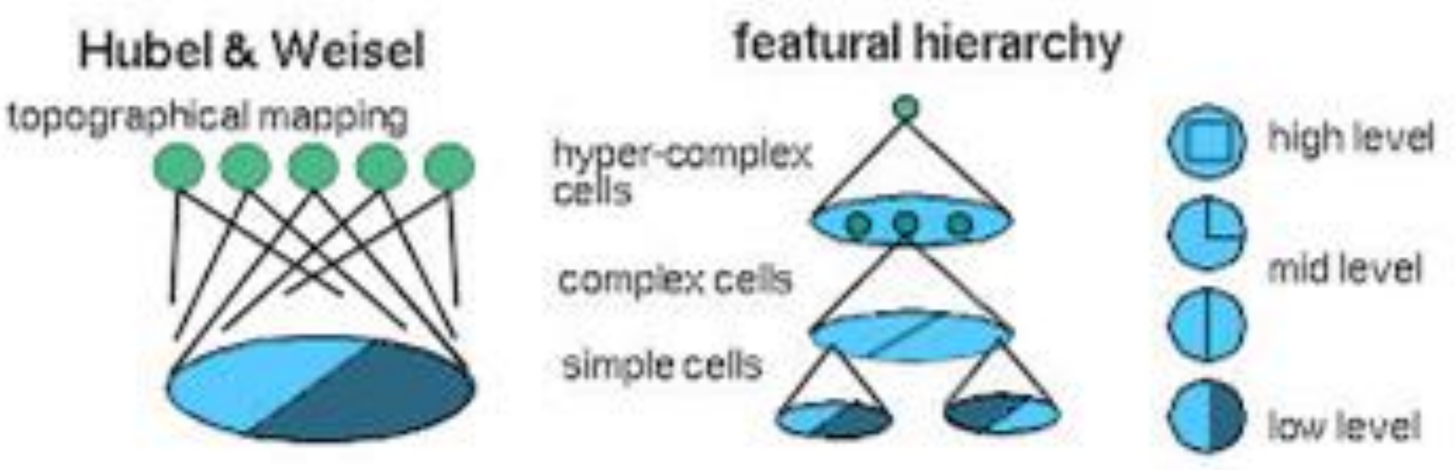
Topographical mapping in the cortex:
nearby cells in cortex represented
nearby regions in the visual field



Sourced from: Receptive fields of single Neurons in the Cat's Striate Cortex by D.H.Hubel and T.N.Wiesel

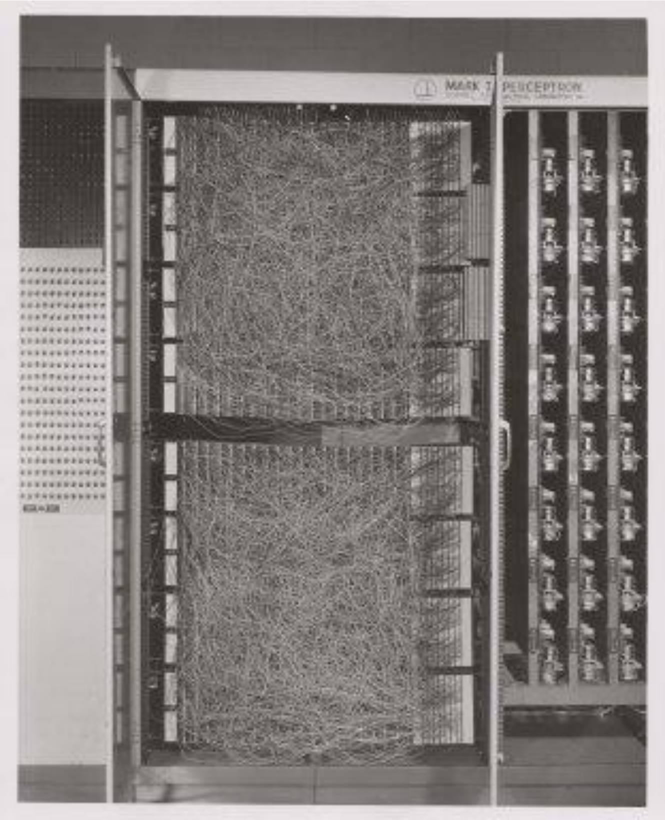
Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

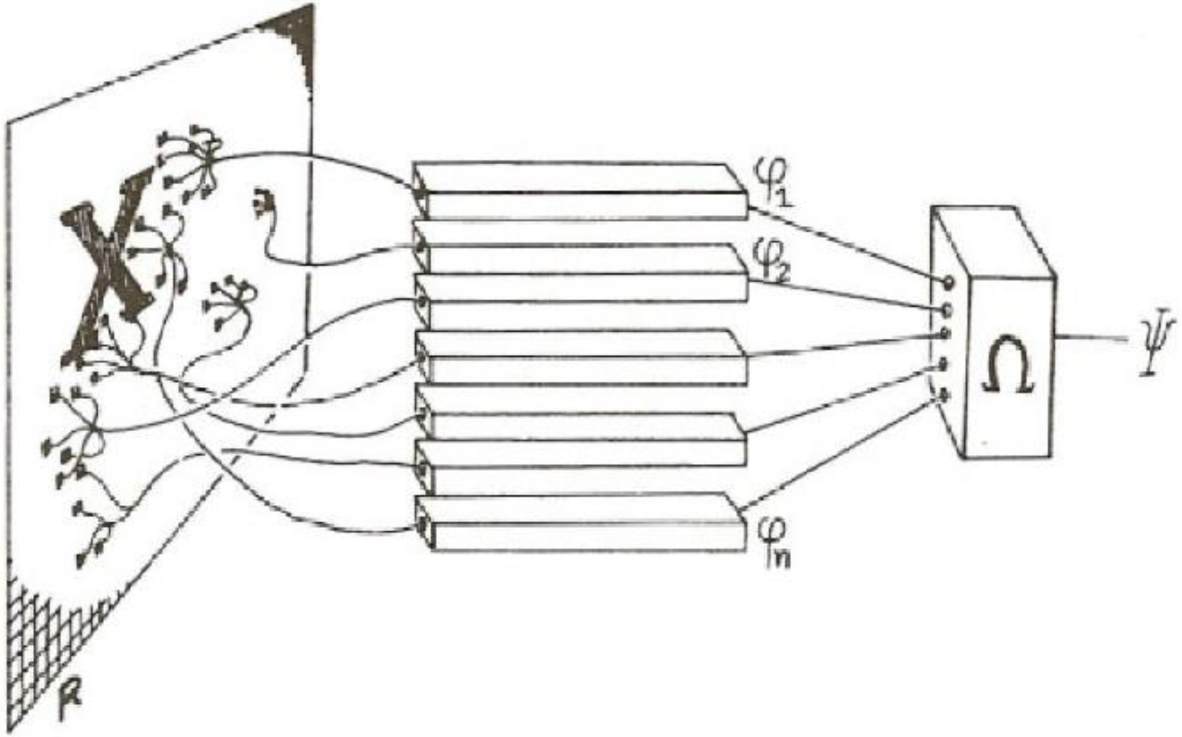
Hierarchical Organization



Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History – Mark I Perceptron – 1958





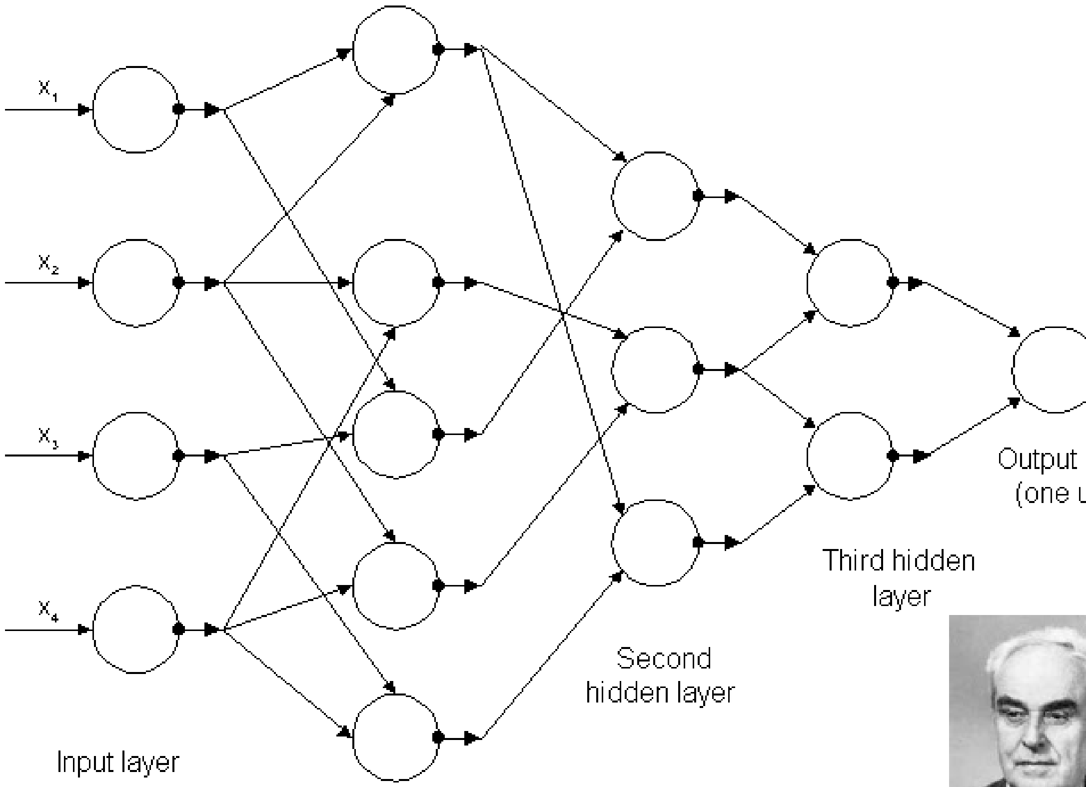
Source: Perceptrons, Wikipedia

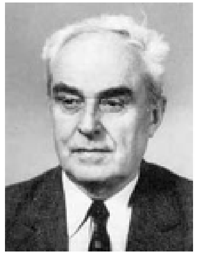
Source: Perceptrons by M. L Minsky and S. Papert, 1969

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History – The First Deep Networks

- Perceptron: single layer 1960s
- Multiple layers of non-linear features - Ivakhnenko and Lapa in 1965
- Thin but deep models with polynomial activation functions
- They did not use backpropagation



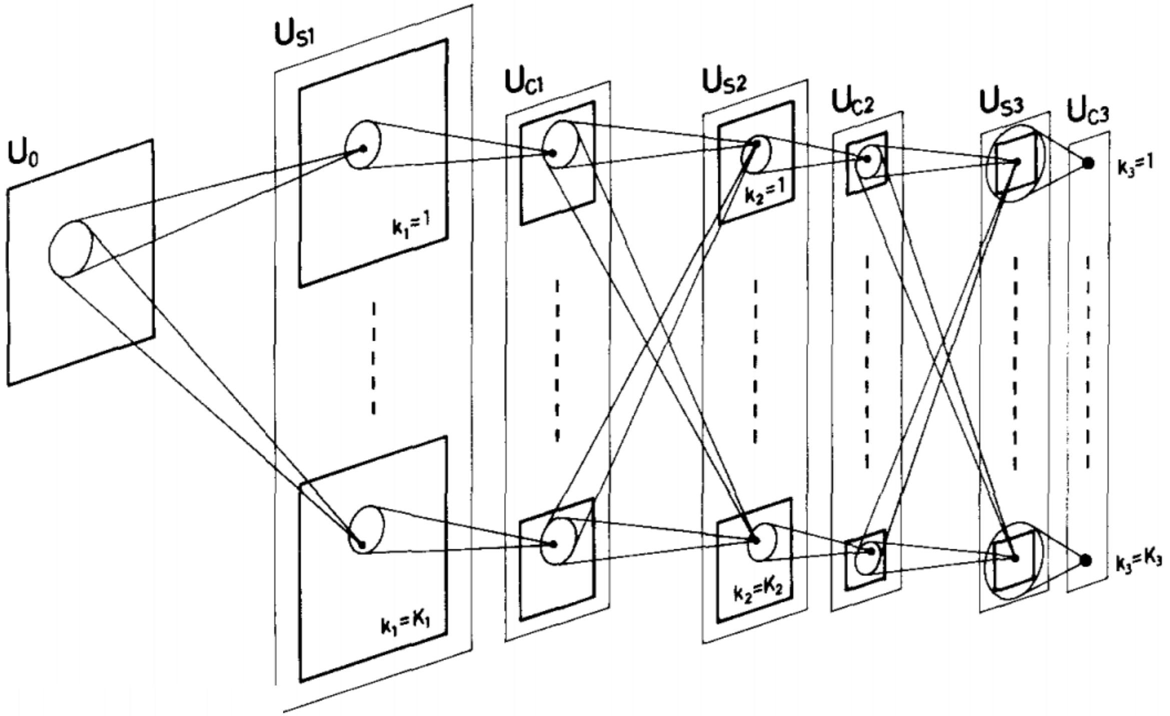


Alexey Ivakhnenko

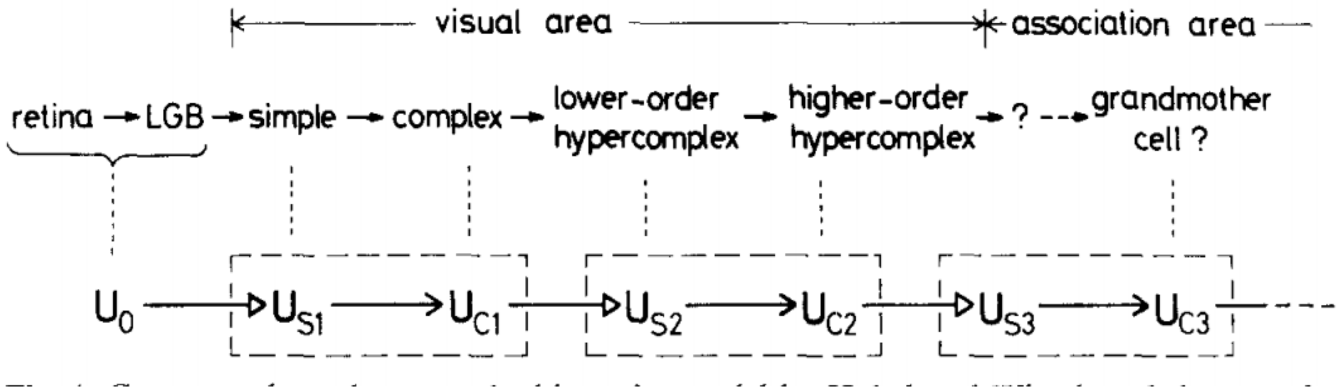
Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

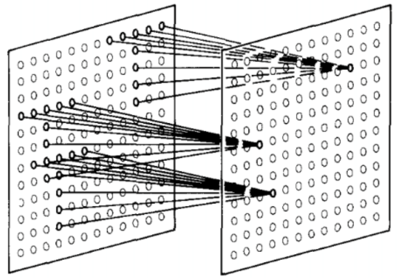
Brief History – The First ConvNet - 1980

greatlearning
Learning for Life



- Neocognitron: multiple convolutional and pooling layers similar to modern networks, but the network was trained by using a reinforcement scheme
- Did not still use backpropagation
- Translational invariant





Kunihiko Fukushima

Image Credits : Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

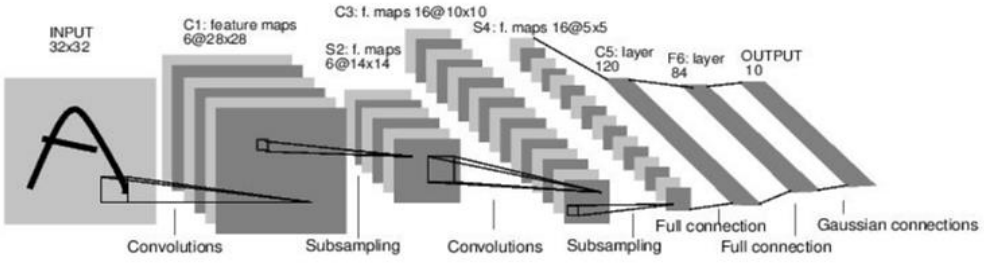
Brief History

greatlearning
Learning for Life

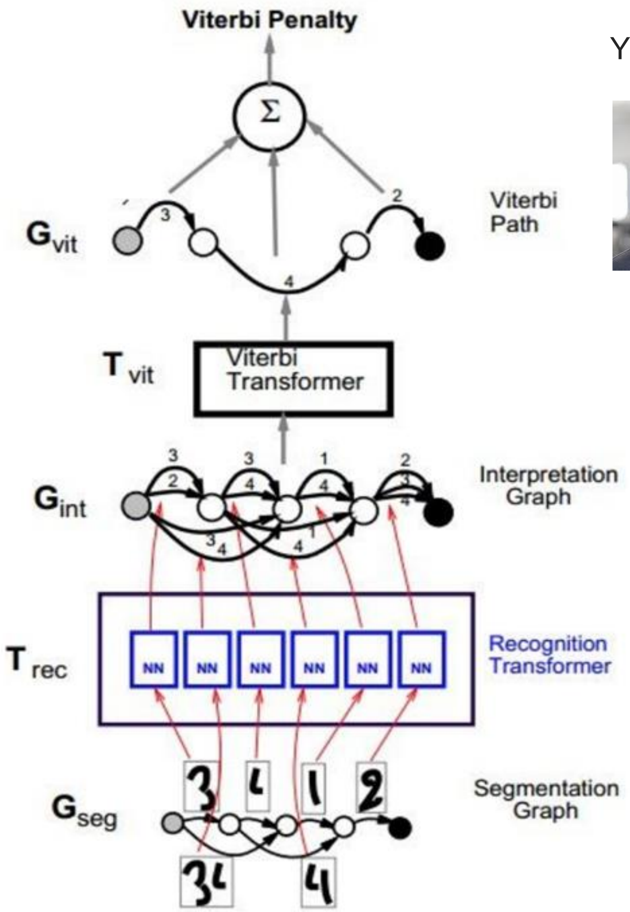
A bit of history:

Gradient-based learning applied to document recognition

[LeCun, Bottou, Bengio, Haffner 1998]



LeNet-5



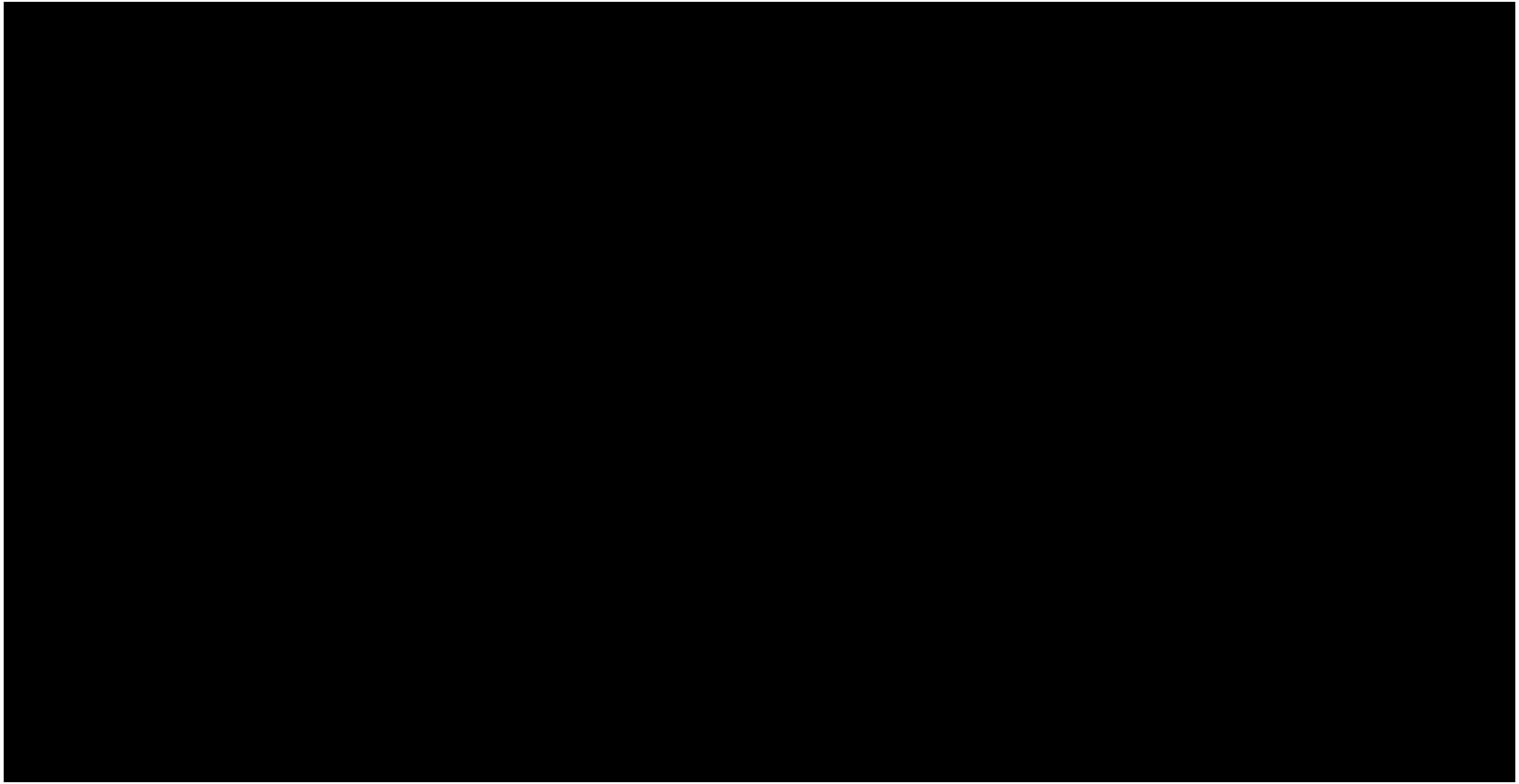
Yann LeCun

Sourced from: GradientBased Learning Applied to Document Recognition, LeCun et al. (1998)

Image Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History – LeNet-5 in Action



Video Source: Convolutional Network Demo from 1993, Youtube (https://youtu.be/FwFduRA_L6Q?t=6s)

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History – AI Winter

- Rapid advances led to a hype of artificial intelligence (similar to the buzz around deep learning today)
- Researchers made promises to solve AI and received lots of funding
- In the 1970s it became clear that those promises could not be kept, funding was cut dramatically
- The field of artificial intelligence dropped to near pseudo-science status
- Research became very difficult (little funding; publications almost never made it through peer review)
- Further advances such as SVMs with nice properties in terms of training, provable error bounds were preferred and took the front seat
- However, a handful of researchers continued further down this path

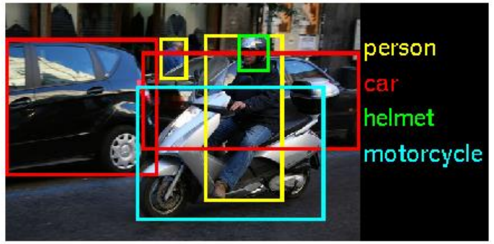
Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History – AI Winter

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History – The Tipping Point

- 2012 ILSVRC: ImageNet Large-Scale Visual Recognition Challenge – Annual World Cup of Computer Vision
- More than a million training images and 1000 categories



ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

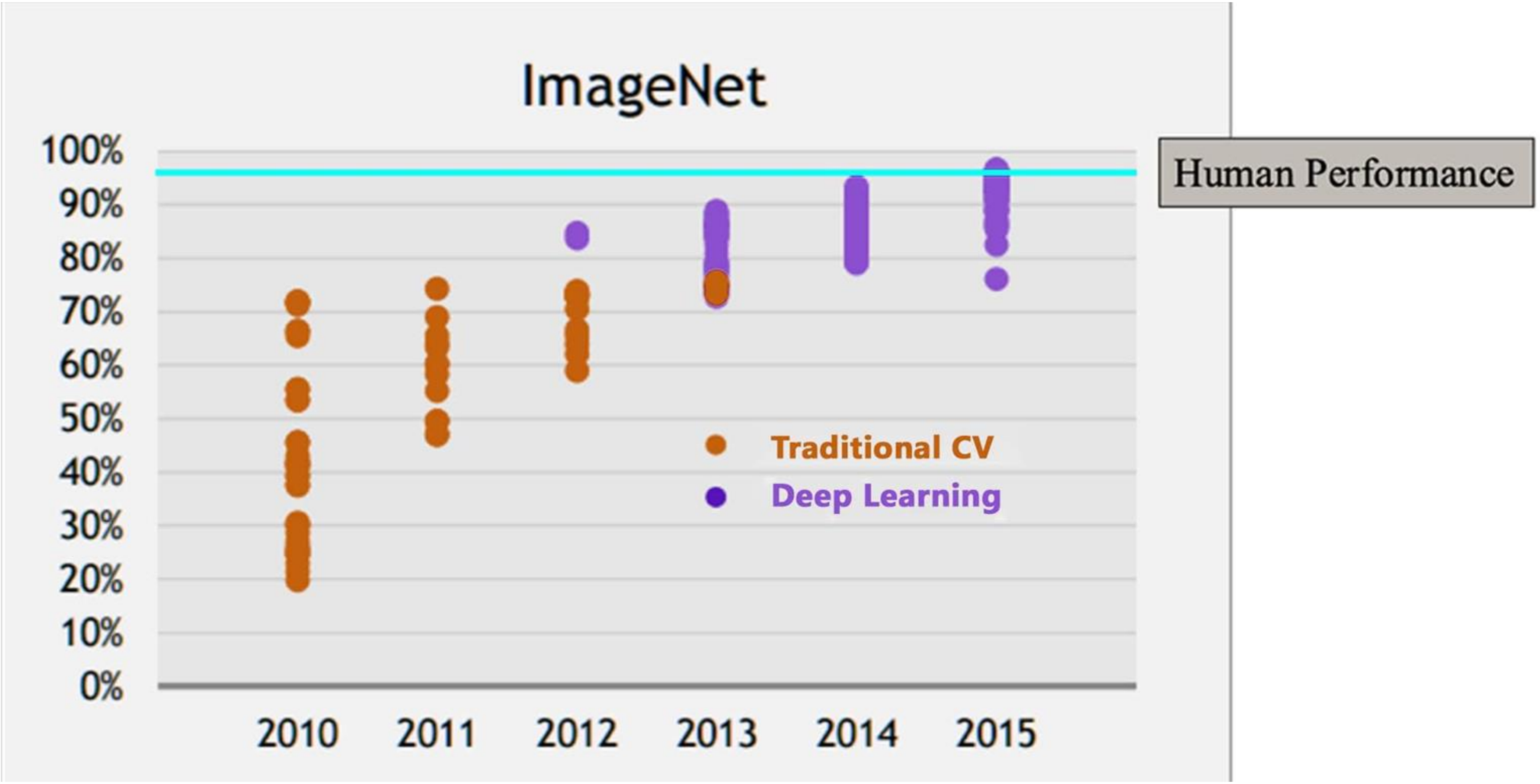
Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Brief History – The Tipping Point

- Reported 15.4% Top 5 error rate. The next best entry achieved an error of 26.2%
- > 8000 citations (last year), by today >19000!
- The coming out party for CNNs in the computer vision community
- Shocked the computer vision community. Trained end-to-end on raw pixels, without using any feature engineering methods
- From here it was apparent that deep learning would take over computer vision and that other methods would not be able to catch up

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why ConvNets?



Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

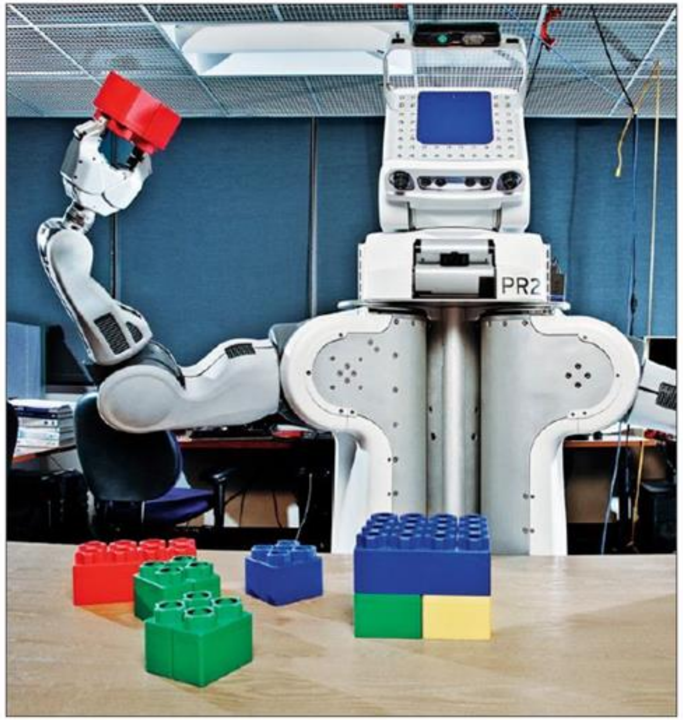
Brief History – So What Changed (since the ‘70s)?

1. **Three things**:
    a. Availability of large amounts of labeled data e.g. ImageNet
    b. Compute power – A single NVidia TITAN X card churns of 11 TFLOPS with ~3500 cores, **TITAN V**?
    c. **Algorithms:**
        i. ReLU - Found to decrease training time
        ii. Dropout – prevent overfitting to the training data

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Deep Learning – Today – One Net To Rule Them All

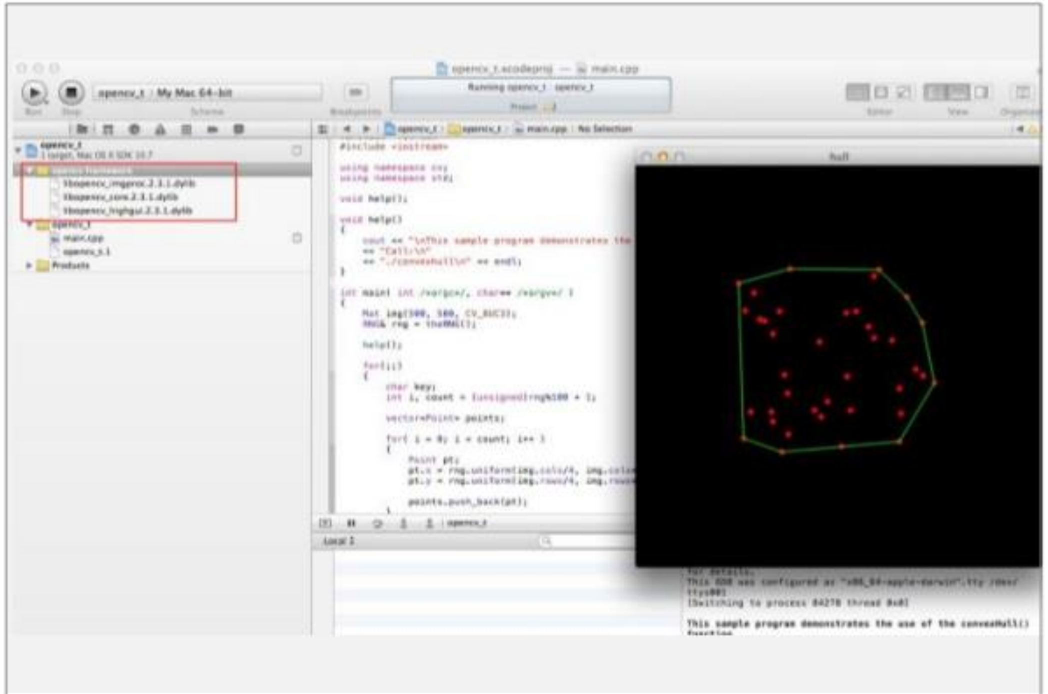
- Deep Learning == AI
- Solves problems previously unsolvable



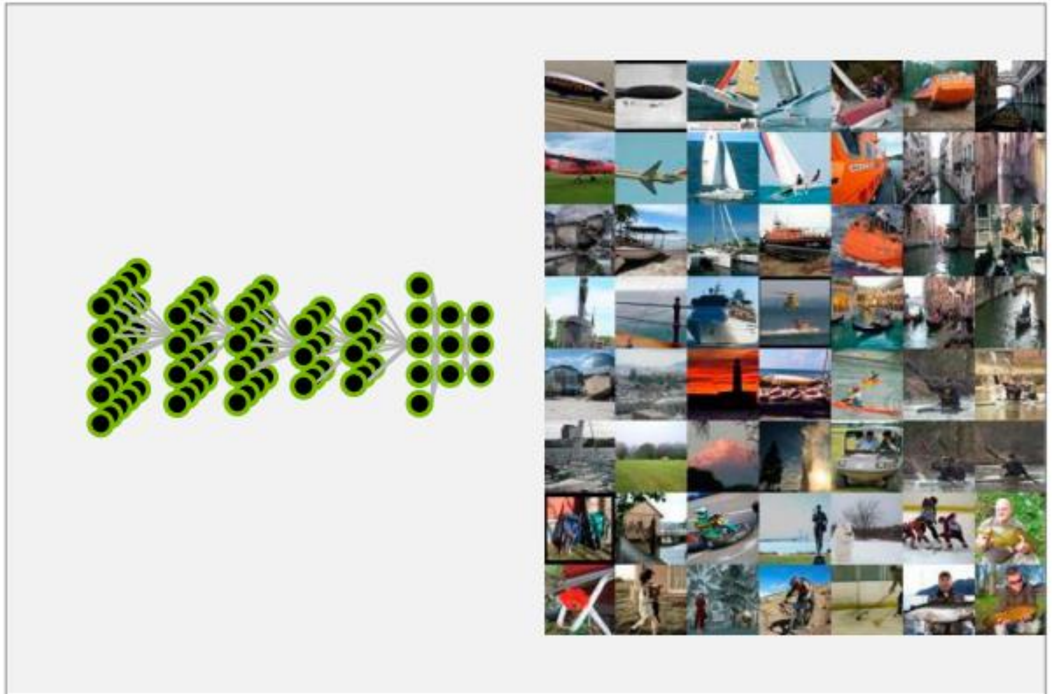
Images Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

A New Programming Model – Data Driven Paradigm



Traditional Computer Vision
Experts + Time



Deep Learning Object Detection
DNN + Data + HPC

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm?

- Consider Image Classification: a core task in Computer Vision

(assume given set of discrete labels)

{dog, cat, truck, plane, ...}

Dog

Image Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

What is Classification?

Input (Image, text, audio, etc.)

Classifier

(choose one from N labels)

{dog, cat, truck, plane, ...}

Dog

Image Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm?

Images are represented as 3D arrays of numbers, with integers between [0, 255].

E.g.
300 x 100 x 3

(3 for 3 color channels RGB)

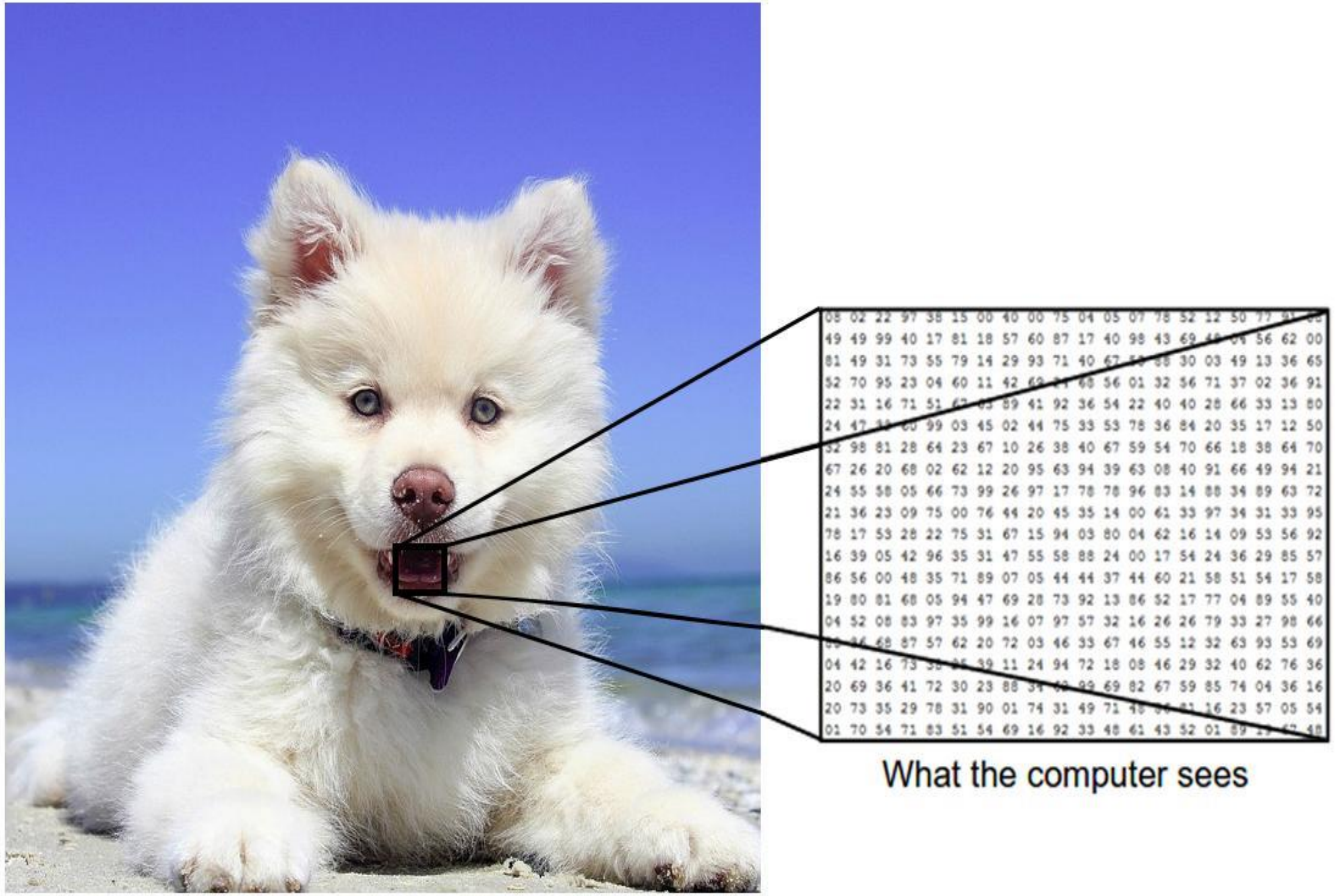


Image Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm? – Invariant to Illumination

Image Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm? – Invariant to Viewpoint

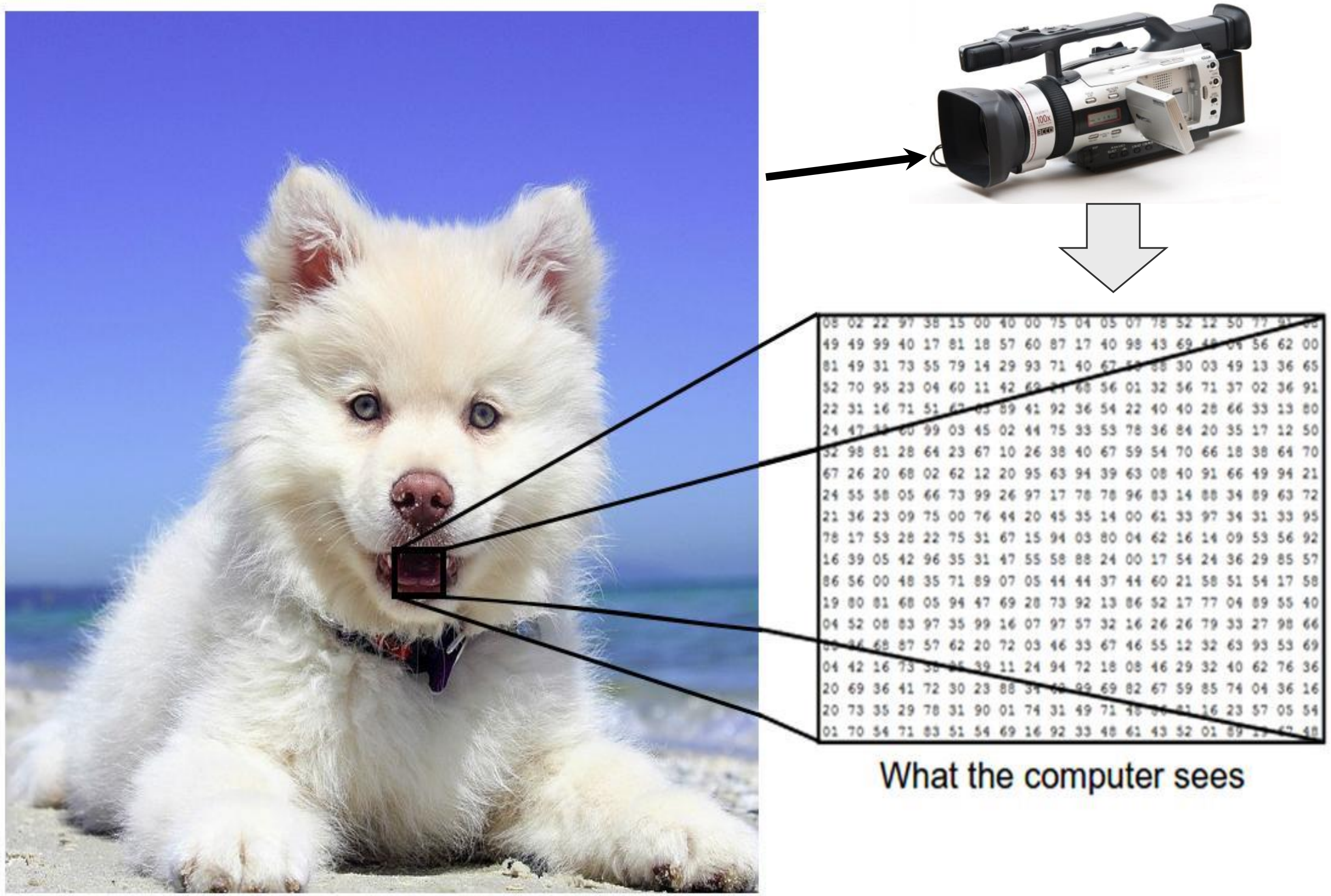


Image Source: Google

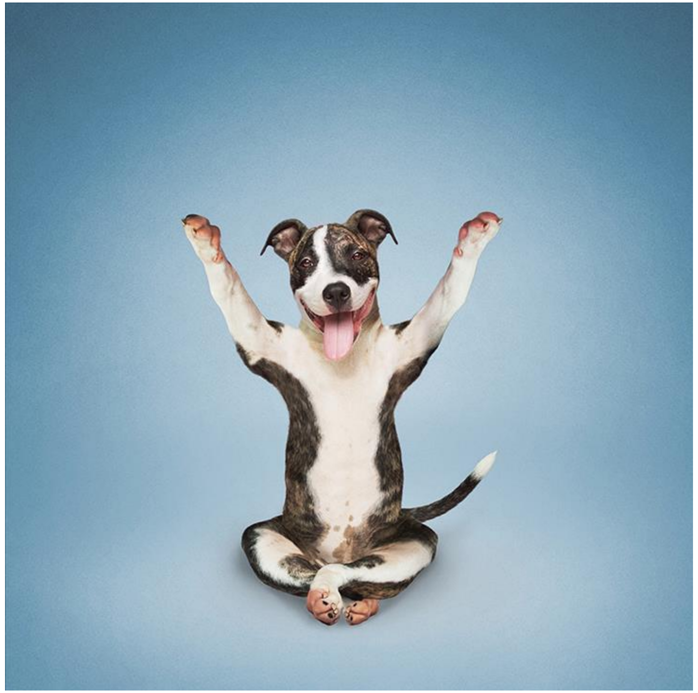
Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm? – Deal with Occlusion

Images Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm? – Invariant to Deformation



Images Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm? Deal with Background Clutter

Image Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm? Deal with Intra-class Variation

Image Source: Google

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm? No Way To Hand Code It!

- Image classification:

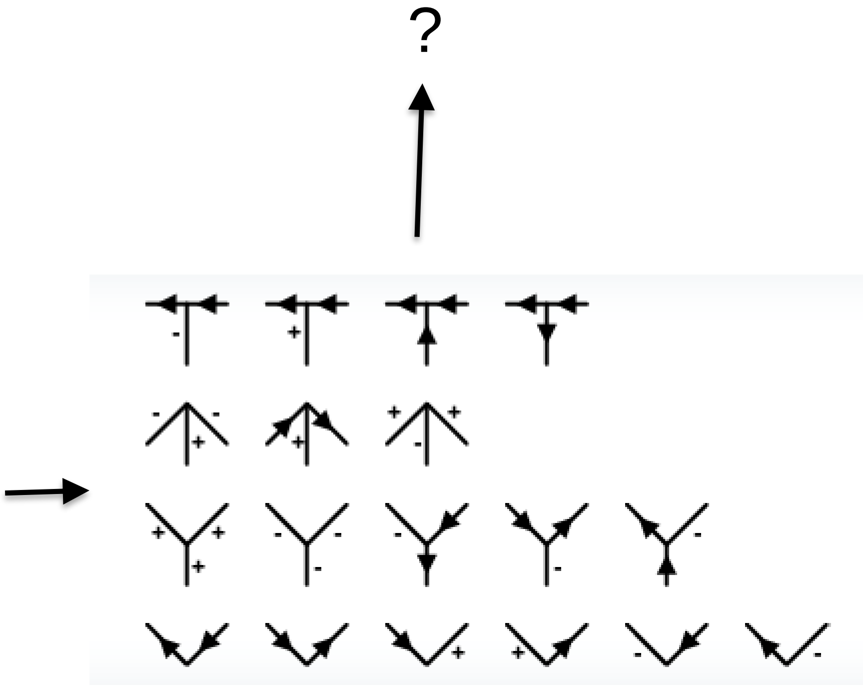
```
Def predict(image)
  -- ????
  return class_label
end
```

- Unlike e.g. sorting a list of numbers
- No obvious way to hard-code the algorithm for recognizing a cat, or other classes

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Why Data Driven Paradigm?

- Image classification:



Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

The Data Driven Paradigm

1. Prepare a dataset of labelled images
2. Use Machine Learning to train an image classifier
3. Then evaluate the performance of classifier using a withheld set of test images(these images shouldn't be used to train the classifier).

```
def train(train_images, train_labels)
  -- Build model: images -> labels
  return model
end


def predict(model, test_images)
  -- Predict test_labels using the model
  return test_labels
end
```

Example Training Set

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Agenda

- History and Evolution of Neural Networks
- The AI Winter and then the Tipping Point for Deep Learning
- Why Deep Learning
- The Data Driven Paradigm
- **Training a classifier (without any trainable parameters) – KNN**
- Training a classifier (with trainable parameters) – Linear Classifier
- How to split our data into training, validation and test sets

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Classifier 1: Nearest Neighbor Classifier

```
def train(train_images, train_labels)
  -- Build model: images -> labels
  return model
end
```

Train the model with labelled training data and remember all training images with their labels

```
def predict(model, test_images)
  -- Predict test_labels using the model
  return test_labels
end
```

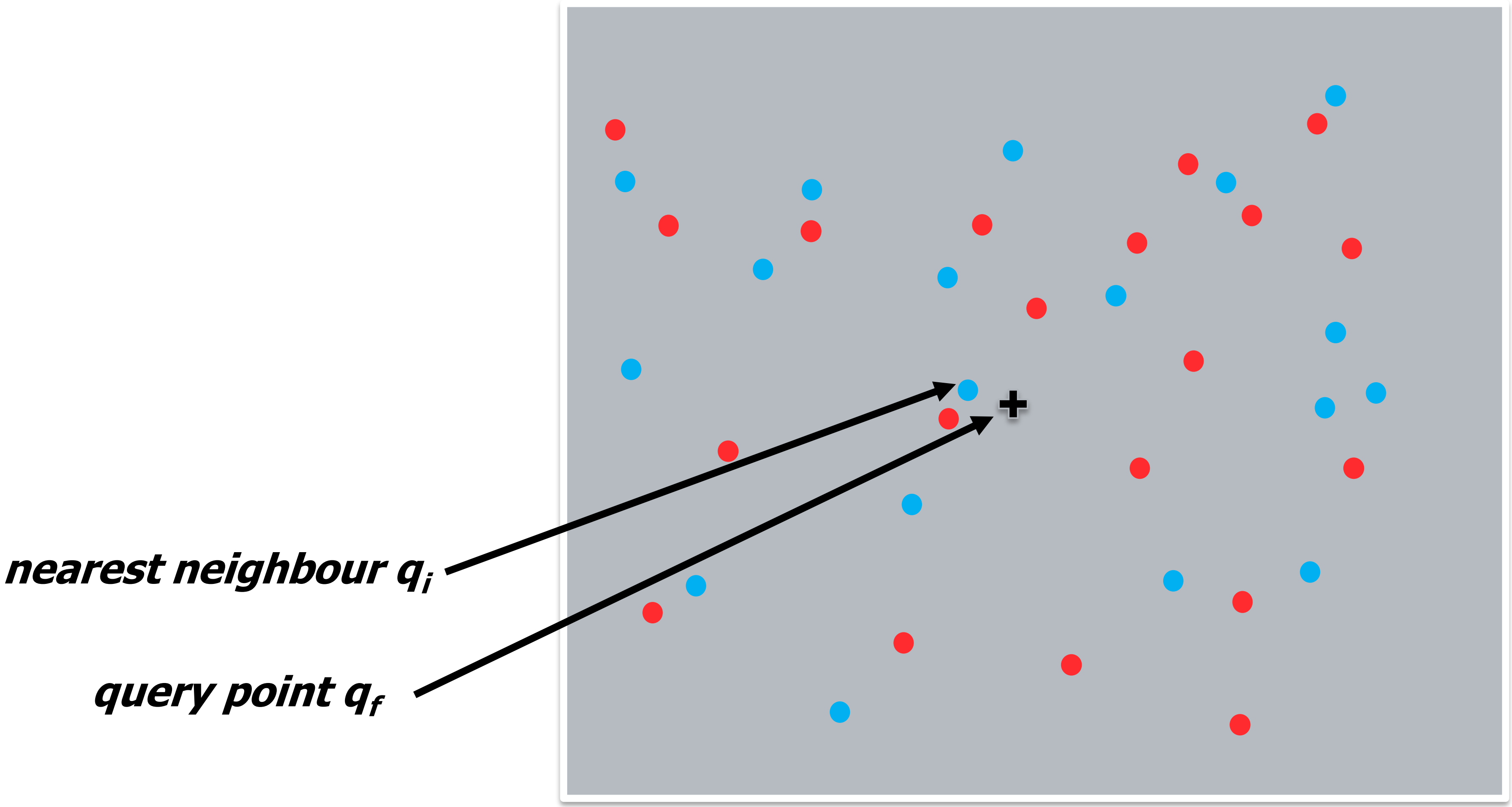
Predict the label of the most similar training image

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Train vs. Test

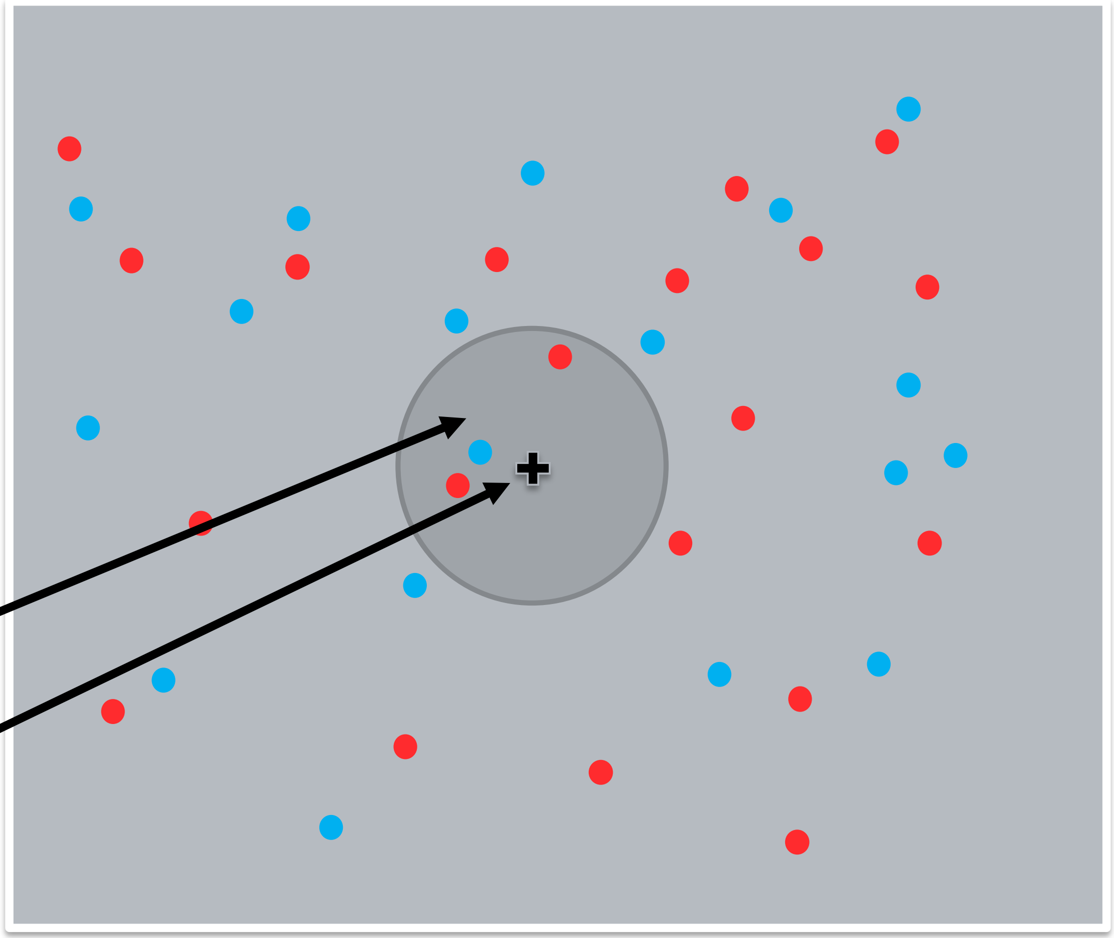
Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

1-Nearest Neighbour



Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

1-Nearest Neighbour

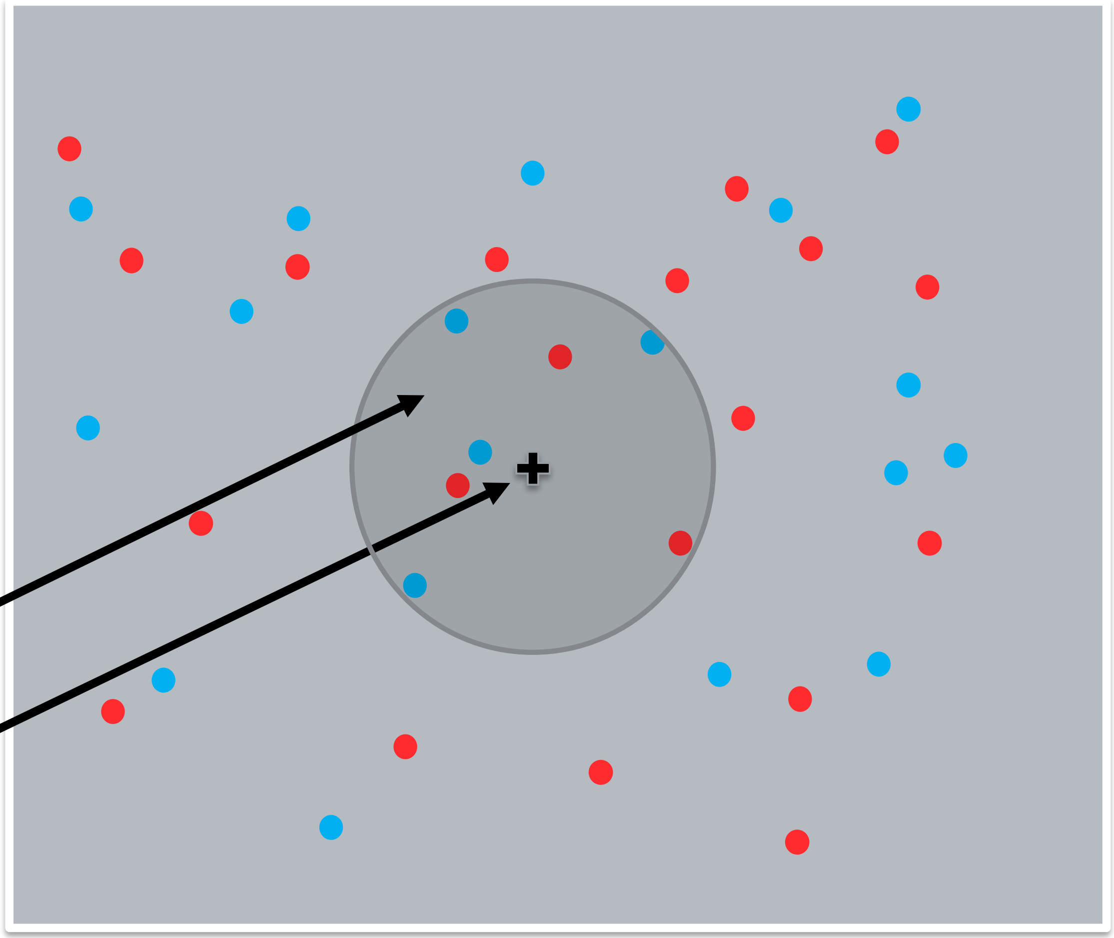


3 nearest neighbors,
2 red, 1 blue

query point q_f

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

1-Nearest Neighbour



3 nearest neighbors,
3 red, 4 blue

query point q_f

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Example Dataset: MNIST

0
1
2
3
4
5
6
7
8
9

10 labels
60,000 training images
10,000 test images.

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

How Do We Compare the Images? What is the Distance Metric?

test image

56	32	10	18
90	23	128	133
24	26	178	200
2	0	255	220

-

training image

10	20	24	17
8	10	89	100
12	16	178	170
4	32	233	112

=

pixel-wise absolute value differences

46	12	14	1
82	13	39	33
12	10	0	30
2	32	22	108

→ 456

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Nearest Neighbor Classifier

```python
class NN:
    def __init__(self):
        pass

    def train(self, X, y):
        # X is 2D if size N x D = 23x23x3, so each row is an example
        # y is 1D of size N
        self.tr_x = X
        self.tr_y = y

    def predict(self, x):
        # x is of size D = 32x32x3 for which we want to predict the label
        # returns the predicted label for the input x
        min_idx = None
        min_dist = 100000000
        for test_sample in range(len(self.tr_x)):
            dist = 0
            for each_value in range(len(self.tr_x[0])):
                dist += abs(float((self.tr_x[test_sample][each_value] - x[each_value])))
            if dist < min_dist:
                min_dist = dist
                min_idx = test_sample
        return self.tr_y[min_idx]
```

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Nearest Neighbor Classifier

remember the training data

```
class NN:
    def __init__(self):
        pass

    def train(self, X, y):
        # X is 2D if size N x D = 23x23x3, so each row is an example
        # y is 1D of size N
        self.tr_x = X
        self.tr_y = y

    def predict(self, x):
        # x is of size D = 32x32x3 for which we want to predict the label
        # returns the predicted label for the input x
        min_idx = None
        min_dist = 100000000
        for test_sample in range(len(self.tr_x)):
            dist = 0
            for each_value in range(len(self.tr_x[0])):
                dist += abs(float((self.tr_x[test_sample][each_value] - x[each_value])))
            if dist < min_dist:
                min_dist = dist
                min_idx = test_sample
        return self.tr_y[min_idx]
```

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Nearest Neighbor Classifier

```
class NN:
    def __init__(self):
        pass

    def train(self, X, y):
        # X is 2D if size N x D = 23x23x3, so each row is an example
        # y is 1D of size N
        self.tr_x = X
        self.tr_y = y

    def predict(self, x):
        # x is of size D = 32x32x3 for which we want to predict the label
        # returns the predicted label for the input x
        min_idx = None
        min_dist = 100000000
        for test_sample in range(len(self.tr_x)):
            dist = 0
            for each_value in range(len(self.tr_x[0])):
                dist += abs(float((self.tr_x[test_sample][each_value] - x[each_value])))
            if dist < min_dist:
                min_dist = dist
                min_idx = test_sample
        return self.tr_y[min_idx]
```

For the test image:

- find nearest train image with minimum distance from the test image
- predict the label of nearest training image

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Nearest Neighbor Classifier

```
In [ ]: # Changing dimensions to N x D = 28x28x3
        # Number of samples you want in training data. 60,000 is max.
        N = 5000
        x_tr = []
        for i in range(N):
            x_tr.append(x_train[i,:].flatten())
        x_te = []
        for i in range(10000):
            x_te.append(x_test[i,:].flatten())
        print(len(x_tr),len(x_tr[0]))
        print(len(x_te),len(x_te[0]))


        class NN:
            def __init__(self):
                pass

            def train(self, X, y):
                # X is 2D if size N x D = 23x23x3, so each row is an example
                # y is 1D of size N
                self.tr_x = X
                self.tr_y = y

            def predict(self, x):
                # x is of size D = 32x32x3 for which we want to predict the label
                # returns the predicted label for the input x
                min_idx = None
                min_dist = 100000000
                for test_sample in range(len(self.tr_x)):
                    dist = 0
                    for each_value in range(len(self.tr_x[0])):
```

Q: how does the classification speed depend on the size of the training data?

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Nearest Neighbor Classifier

```
In [ ]: # Changing dimensions to N x D = 28x28x3
        # Number of samples you want in training data. 60,000 is max.
        N = 5000
        x_tr = []
        for i in range(N):
            x_tr.append(x_train[i,:].flatten())
        x_te = []
        for i in range(10000):
            x_te.append(x_test[i,:].flatten())
        print(len(x_tr),len(x_tr[0]))
        print(len(x_te),len(x_te[0]))


        class NN:
            def __init__(self):
                pass

            def train(self, X, y):
                # X is 2D if size N x D = 23x23x3, so each row is an example
                # y is 1D of size N
                self.tr_x = X
                self.tr_y = y

            def predict(self, x):
                # x is of size D = 32x32x3 for which we want to predict the label
                # returns the predicted label for the input x
                min_idx = None
                min_dist = 100000000
                for test_sample in range(len(self.tr_x)):
                    dist = 0
                    for each_value in range(len(self.tr_x[0])):
```

Q: how does the classification speed depend on the size of the training data?

A: Linearly!

This is **backwards**:

- test time performance is usually much more important in practice.
- CNNs flip this: expensive training, cheap test evaluation

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

The Choice of Distance is a **Hyperparameter**

L1 distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

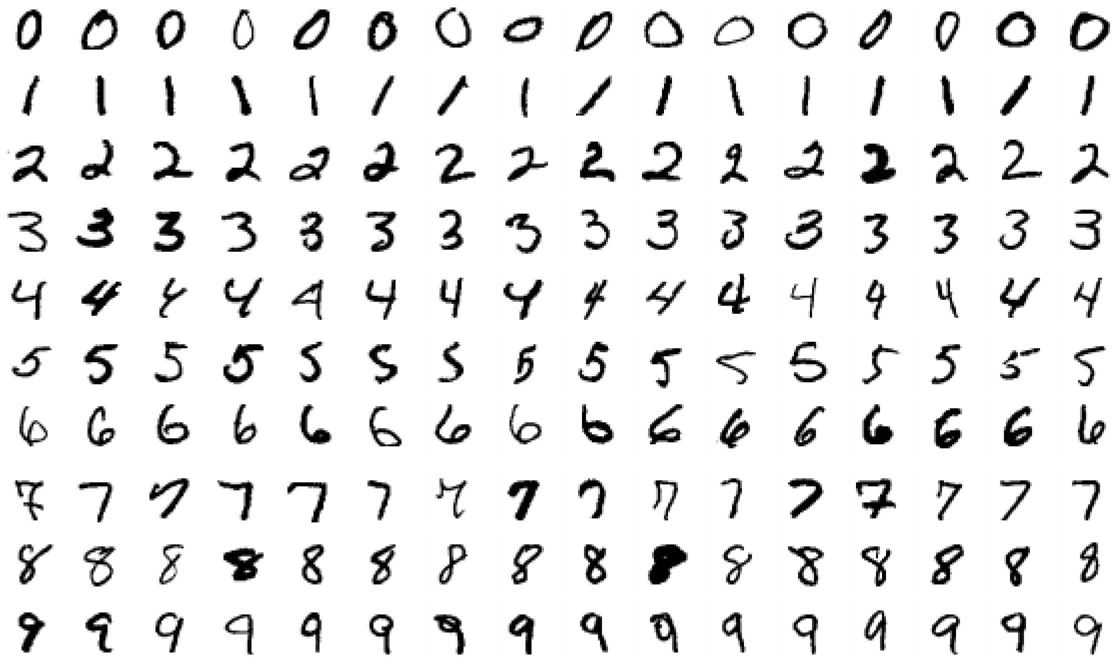
L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p \left(I_1^p - I_2^p\right)^2}$$

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Example Dataset: MNIST

0
1
2
3
4
5
6
7
8
9



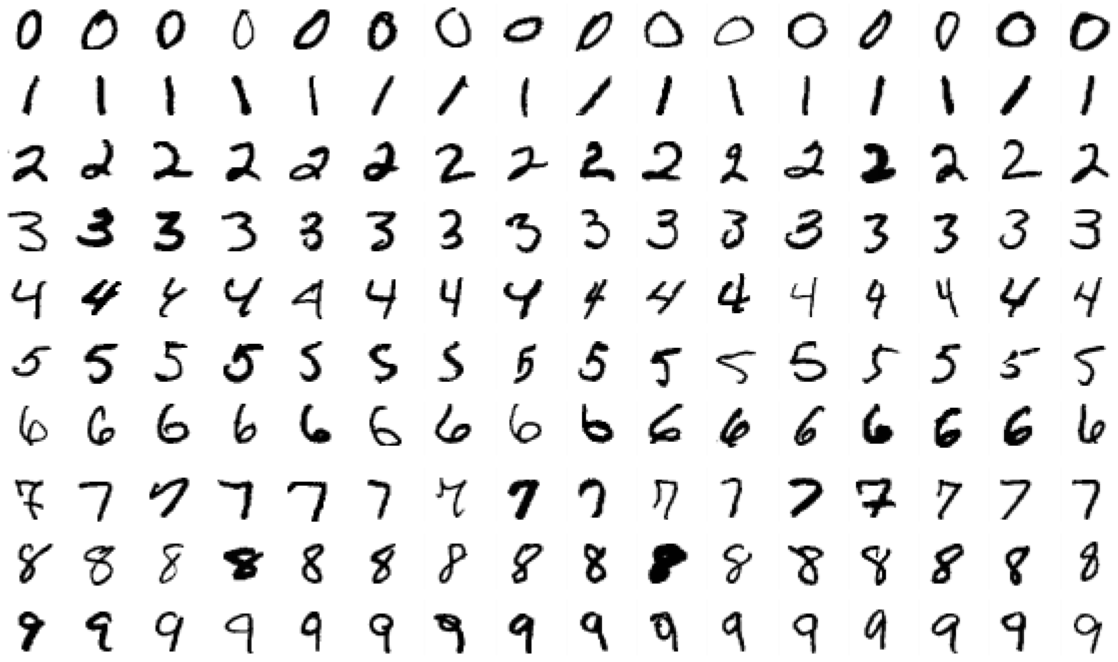
10 labels
60,000 training images
10,000 test images.

Q1: What is the accuracy of the nearest neighbor classifier on the **test** data, when using the Euclidean distance? What about L1 distance?

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Example Dataset: MNIST

0
1
2
3
4
5
6
7
8
9



10 labels
60,000 training images
10,000 test images.

Q2: What is the accuracy of the **k**-nearest neighbor classifier on the **test** data? What is the best value of **k**?

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

How Do We Set the **Hyperparameters**?

Best Distance to use?

What should be the best value of k to use? Is it random?

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

How Do We Set the **Hyperparameters**?

Best Distance to use?

What should be the best value of k to use? Is it random?

This is a very problem dependent.

One should try various K and see what works best with the problem.

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Try Out What Hyperparameters Work Best on **Test Set?**

Validation Data
(used to tune hyperparameters)

Bad idea to use test data to tune hyperparameters

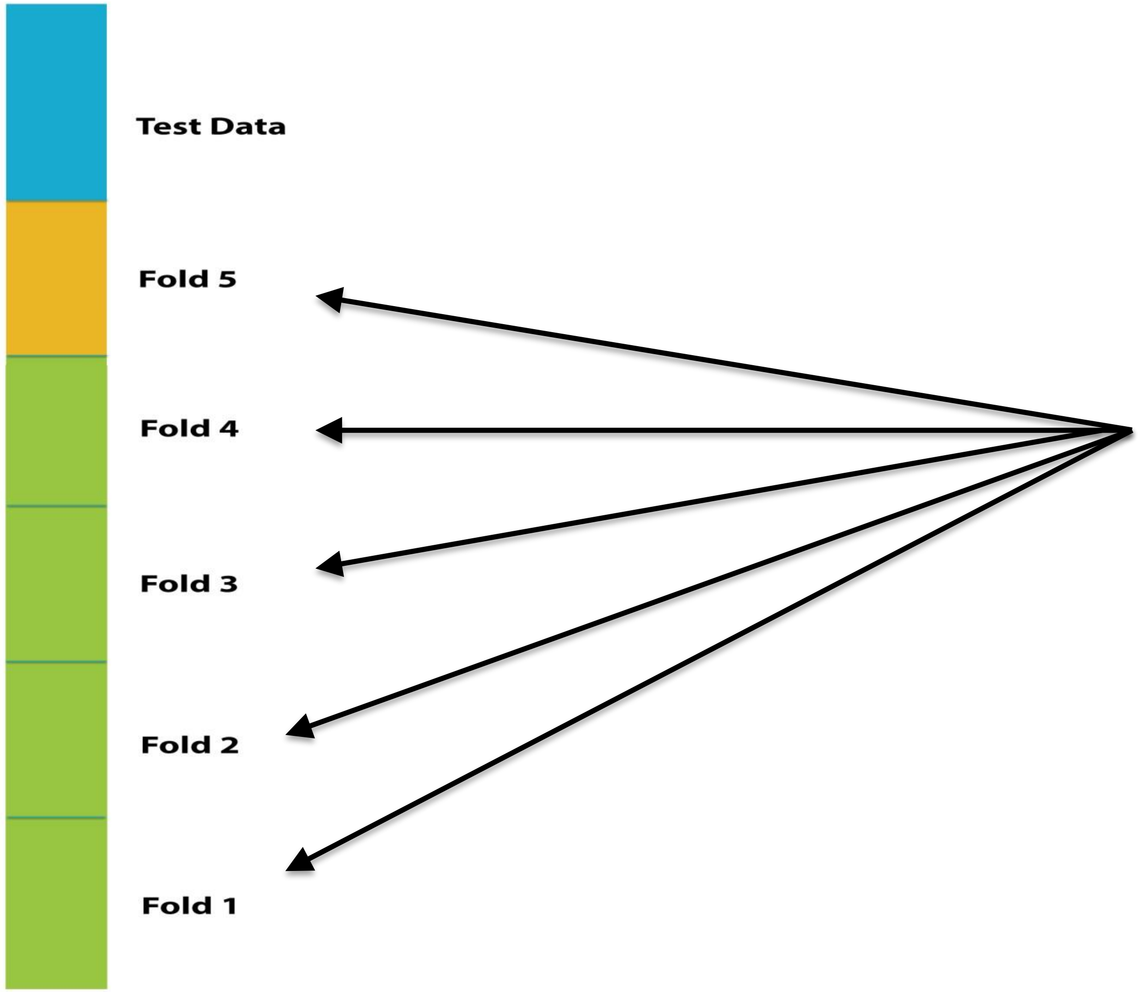
- *unless trying to win a competition where test set is given*

The test set is a proxy for the generalization performance!

- Use only **VERY SPARINGLY,** at the end.

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Try Out What Hyperparameters Work Best on **Test Set?**

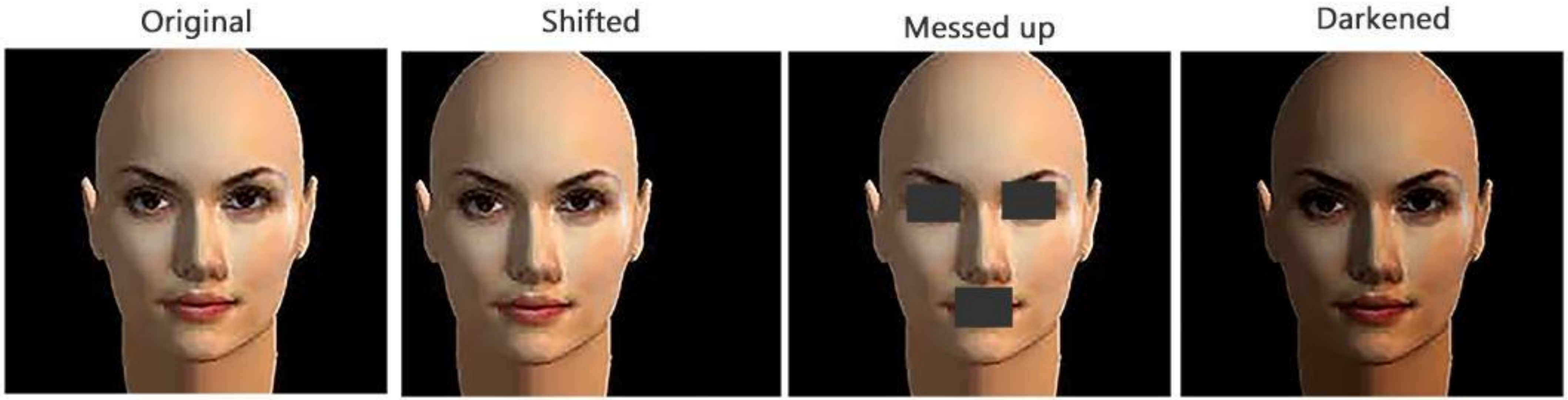


Cross-validation
cycle through the choice of which fold is the validation fold, average results.

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

K-Nearest Neighbor on Images Never Used

- Terrible performance at test time
- Distance metrics on level of whole images can be very unintuitive



(all 3 images have same L2 distance to the one on the left)

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

K-Nearest Neighbor Summary

- **Image Classification:** We are provided with given **Training Data set** and labeled images, and we are asked to predict labels on **Test Data Set**. Common to report the **Accuracy** of predictions (fraction of correctly predicted images)
- We introduced the **k-Nearest Neighbor Classifier**, whose prediction is based on the labels of the nearest images in the **Training Data Set**
- If the size of data set is small the choice of distance and value of hyper parameters are tuned using a Validation Data Set or cross validation method.
- Once we choose the best set of hyper parameters, the classifier model is evaluated using **Test Data Set** and its performance on this data set is reported as the KNN classifier performance on that data.

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

K-Nearest Neighbor

- k-NN classifier:
  - Need to remember all of the training data and store it for future comparisons with the test data or unseen data
  - This can be very Space consuming since datasets may easily be Gigabytes in size
  - Image classification using k-NN can be very expensive as it needs to compare test image with all the training images

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

K-Nearest Neighbor Algorithm

When to consider

- Data a vector $\in \mathbb{R}^d$
- Lots of Training Data

Advantages

- Training is very fast
- Learn complex target functions
- Do not lose information

Disadvantages

- Slow at query time
  - Presorting and indexing training samples into search trees can reduce query time
- Easily fooled by irrelevant attributes

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Agenda

- History and Evolution of Neural Networks
- The AI Winter and then the Tipping Point for Deep Learning
- Why Deep Learning
- The Data Driven Paradigm
- Training a classifier (without any trainable parameters) – KNN
- **Training a classifier (with trainable parameters) – Linear Classifier**
- How to split our data into training, validation and test sets

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Parametric Approach: MNIST

10 labels
60,000 training images
10,000 test images
each image is an array of size **28 x 28 = 784** numbers total

0
1
2
3
4
5
6
7
8
9

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Parametric Approach: Linear Classifier

greatlearning
Learning for Life

Image

[28x28]
array of numbers
0...255
(784 numbers total)

10x1 **10x784** **784x1**

$$f(x, W) = Wx \quad (+b)$$

10x1

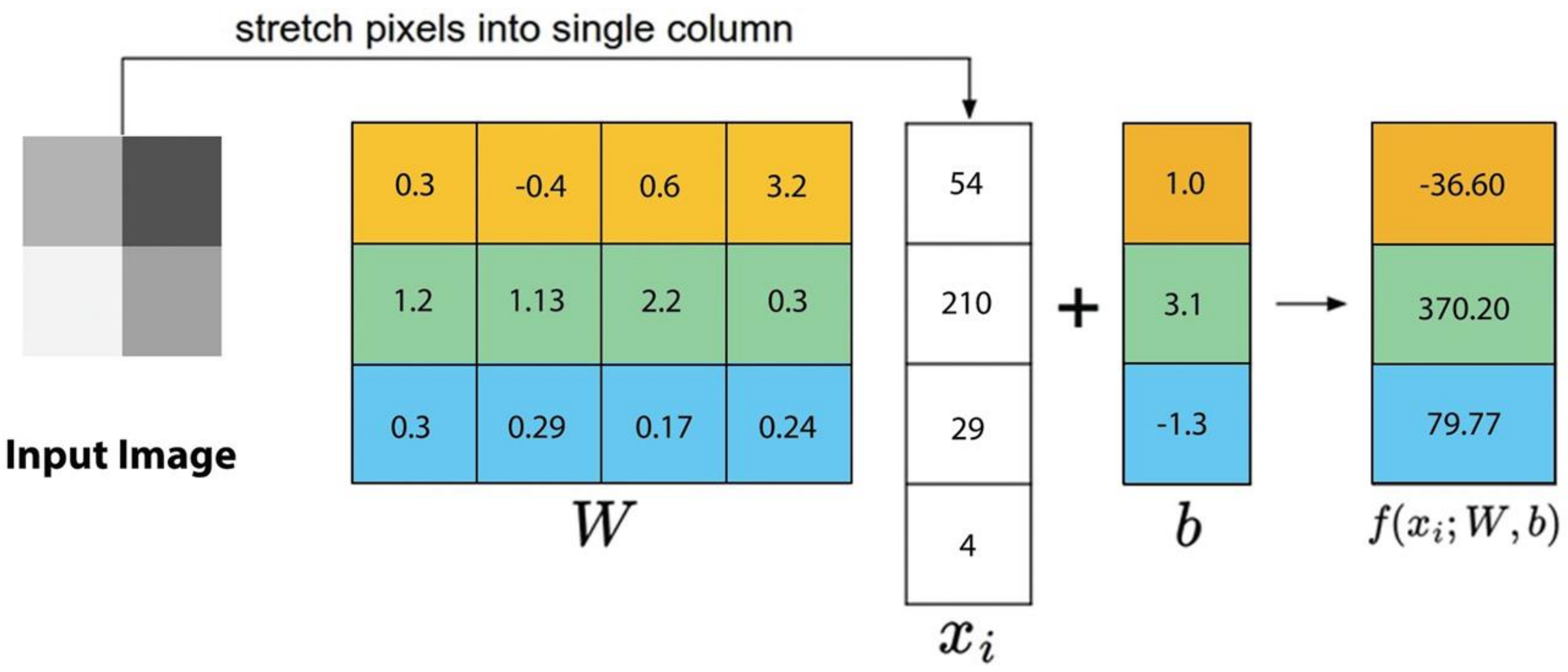
10 numbers, indicating class scores

Input image

weights or parameters

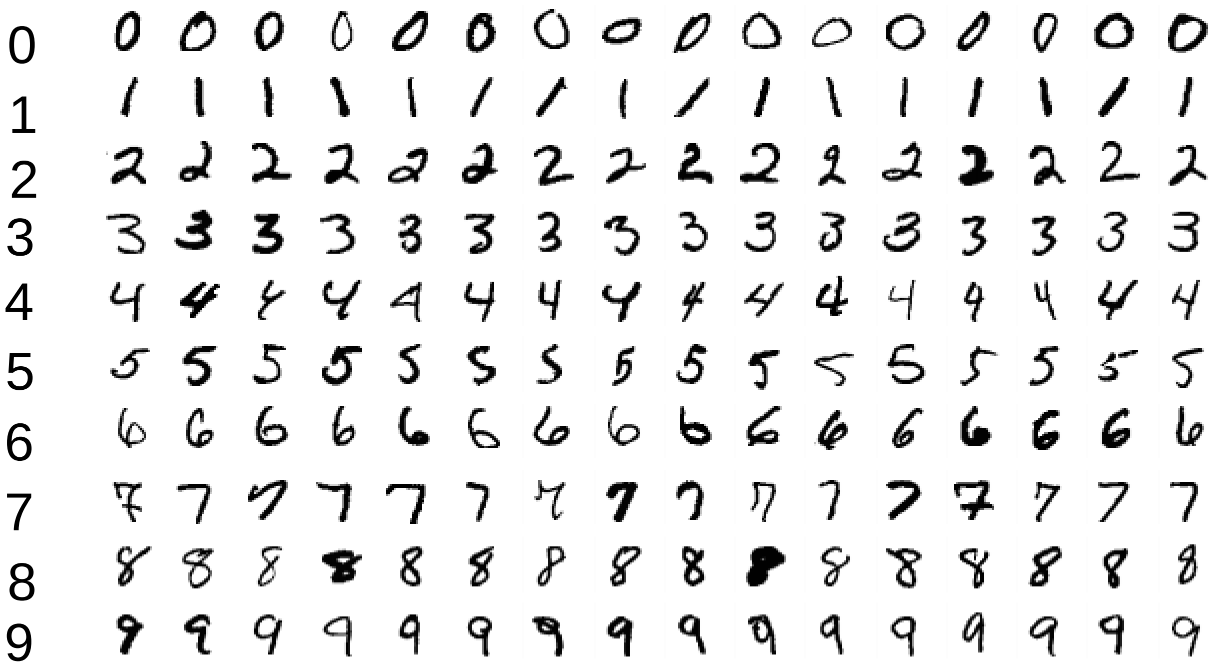
Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Example with an Image with 4 Pixels, and 3 Classes (1/2/3)



Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

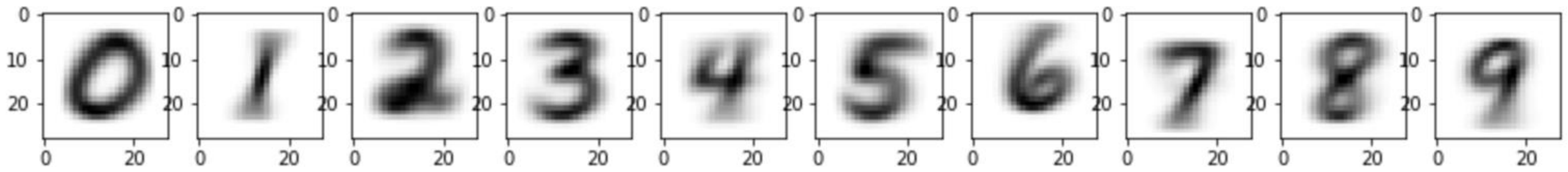
Interpreting a Linear Classifier



10x784

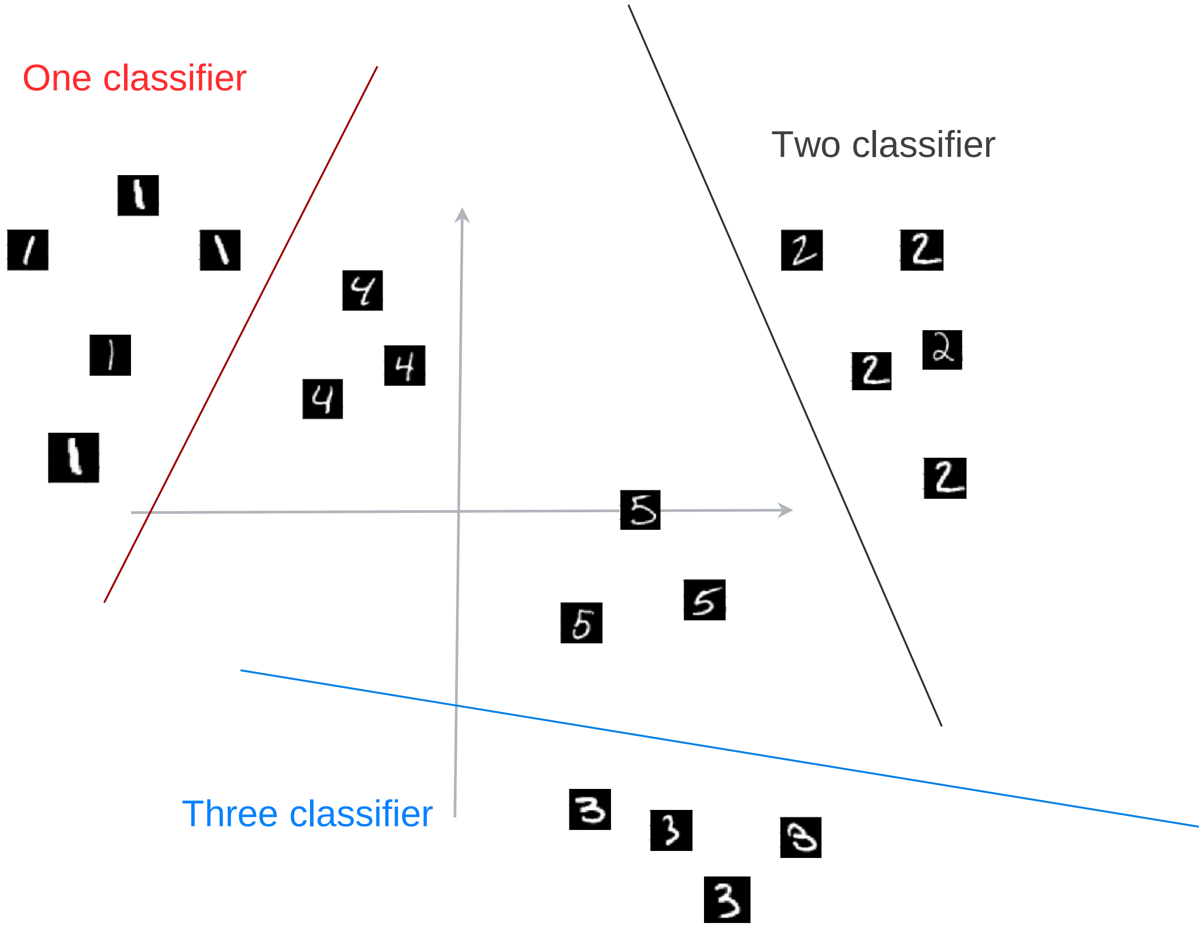
$$f(x_i, \boxed{W}, b) = Wx_i + b$$

Example trained weights of a linear classifier trained on MNIST:



Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Interpreting a Linear Classifier



$$f(x_i, W, b) = Wx_i + b$$

[28x28]
array of numbers 0…255
(784 numbers total)

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Summary

- Evolution of NN
- Why data-driven
- Simple classifier like k-NN and its limitations
- Parametric approach and its benefits
- What next - what does a parametric approach lead us to?

Proprietary content. ©Great Learning. All Rights Reserved. Unauthorized use or distribution prohibited

Thank you!