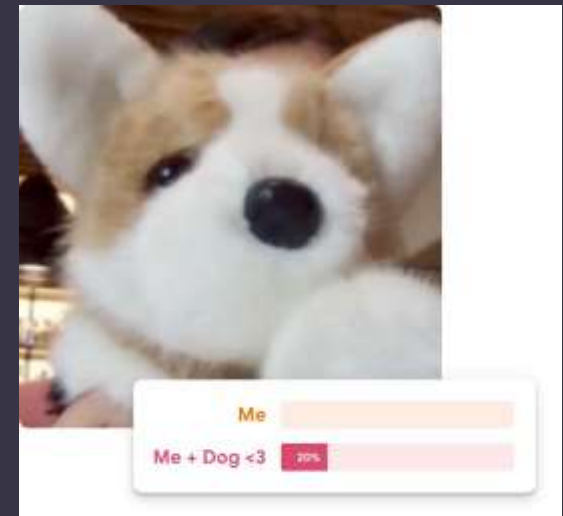


# Teachable machine

**PRESENTER : YIN JEH NGUI, JASON**

**NATIONAL YANG MING CHIAO TUNG UNIVERSITY**

**10TH NOV 2021**



# Roadmap

---

AI, ML, DL?

How have AI evolved?

Teachable machine

- Intro
- Collect data
- Train models
- Infer from image
- Export model

Prospects

- Applications

# Background of AI

Back in 1958, Frank Rosenblatt at Cornell design the first artificial neural network

- Described presciently as “**Pattern-recognizing device**”
- Era of mainframe computers filled rooms and ran on vacuum tubes
- Inspired by the interconnections between neurons in the brain

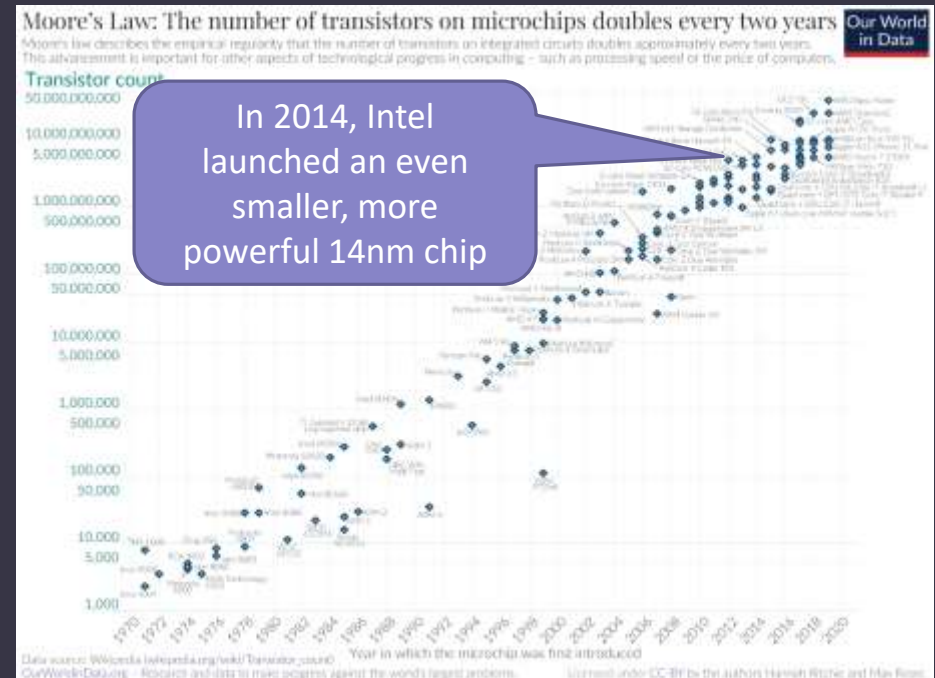
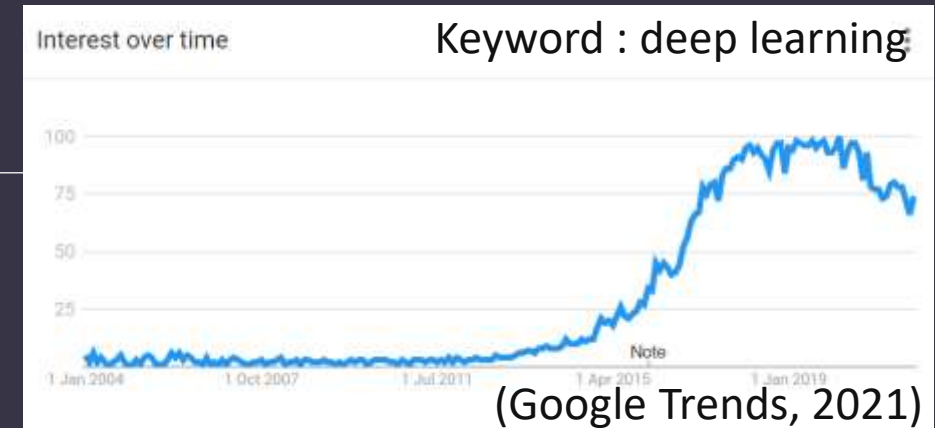
Limitation of computing hardware soon overcame

- **Moore's Law** and other improvements in hardware
- Yielded a roughly 10-million-fold increase in the number of computations that a computer could do in a second
- Inclusion GPU in computation

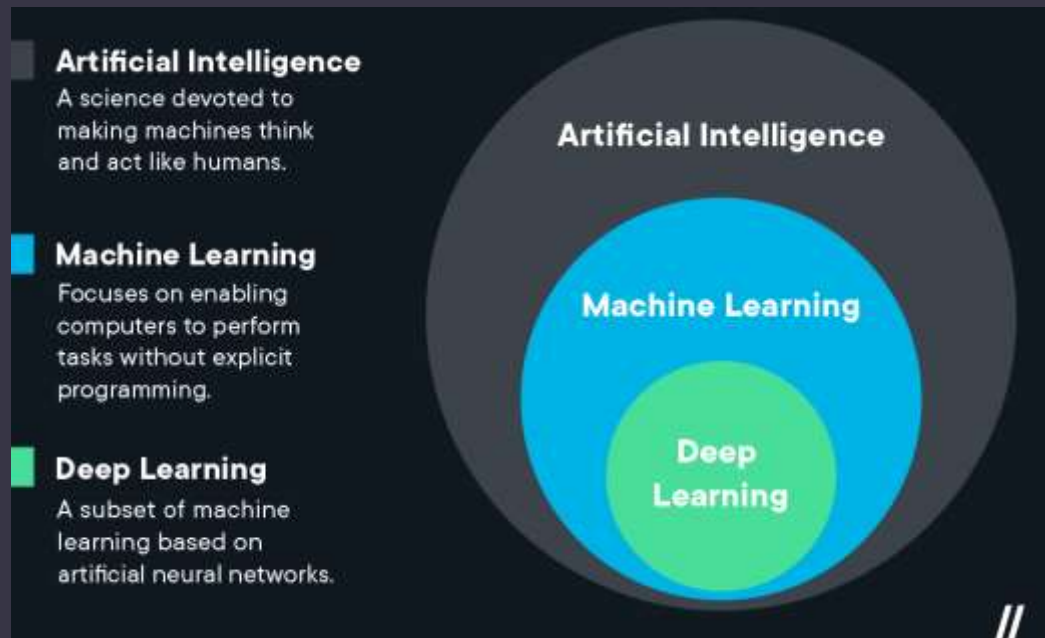
Interest in artificial intelligence (AI) is revisited in the late 2000s

- Tools available up to the computing challenge
- Renamed as “**deep learning**”
- Extra layers of neurons is introduced

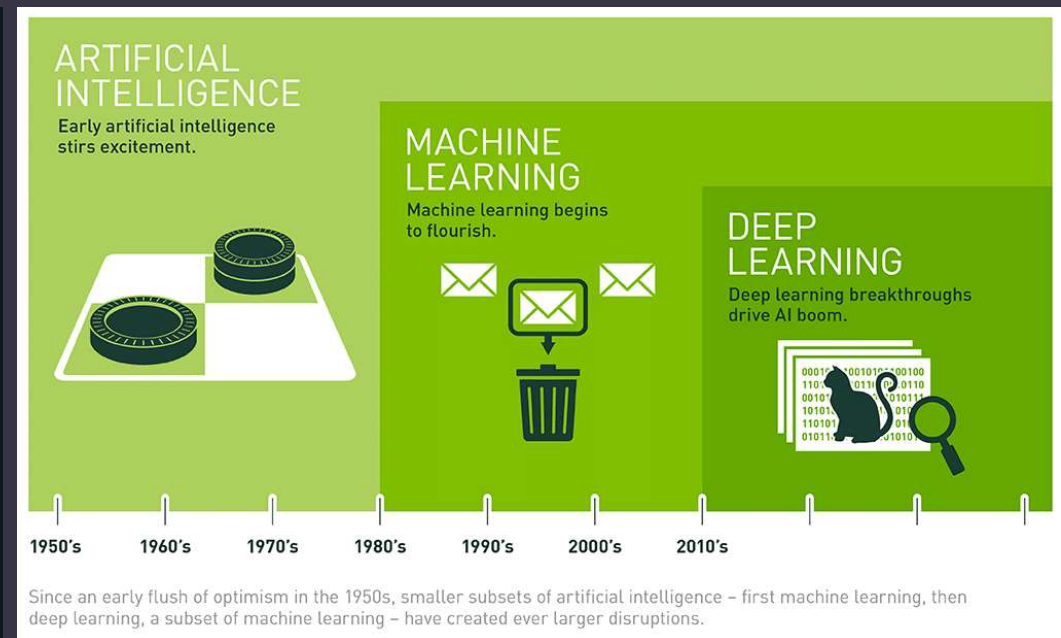
**Observation** that the transistors amount in a dense integrated circuit (IC) doubles about every two years



# AI, ML, DL?



(Middleton, 2021)



(Copeland, 2016)

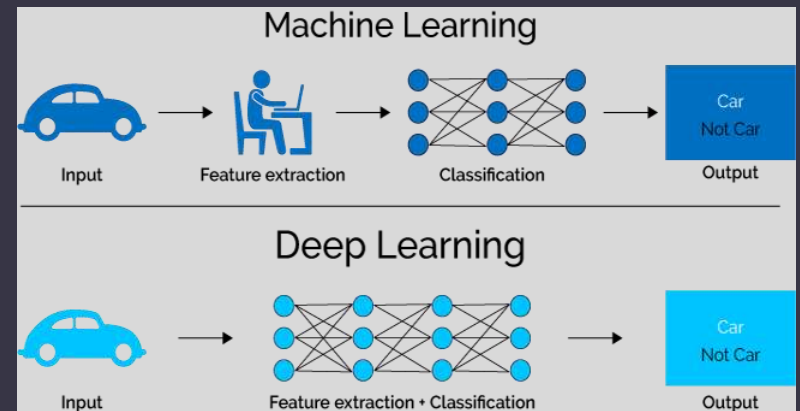
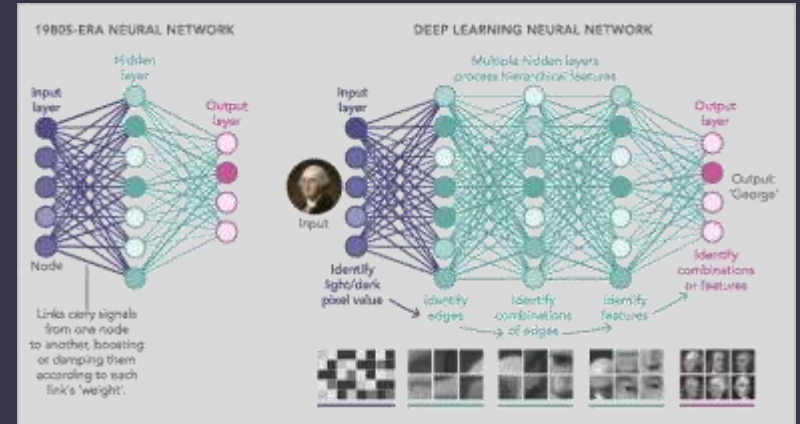
# ML / DL

## Expert-knowledge based model (ML)

- Modelling based on professional know-how
- (Geotechnical) modelling with governing equations
- Early AI were rule based, applying logic and expert knowledge to derive results (machine learning)
  - Supervised / semi-supervised / unsupervised
  - Reinforcement learning

## Flexible statistical models (most AI/DL)

- Require myriad combinations of priori, activation functions, bias, weighting and networks
- Use numerous neurons (in **neural networks**) to train suitable AI model
- Provide outcomes with probability [87% cat]



(Mahapatra, 2019)

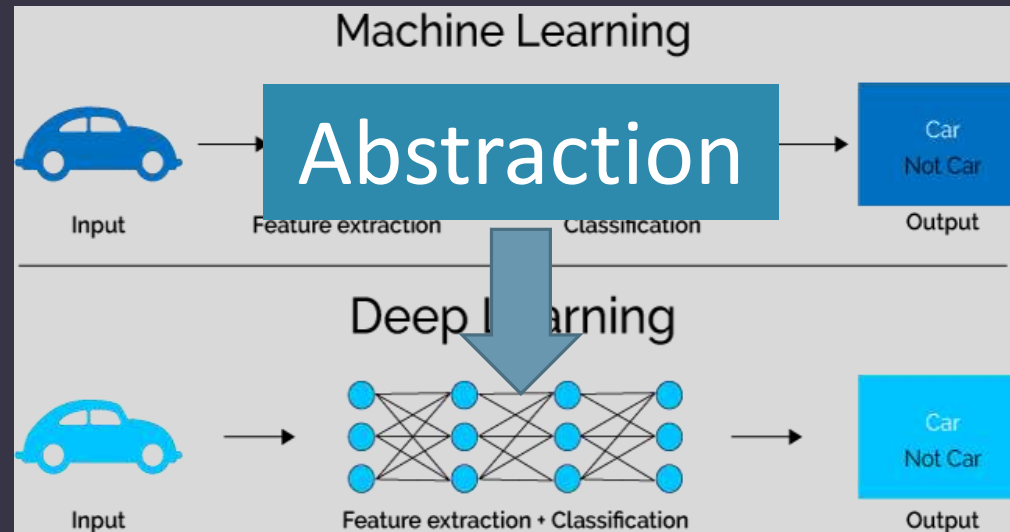
# From ML to DL

## Machine learning

- Computers still think and act like machines
- Needs **manual** feature extraction

## Deep learning

- Computers learn to think using structures modeled from human brain
- Multi-layered “deep neural networks”
- Non-linear data transformation => increasingly abstract



# ML vs DL

	Machine learning	Deep learning (NN)
<b>Hardware</b>	Less complex Run on conventional computers	Powerful hardware GPU for parallel computing
<b>Time</b>	Quick set up and operation Limited in performance	More setup time Instantaneous result, improved quality over time with more data
<b>Approach</b>	Structured data Traditional algorithms (e.g. linear regression)	large volumes of unstructured data Neural networks
<b>Applications</b>	Already in use in your email inbox, bank, and doctor's office	Complex and autonomous programs (e.g. self-driving cars / robots that perform advanced surgery)

# Deep learning

## Feedforward neural network

- Simplest neural network
- Maybe with single/multi-layer neurons + backpropagation (multi-layer perceptron)
- Input layer => hidden layer => output layer

## Convolutional Neural Networks (CNN)

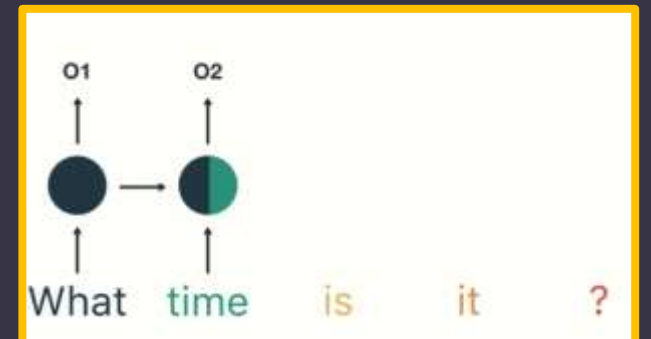
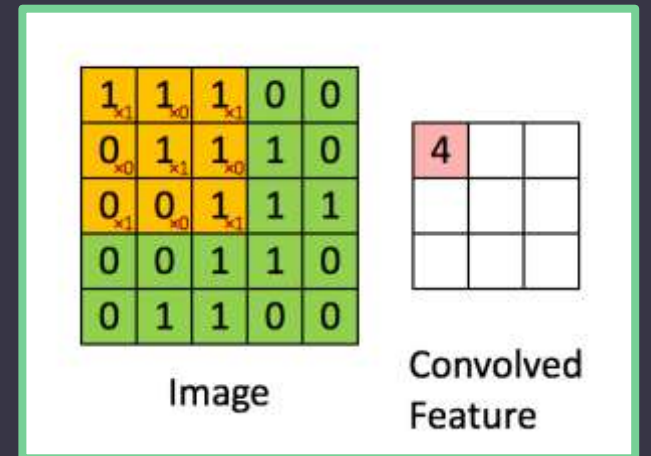
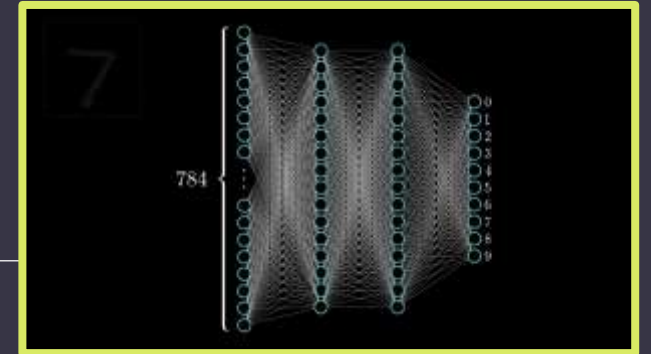
- Convolution : process of applying weight-based filter across every element of an image
  - Assist in computer vision, helping computer to “see” and “understand” what is in the picture
- Designed to work with images
- Good for feature scanning in image
  - Identify people, cars, shipwreck at ocean floor, landslide
- Highest growth over the decade

## Recurrent Neural Networks (RNN)

- Introduce **MEMORY** in algorithm
  - Computer remembers past data and decisions, include in consideration of current data
- Natural language processing : considers tone and content in text comprehension
  - Personality chat robot : **NLP** (Natural language processing) + **LSTM** (Long Short-term Memory)
  - Personal assistance (Hey Siri , OK Google...)
- Map direction : remembers best route by majority of people

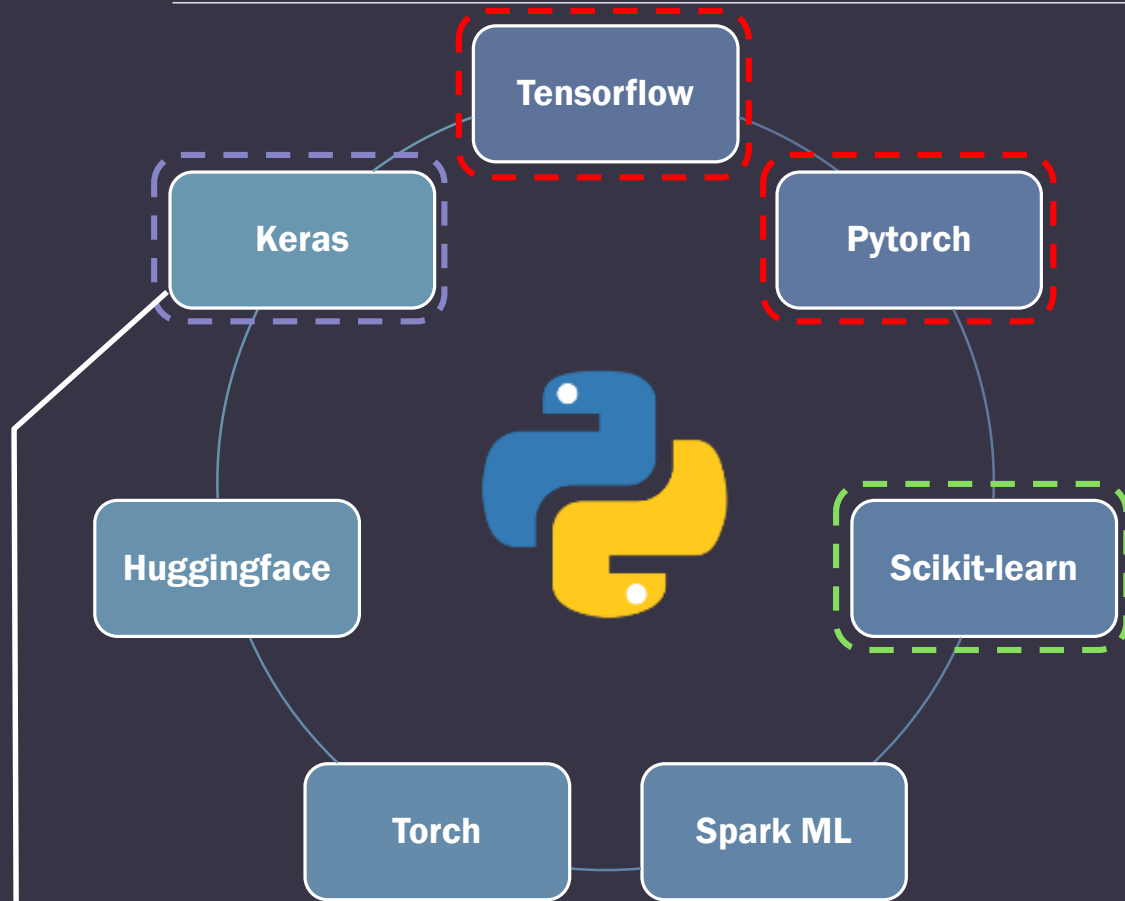
More to come...

- Combinations of multiple RNN/CNN





# Main ML/DL frameworks



Neural network library built on top of **TensorFlow**

Machine learning relies on algorithms

- ML/DL frameworks simplify this process
- tool, interface, or library for easy development of AI model
- without understanding the underlying algorithms

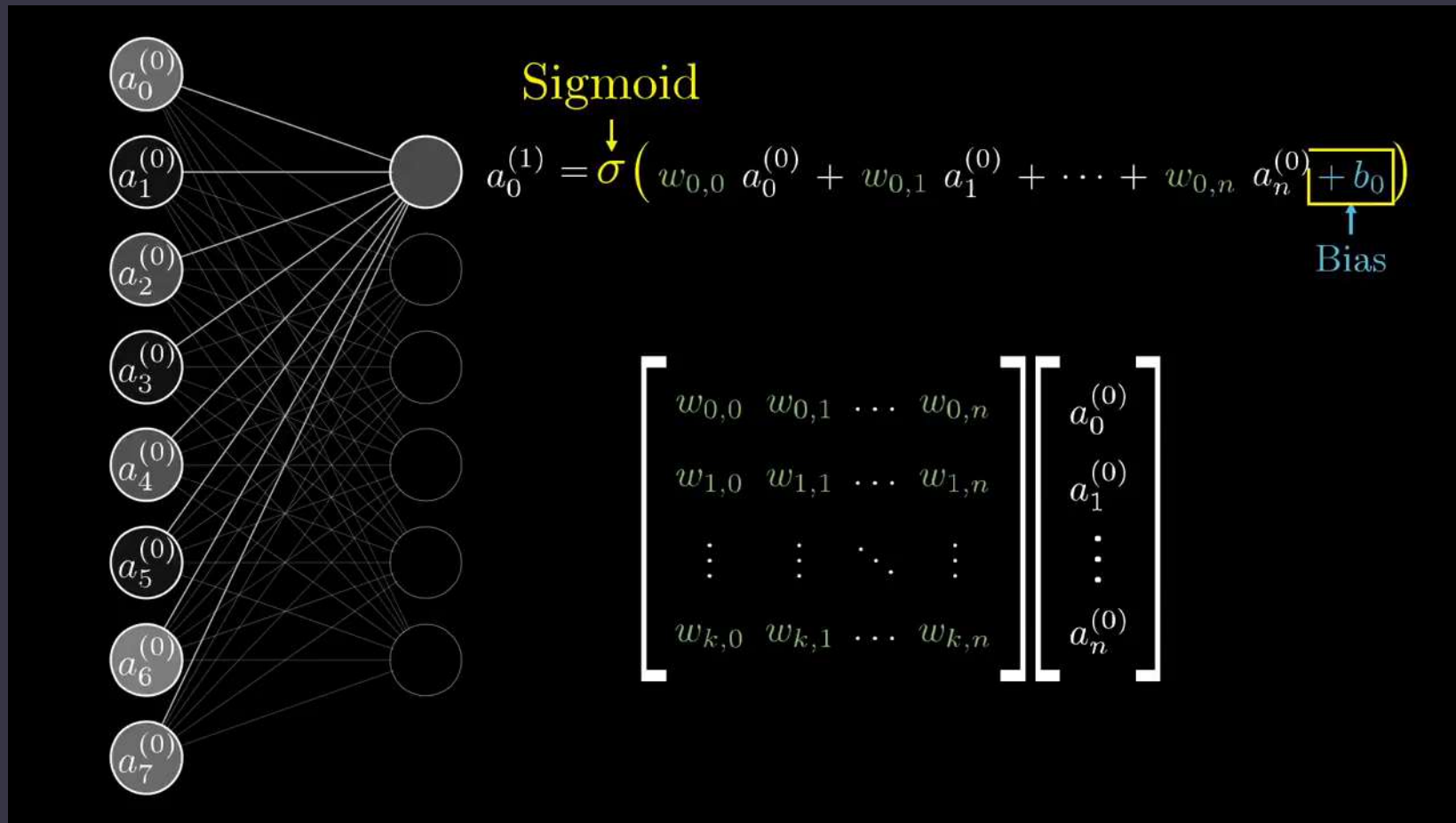
**Python** is the predominant ML programming language

- Do we use Tensorflow or Pytorch?
  - Tensorflow : Google Brain
  - Pytorch : Facebook AI Research (FAIR)
  - scikit-Learn : Old standards of the data science world, great ML tool for quick interpretation
- All are interchangeable

**Don't reinvent the wheel**

- Unless you want to change fundamentally

# But what is a neural network?



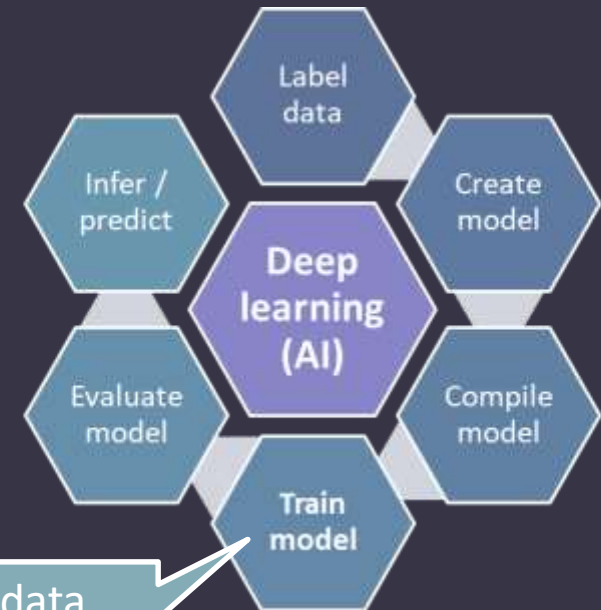
# AI too complicated?

---

- **START SIMPLE**

# Teachable machine

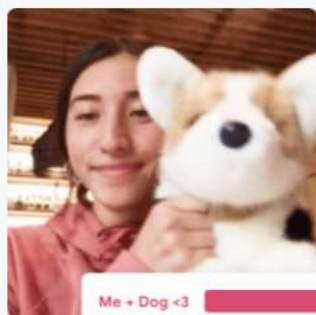
- GOOGLE INC.
- BRIDGE FROM ML TO DL
- TEACH MACHINES TO THINK (DL)



Using shuffled data  
(CPU/GPU exhaustive)

## What can I use to teach it?

Teachable Machine is flexible – use files or capture examples live. It's respectful of the way you work. You can even choose to use it entirely on-device, without any webcam or microphone data leaving your computer.



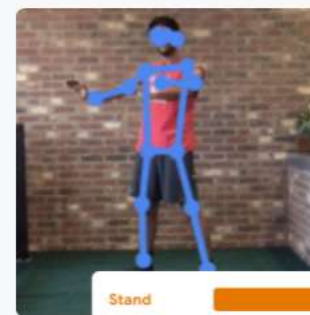
### Images

Teach a model to classify images using files or your webcam.



### Sounds

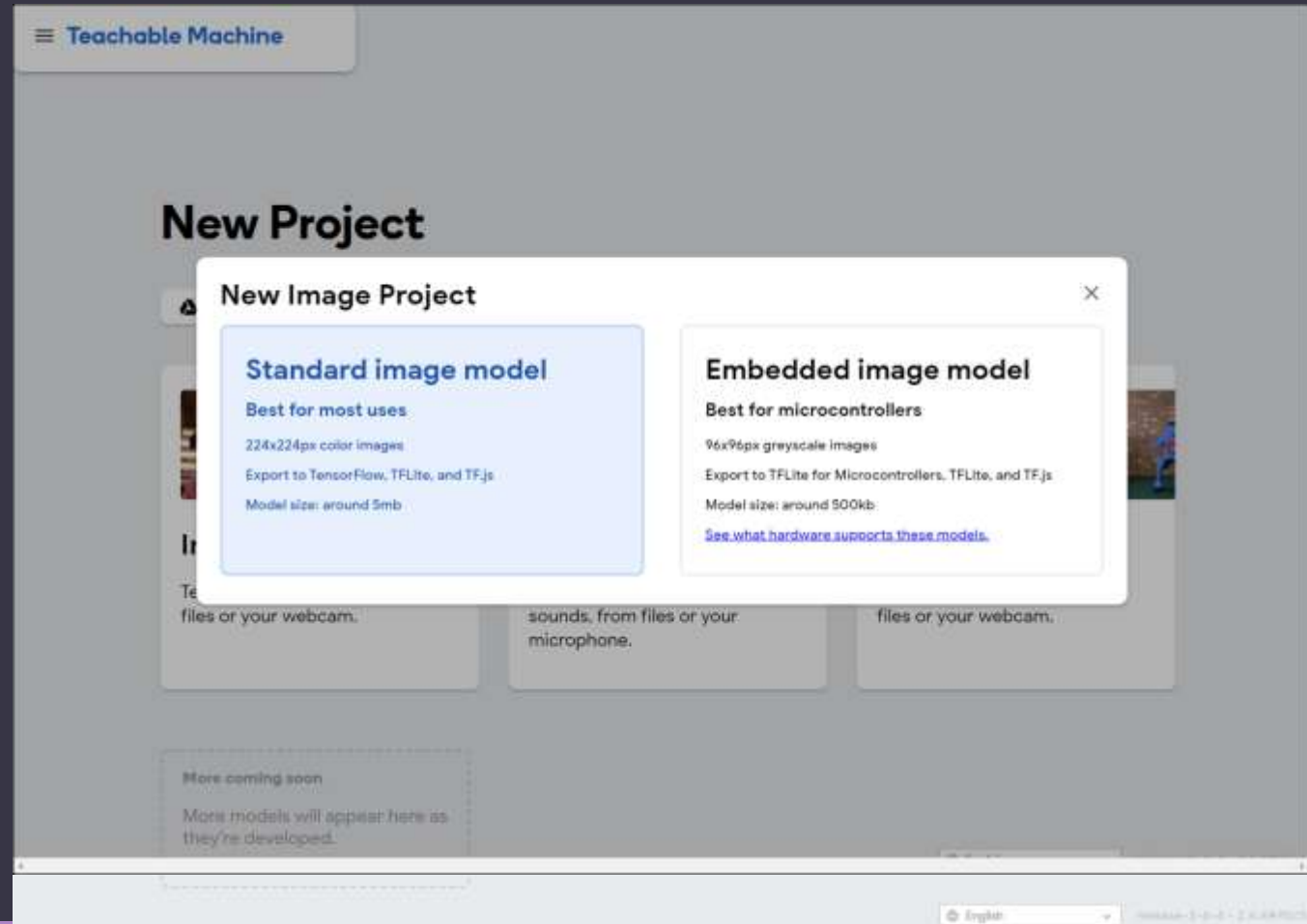
Teach a model to classify audio by recording short sound samples.



### Poses

Teach a model to classify body positions using files or striking poses in your webcam.

# Create a new project



Gather

Train

Infer

Teachable Machine

blank

29 Image Samples



Webcam



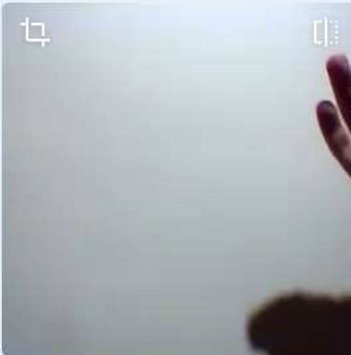
Upload



paper

Webcam

USB Camera (0c45:62c0)



Hold to Record

Add Image Samples:

Training

Train Model

Advanced

Preview



Export Model

You must train a model on the left before you can preview it here.

+ Add a class

English


release-2-4-4-2.4.4#95c54c

Gather









Train


Infer

Teachable Machine









blank 


29 Image Samples

 Webcam  Upload      









paper 


188 Image Samples


 Webcam  Upload      


scissor 

193 Image Samples




 Webcam  Upload      

rock 

Webcam 

USB Camera (0c45:62c0) 



222 Image Samples

Training

Train Model

Advanced 

Epochs: 50  

Batch Size: 16  

Learning Rate:  
0.001  

Reset Defaults 

Under the hood 

Preview

 Export Model

You must train a model on the left before you can preview it here.

 English 

release-2-4-4 - 2.4.4#95c54c




# Gather









# Train


# Infer

Teachable Machine









blank 


29 Image Samples

 Webcam  Upload      









paper 


188 Image Samples

 Webcam  Upload      









scissor 

193 Image Samples

 Webcam  Upload      

rock 

222 Image Samples


 Webcam  Upload      

## Training

Model Trained

Advanced 


Epochs: 50 

Batch Size: 16 

Learning Rate:

0.001 

Reset Defaults 

Under the hood 

## Preview

 Export Model


Input  ON Webcam 

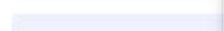
Switch Webcam 




## Output

blank 

paper 

scissor 

rock 

## Vocab

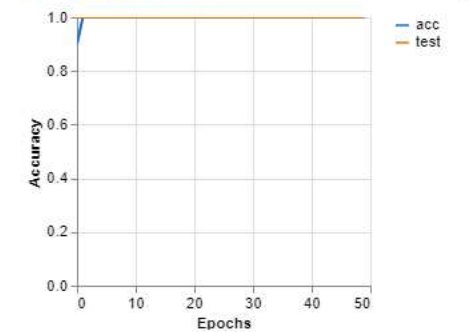
### Accuracy per class

Calculate accuracy per class

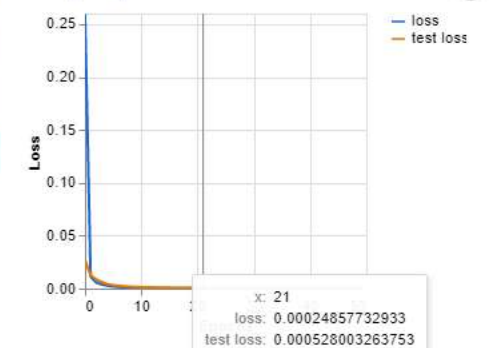
### Confusion Matrix

Calculate confusion matrix

### Accuracy per epoch



### Loss per epoch



Gather

Train

Infer

# Image inference/prediction

The screenshot displays the Teachable Machine web interface during the inference phase. On the left, four data classes are listed: 'blank' (29 Image Samples), 'paper' (188 Image Samples), 'scissor' (193 Image Samples), and 'rock' (222 Image Samples). Each class has a 'Webcam' button and a row of sample images. In the center, the 'Training' panel shows 'Model Trained' and advanced settings: Epochs: 50, Batch Size: 16, and Learning Rate: 0.001. On the right, the 'Preview' panel features a 'Webcam' input toggle, a live video feed, and an 'Output' section with four color-coded bars: 'blank' (orange, 100%), 'paper' (pink), 'scissor' (purple), and 'rock' (blue). The interface is in English and version 2.4.4.

Gather

Train

Infer

# Slanted image detection

The screenshot displays the Teachable Machine web interface. On the left, four data collection panels are visible: 'blank' (29 samples), 'paper' (188 samples), 'scissor' (193 samples), and 'rock' (222 samples). Each panel includes 'Webcam' and 'Upload' buttons and a row of image thumbnails. The 'paper' and 'scissor' panels show slanted hand gestures. In the center, the 'Training' panel shows 'Model Trained' and advanced settings: Epochs (50), Batch Size (16), and Learning Rate (0.001). On the right, the 'Preview' panel shows a live webcam feed and an 'Output' section with four color-coded bars: 'blank' (orange, 100%), 'paper' (pink), 'scissor' (purple), and 'rock' (blue). The interface is in English and version 2.4.4.

# Training details

---

Teachable Machine splits your samples into two buckets

- Training samples: 85% to train the model
  - How to correctly classify new samples into the classes
- Test samples: 15% never used to train the model, for checking
  - Used to check how well is the model on new, never-before-seen data



# Training details

## Underfit

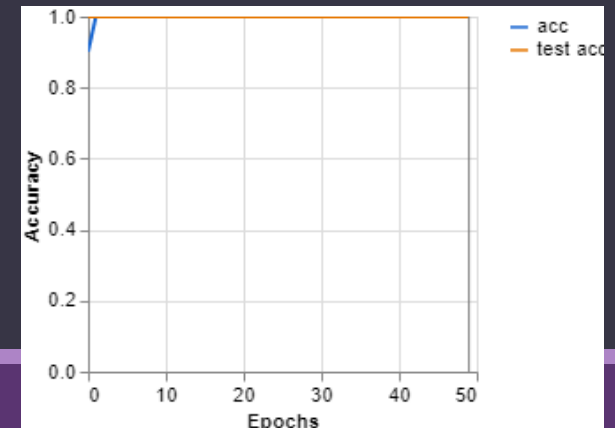
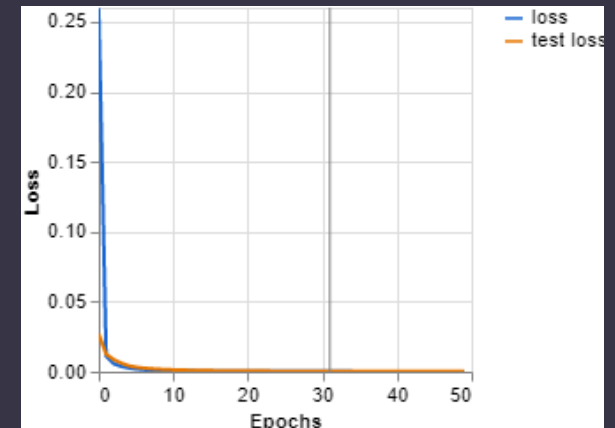
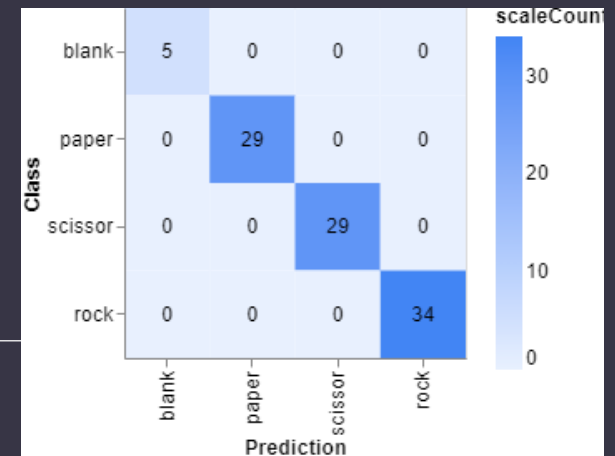
- Classifies poorly
- Model hasn't captured the complexity of the training samples

## Overfit

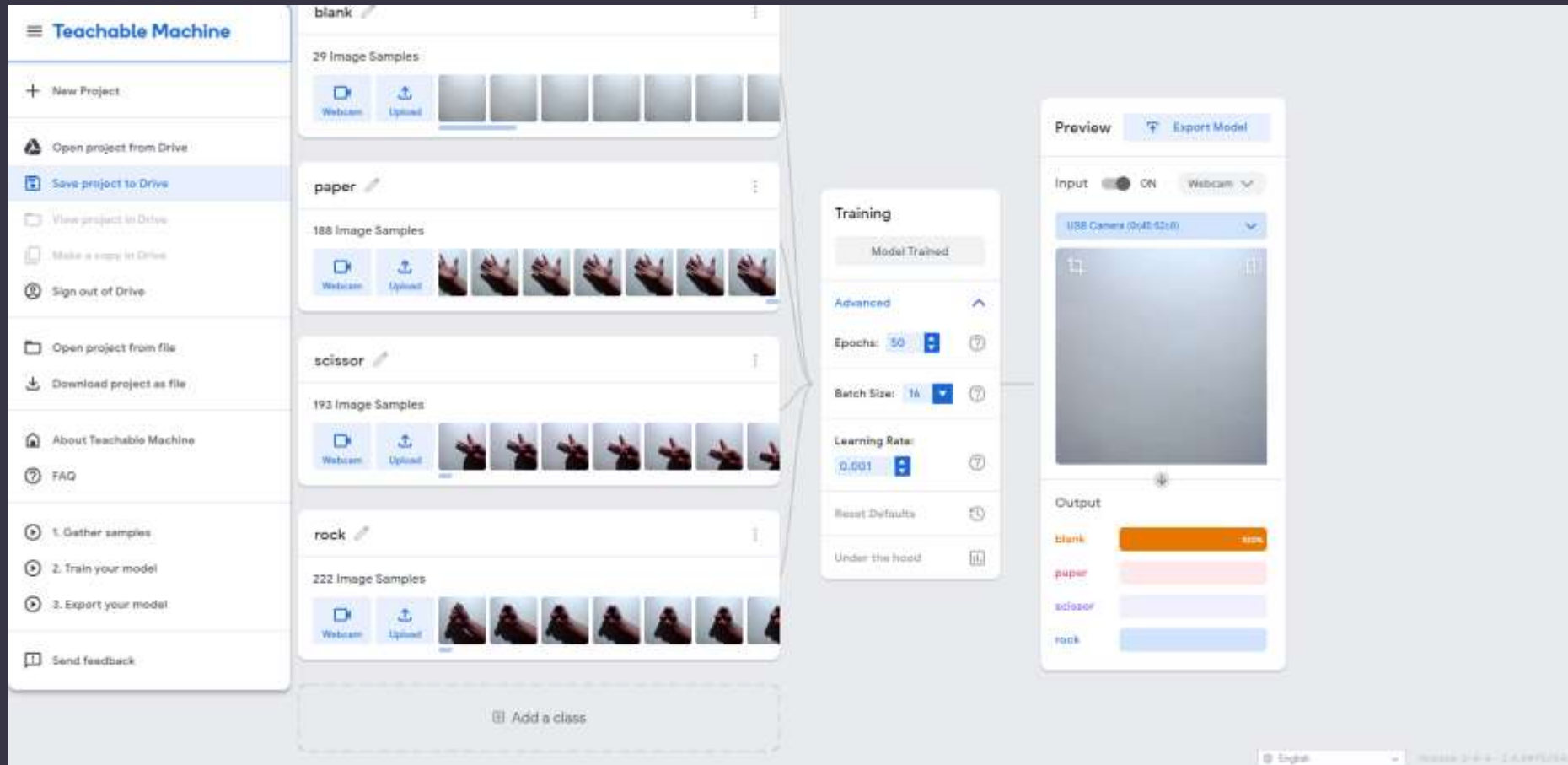
- Model classified the training samples too closely
- Fails to make correct classifications on the test samples
- Usually sample dropout is used (discard, throw away some training samples)

## Epochs

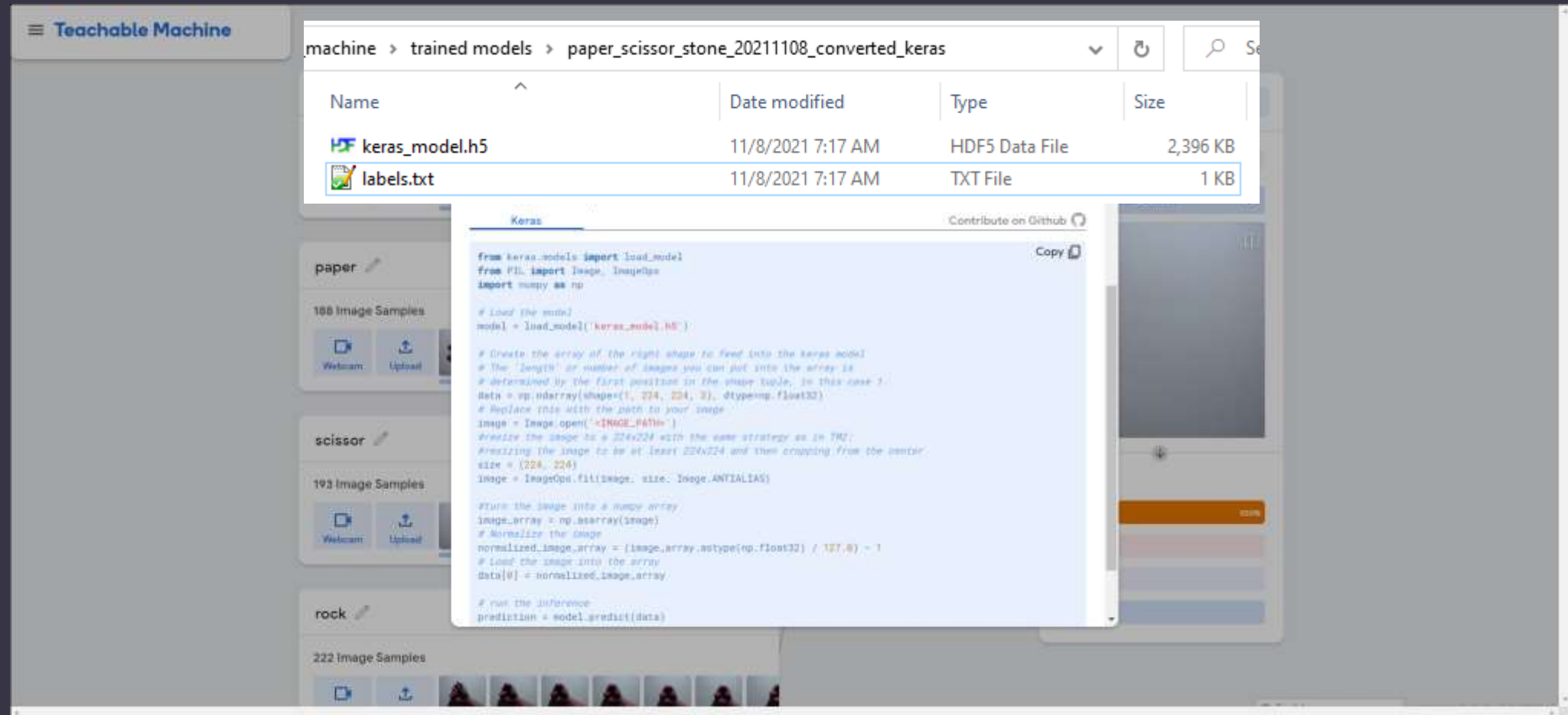
- Every training sample has been fed through the model at least **once**
- For epochs = 50, the model you are training will work through the entire training dataset 50 times.



# Export to google drive



# Export trained model for GTM



The screenshot shows the Teachable Machine interface with a file explorer for the model 'paper\_scissor\_stone\_20211108\_converted\_keras'. The file explorer lists two files: 'keras\_model.h5' (2,396 KB) and 'labels.txt' (1 KB). A code editor window is open, displaying Python code for loading and running the Keras model.

```
from keras.models import load_model
from PIL import Image, ImageOps
import numpy as np

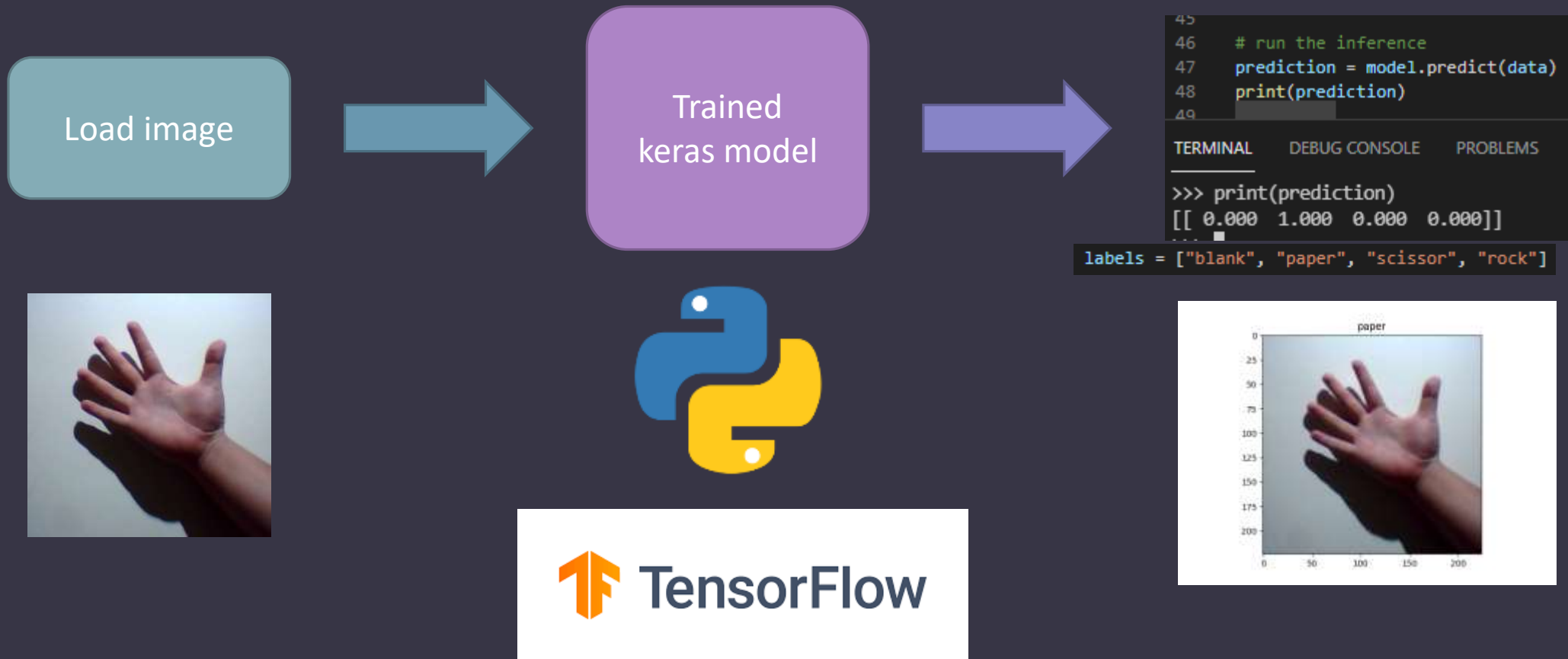
# Load the model
model = load_model('keras_model.h5')

# Create the array of the right shape to feed into the keras model
# The 'length' or number of images you can put into the array is
# determined by the first position in the shape tuple, in this case 1.
data = np.zeros(shape=(1, 224, 224, 3), dtype=np.float32)
# Replace this with the path to your image
image = Image.open('<IMAGE_PATH>')
# Resize the image to a 224x224 with the same strategy as in TRZ:
# Resizing the image to be at least 224x224 and then cropping from the center
size = (224, 224)
image = ImageOps.fit(image, size, Image.ANTIALIAS)

# Turn the image into a numpy array
image_array = np.asarray(image)
# Normalize the image
normalized_image_array = (image_array.astype(np.float32) / 127.5) - 1
# Load the image into the array
data[0] = normalized_image_array

# run the inference
prediction = model.predict(data)
```

# Inference process on python





# Sample code

gtm\_rockpaperscissor.py X

gtm\_rockpaperscissor.py > ...

```
1  """gtm_rockpaperscissor.py"""
2
3  from keras.models import load_model
4  from PIL import Image, ImageOps
5  import numpy as np
6
7  # Load the model
8  model = load_model('keras_model.h5')
9
10 # Create the array of the right shape to feed into the keras model
11 # The 'length' or number of images you can put into the array is
12 # determined by the first position in the shape tuple, in this case 1.
13 data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
14 # Replace this with the path to your image
15 image = Image.open('<IMAGE_PATH>')
16 #resize the image to a 224x224 with the same strategy as in TM2:
17 #resizing the image to be at least 224x224 and then cropping from the center
18 size = (224, 224)
19 image = ImageOps.fit(image, size, Image.ANTIALIAS)
20
21 #turn the image into a numpy array
22 image_array = np.asarray(image)
23 # Normalize the image
24 normalized_image_array = (image_array.astype(np.float32) / 127.0) - 1
25 # Load the image into the array
26 data[0] = normalized_image_array
27
28 # run the inference
29 prediction = model.predict(data)
30 print(prediction)
31
```

```
# Function to determine label
def prediction_outcome(prediction, labels):
    # Find index with maximum probability
    max_index = np.argmax(prediction)
    # Treat probability with higher than 0.5 as identified
    if prediction.max() > 0.5:
        show_text = labels[max_index]
    else:
        show_text = "Not identified"
    print(show_text)
    return show_text

# Plot data and prediction
result = prediction_outcome(prediction, labels)
img = mpimg.imread(test_image)
imgplot = plt.imshow(img)
plt.title(result)
plt.show()
```

# Extra notes

---

CPU/GPU version of trained model are sometimes not interchangeable

- Make sure training is done on similar setup
- Train on CPU => Infer on CPU
- Train on GPU => Infer on GPU
- ~~Train on GPU => Infer on CPU~~

The analogy to deep learning is that

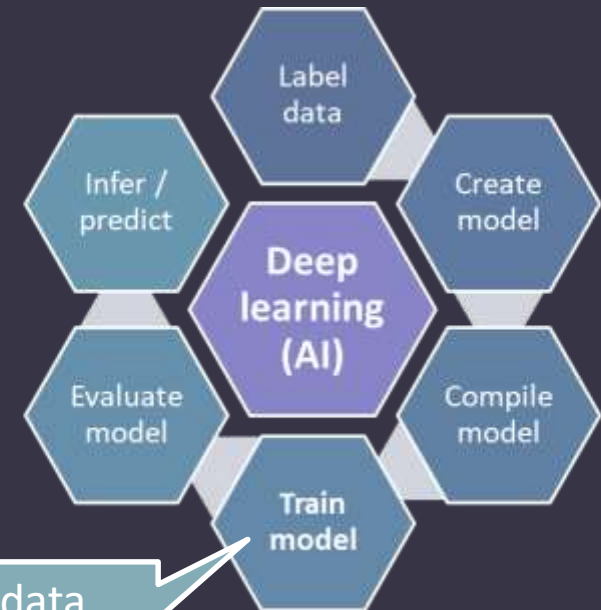
- the rocket engine is the deep learning models and
- the fuel is the huge amounts of data we can feed to these algorithms.

- Andrew Ng

- Co-founder and head of Google Brain
  - Former Chief Scientist at Baidu
  - Co-Founder of Coursera

# Teachable machine

- GOOGLE INC.
- BRIDGE FROM ML TO DL
- TEACH MACHINES TO THINK (DL)



Using shuffled data  
(CPU/GPU exhaustive)

# End of presentation

---

THANK YOU!

# Bibliography

---

Middleton, M. (2021, February 8). *Deep Learning vs. Machine Learning—What's the Difference?*  
<https://flatironschool.com/blog/deep-learning-vs-machine-learning/>

Rowe, W., and Johnson, J. (2020, September 8). *Top Machine Learning Frameworks To Use in 2021*. BMC Blogs.  
<https://www.bmc.com/blogs/machine-learning-ai-frameworks/>

Copeland, M. (2016, July 29). *The Difference Between AI, Machine Learning, and Deep Learning?* | NVIDIA Blog. The Official NVIDIA Blog. <https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/>

Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. *ArXiv:2004.10934 [Cs, Eess]*. <http://arxiv.org/abs/2004.10934>

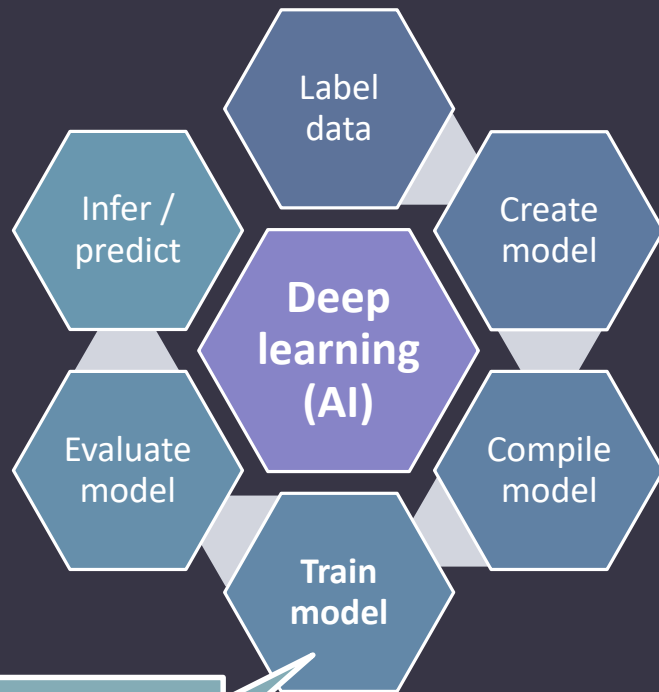
Thompson, N. C., KRISTJAN GREENEWALD, KEEHEON LEE, and GABRIEL F. MANSO. (2021, September 24). *Deep Learning's Diminishing Returns*. IEEE Spectrum. <https://spectrum.ieee.org/deep-learning-computational-cost>

Mahapatra, S. (2019, January 22). *Why Deep Learning over Traditional Machine Learning?* Medium. <https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063>

Garling, C. (2015, May). *Andrew Ng: Why 'Deep Learning' Is a Mandate for Humans, Not Just Machines* | WIRED. <https://www.wired.com/brandlab/2015/05/andrew-ng-deep-learning-mandate-humans-not-just-machines/>

# Machine learning using TDR signals

---



Using shuffled data  
(CPU/GPU exhaustive)