Introduction to Deep Learning

AAG Hawai meeting

April 18, 2024

Benoit Parmentier

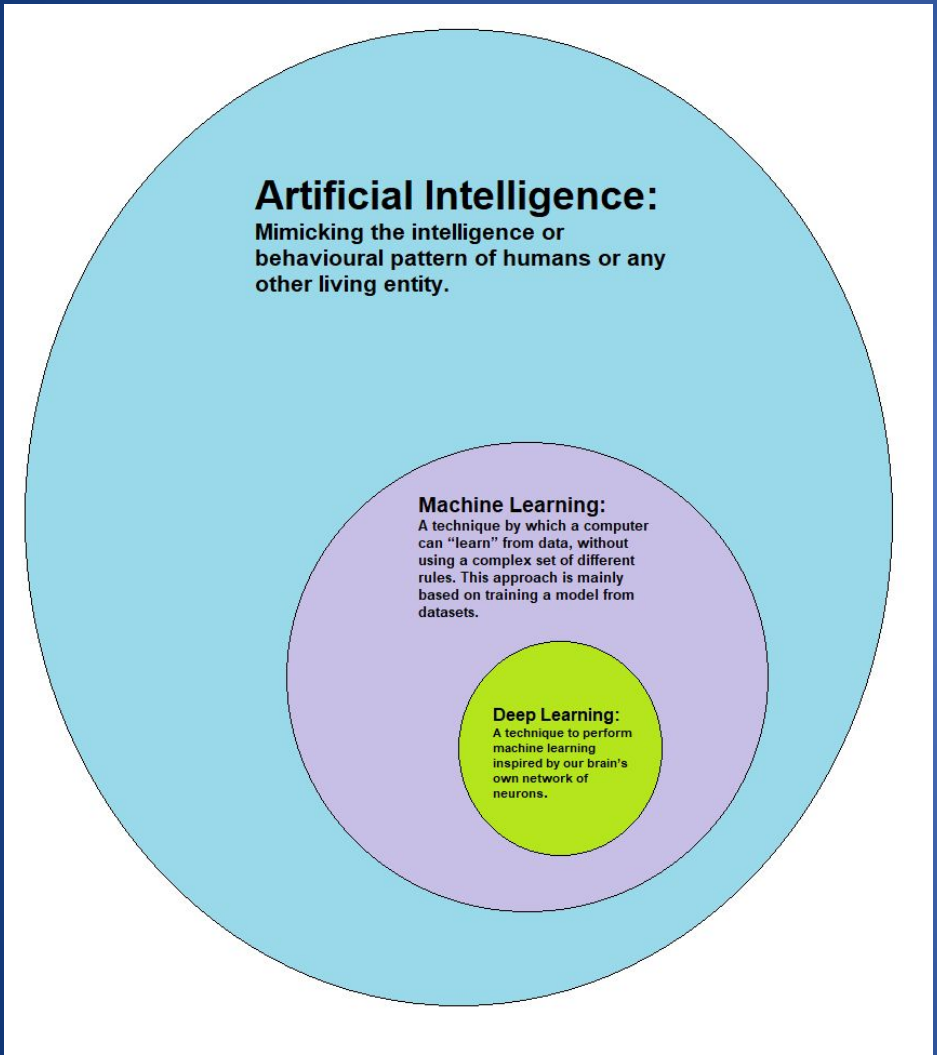
University of Mary Washington, VA

benoit.parmentier@gmail.com

https://www.benoitparmentier.org/

https://github.com/bparment1

Deep Learning



- Artificial Intelligence
- Machine Learning
- Deep Learning

- Image from wikipedia

Deep Learning Key moments

- 1967: multi layers perceptron model published
- 1986: Rumelhart Hinton and Williams popularized backpropagation in a paper.
- 1989 backpropagation to learn kernels in CNN for handwritten zip code recognition
- 1998: Lenet5 architecture inspiration for many later CNNs (7 layers).
- 2006: Hinton et al. introduce methods to train NN with many hidden layers (deep networks).
- **2012:** Alexnet wins the Imagenet competition, data augmentation is introduced
- **2013:** R- CNN published for object detection
- **2015: Unet Architecture for semantic segmentation**

TYPES OF DATA AND INPUTS IN DEEP LEARNING NETWORK

- **structured (tabular):**
  in the form of a table/data frame with column corresponding to the features/covariates

- **text:**
  string of characters with words and sentence. Although, it is sometime called unstructured, word association and sequences contain pattern and structure.

- **images:**
  matrix of width and height, also called unstructured data but contains spatial configuration that can be leveraged in deep learning networks.

- **networks:**
  matrix of width and height, also called unstructured data but contains spatial configuration that can be leveraged in deep learning networks.

- **sound:**
  matrix of width and height, also called unstructured data but contains spatial configuration that can be leveraged in deep learning networks.

- **others:**
  biochemistry, biology etc.

Deep Learning Key Players

- **Yann Lecun:**
  French computer scientist, contributed to CNN with learned kernels. Famous for the Lenet network for digit classification.

- **Geoffrey Hinton:**
  based at the University of Toronto Computer Science department, early pioneer. Contributed to many incremental improvements that led to the Deep Learning revolution. One of the author of Alexenet.

- **Yoshua Bengio:**
  based at Montreal Institute of Machine Learning, early contributor via Theano library.

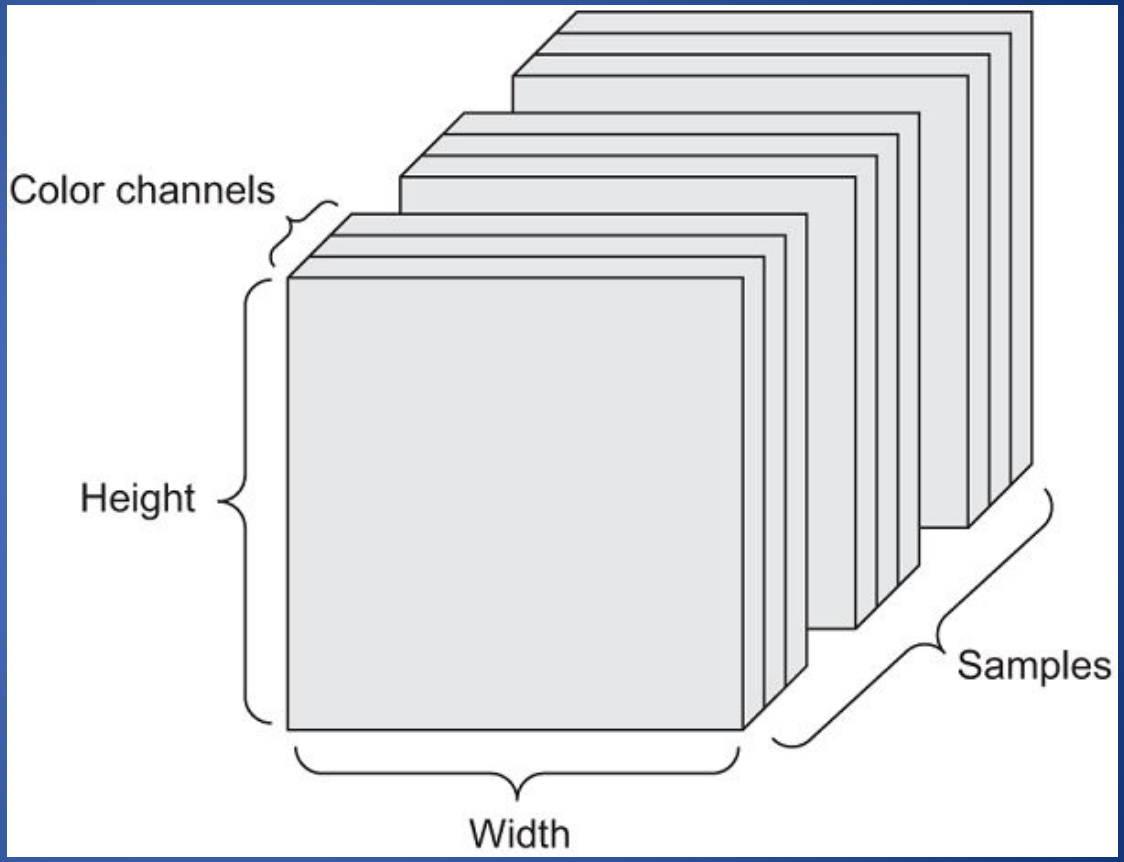
- **Ian Goodfellow:**
  Inventor of GAN (Generative Adversarial Network) now works at Google.

Key Terminology in relation to Earth Observation/RS

- **Image classification**: Assigning a label to a whole image (not at a pixel level).
- **Object localization**: Assigning an Image Label and bounding box for an object (e.g. cat). This assumes that there is only one object of one class present in the image.
- **Object Detection:** assigning labels and bounding boxes to many objects in images.
- **Semantic Segmentation:** assigning a class to every pixel in an image.
- **Instance Segmentation:** assigning a class an item number to object (e.g. car1, car2, bike1, bike2)

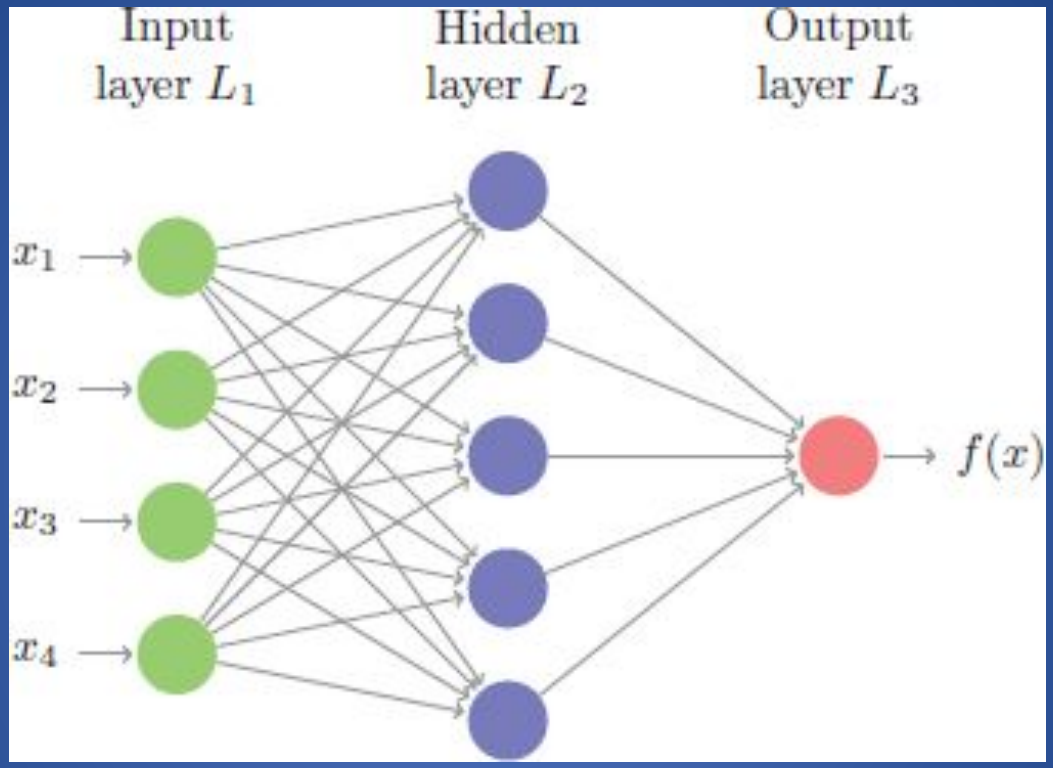
Key Terminology in relation to Earth Observation/RS

- **height: number of rows**
- **width**: number of columns
- channels: RS bands
- samples: number of images



https://livebook.manning.com/concept/deep-learning/shape

DNN FOR CLASSIFICATION/REGRESSION



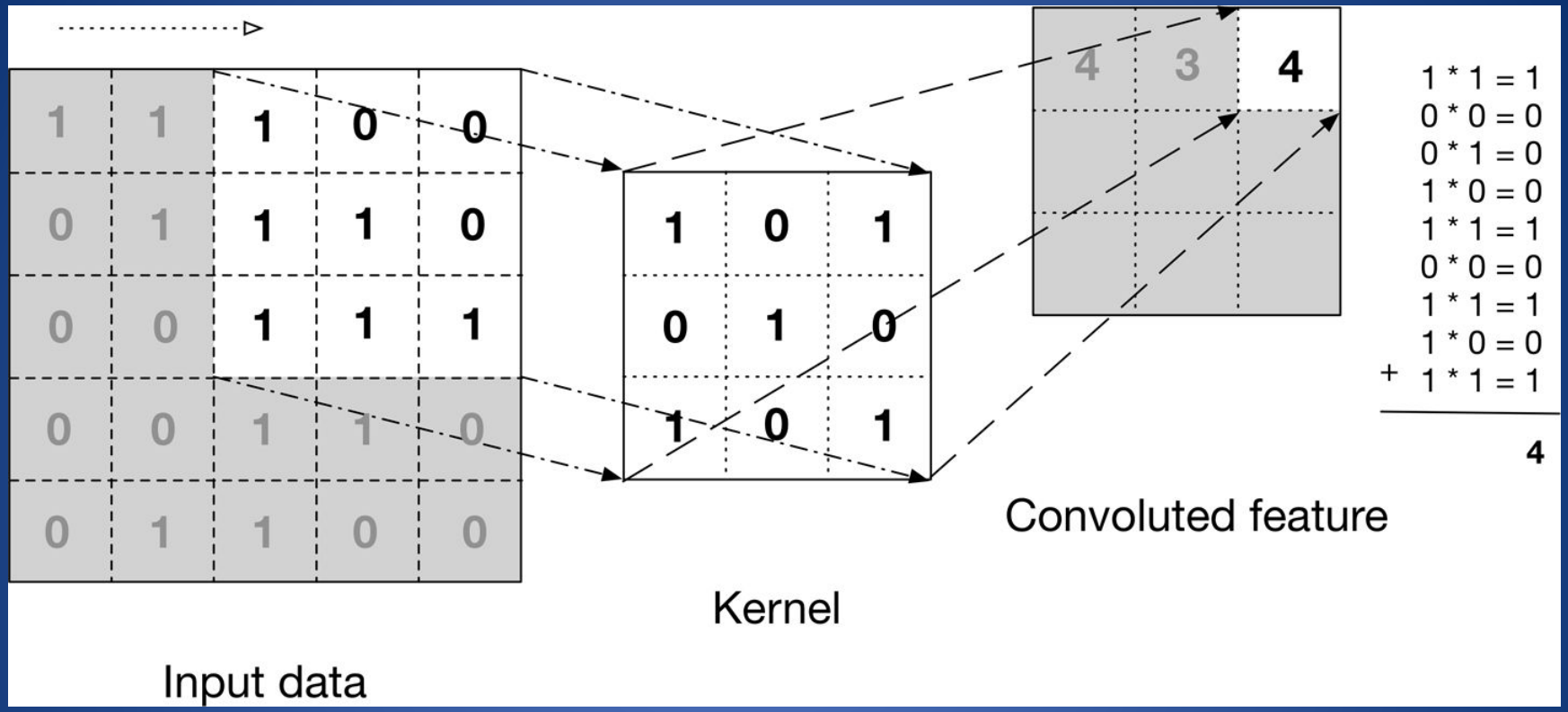
- Input layer has a number of nodes equal to features
- Hidden layers are the layers between input and output.
- Output layer has the number of nodes equal to the number of classes in classification or dimensions in regression.

DNN FOR CLASSIFICATION ON IMAGES

```
model_dnn = keras.Sequential()
model_dnn.add(layers.Flatten(input_shape=x_train.shape[1:]))
model_dnn.add(layers.Dense(512,activation='relu'))
model_dnn.add(layers.Dense(10,activation='softmax'))

model_dnn.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

model_dnn._name = 'model_dnn' #set keras model name
```

- **F**or multi-class classification, the final activation layer is softmax. Scores sum to 1 and can be interpreted as probabilities.
- Note that the categorical cross entropy is used as a loss. This is the case for multi-class classification.

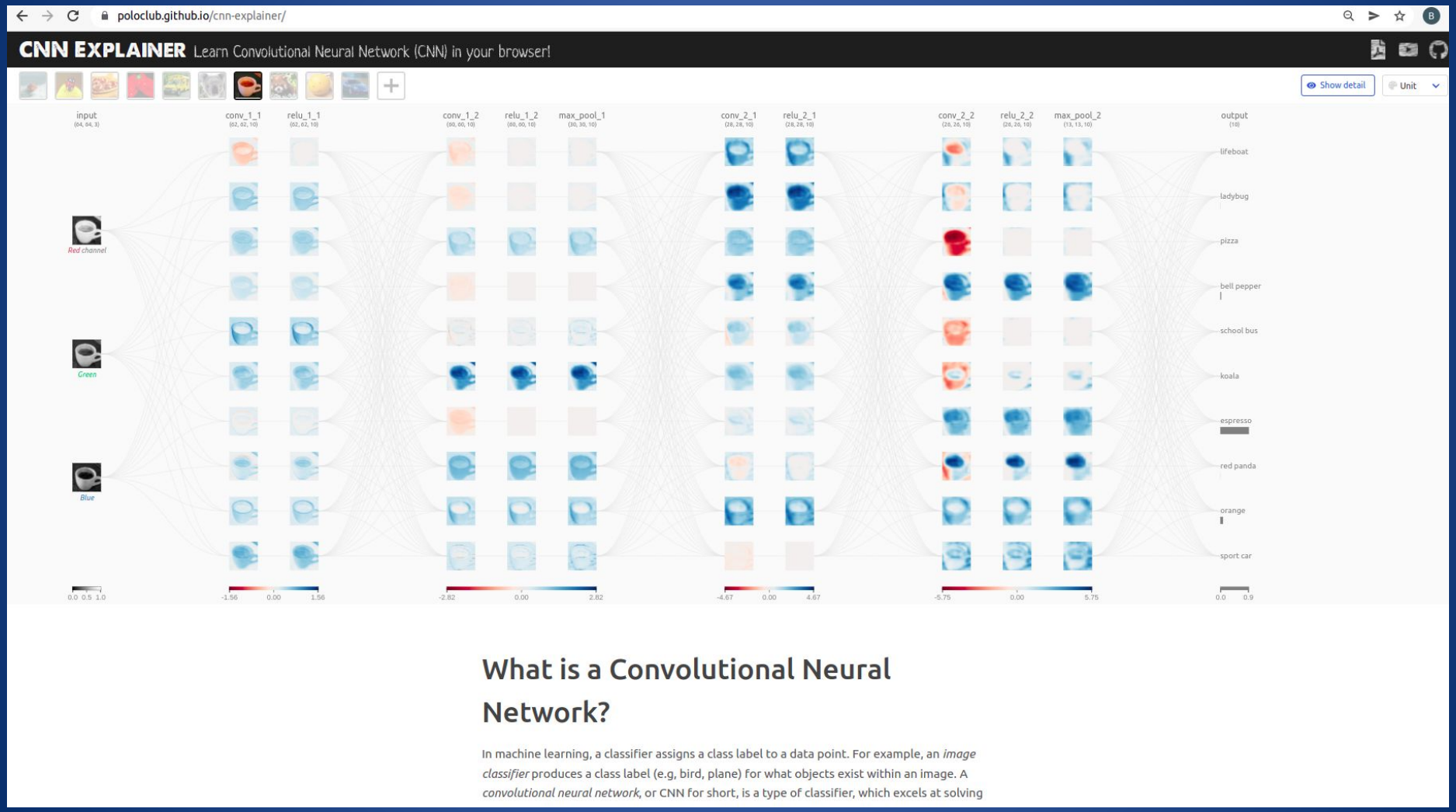
CONVOLUTIONAL LAYER



https://miro.medium.com/max/2880/0*QS1ArBEUJjjySXhE.png

- A layer that uses kernel and generates feature maps.
- The weights in the kernel are learned by the network.
- It reduces greatly the number of parameters compared to a Dense layer and leverage spatial information/structure.
- https://poloclub.github.io/cnn-explainer/

CNN EXPLAINER

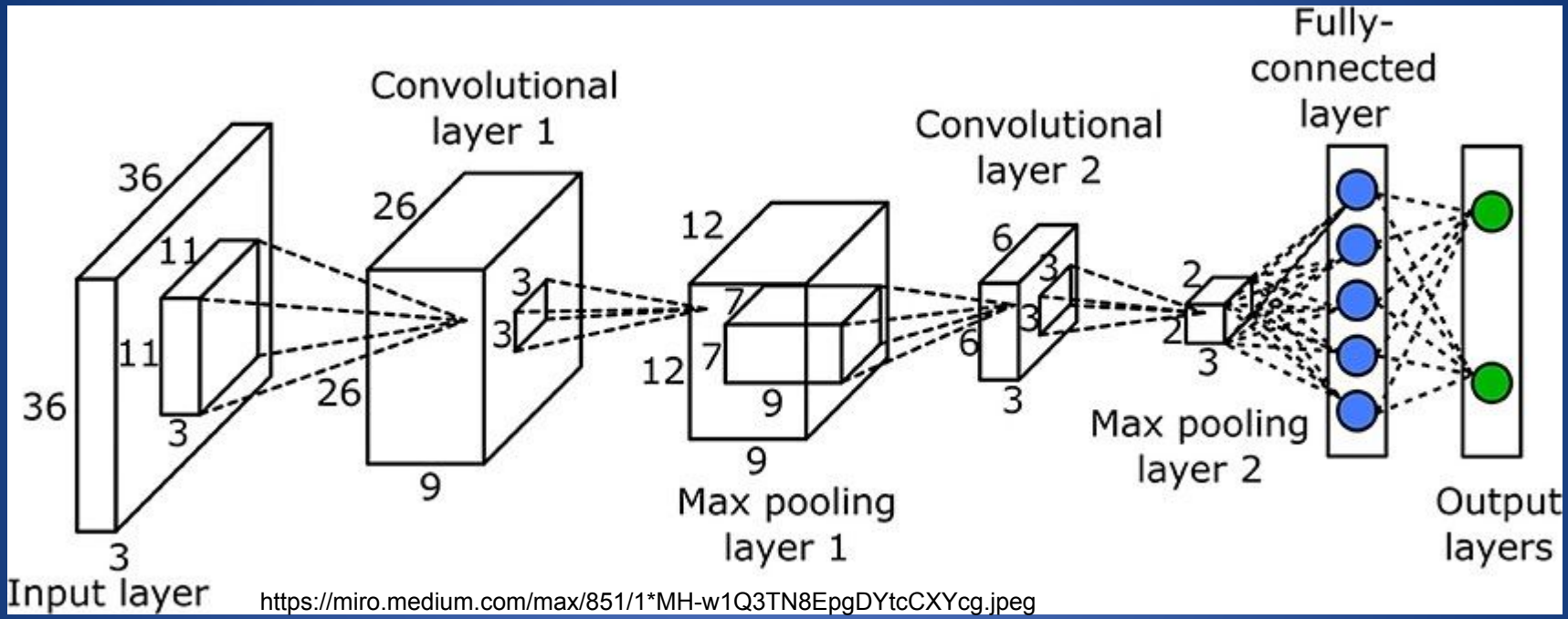


- https://poloclub.github.io/cnn-explainer/

CONVOLUTIONAL LAYER PARAMETERS

- kernel size: Height x Width typically 3x3 or 5x5.
- stride: steps between sliding of the kernels. Typically 1. If greater than one the size of the image may decrease more.
- padding: padding added when modeler wants to keep the input size of the image.
- filter: in keras, it is the number of feature maps (generated by the equivalent number of kernels)

CNN FOR CLASSIFICATION



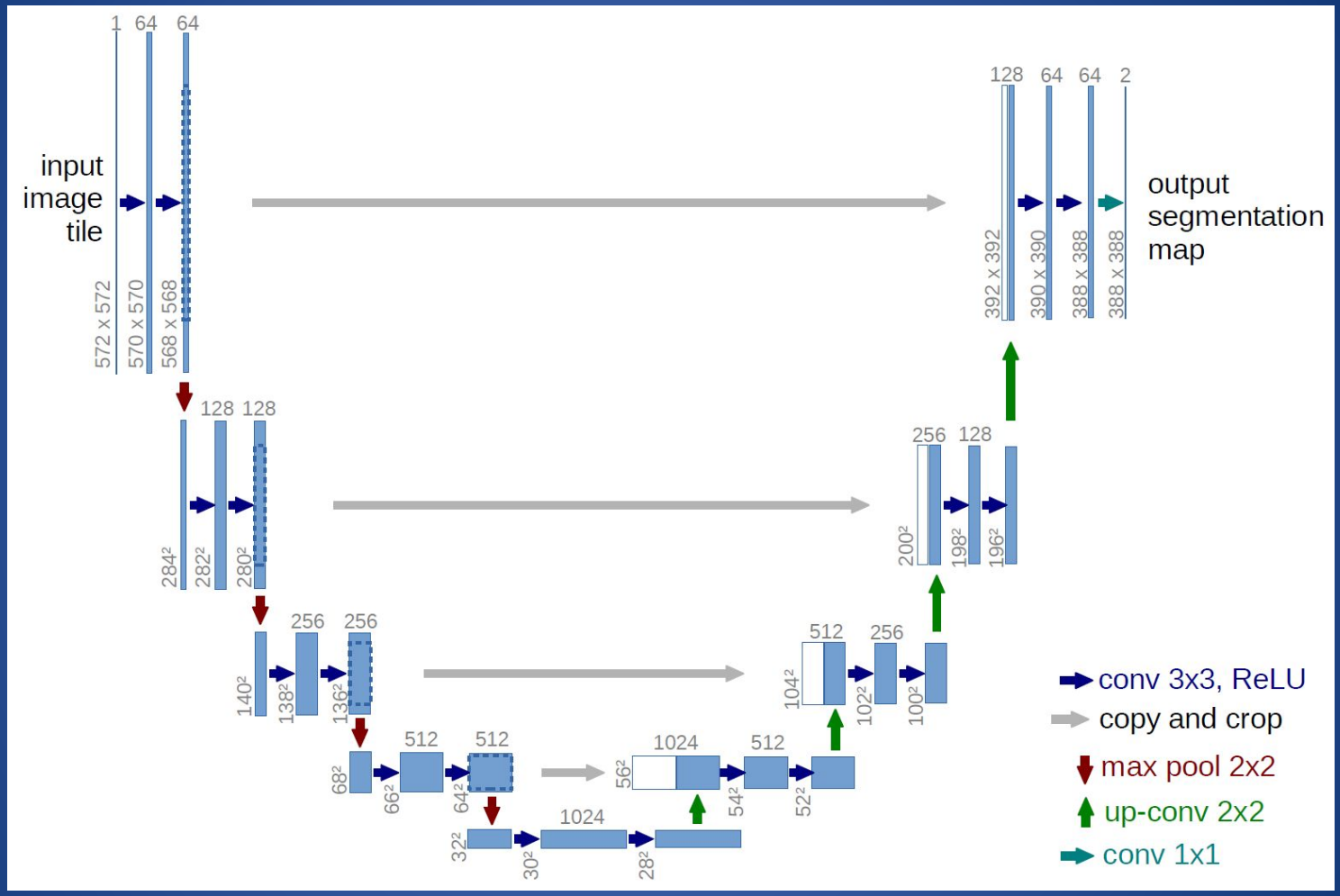
https://miro.medium.com/max/851/1*MH-w1Q3TN8EpgDYtcCXYcg.jpeg

- A typical sequence in a CNN: Convolutional 2D followed by max pooling.
- The network head is a dense neural network (fully connected).

CNN FOR CLASSIFICATION

```
num_classes =10
model_cnn = Sequential()

## CNN part
#Block 1
model_cnn.add(Conv2D(32, (3, 3), padding='same',
                 activation='relu',
                 input_shape=x_train.shape[1:]))
model_cnn.add(Conv2D(32,(3,3),padding='same', activation='relu'))
model_cnn.add(MaxPooling2D(pool_size=(2,2)))

#Block 2
model_cnn.add(Conv2D(64,(3,3),padding='same',activation='relu'))
model_cnn.add(Conv2D(64,(3,3),padding='same',activation='relu'))
model_cnn.add(MaxPooling2D(pool_size=(2,2)))

#Block 3:
model_cnn.add(Conv2D(128,(3,3),padding='same',activation='relu'))
model_cnn.add(Conv2D(128,(3,3),padding='same',activation='relu'))
model_cnn.add(MaxPooling2D(pool_size=(2,2))) #becomes 4x4 image

## DNN part
model_cnn.add(Flatten())
model_cnn.add(Dropout(0.2))
model_cnn.add(Dense(1024,activation='relu',kernel_constraint=maxnorm(3)))
model_cnn.add(Dropout(0.2))
model_cnn.add(Dense(num_classes, activation='softmax'))
```

UNET FOR SEMANTIC SEGMENTATION



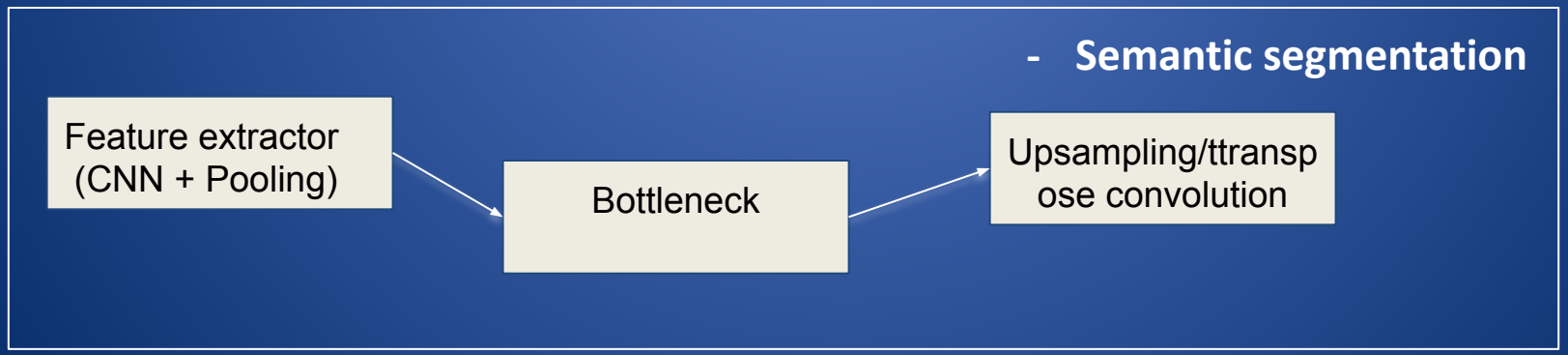
- Encoder-Decoder structure
- Output is an image of the same size with probabilities for every pixel.

NETWORK ARCHITECTURES

- **Image classification**

- **Object localization**

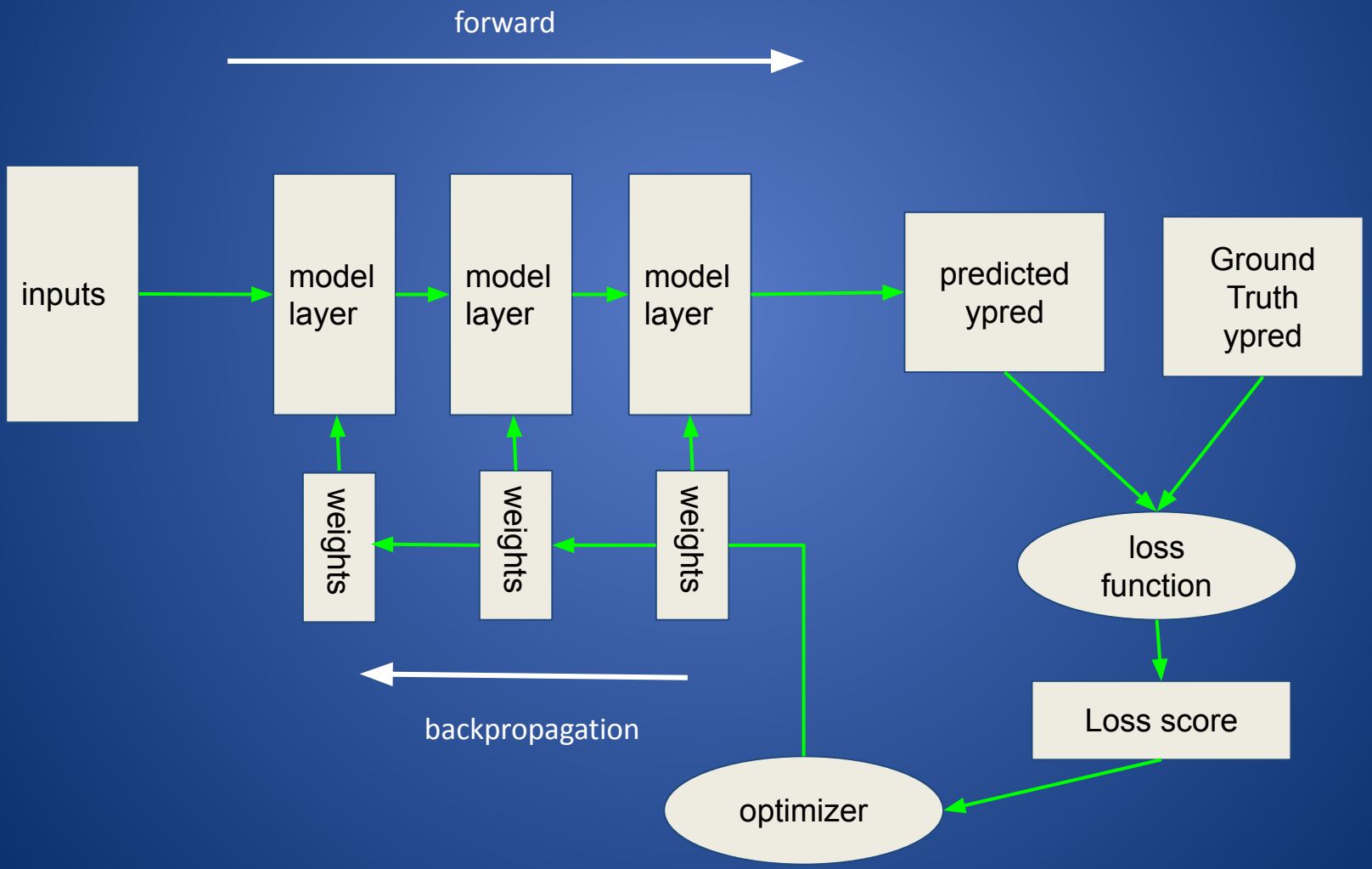
- **Semantic segmentation**



Deep Learning Key Software Libraries

- Theano: one of the early Deep Learning library created by MILA (University of Montreal) by team led by Yoshua Bengio.
- Tensorflow: created a google for internal research in AI, was open sourced in 2015. It is now one of the main library used in industry and in research.
- Keras: created in 2015 as a high level API sitting on top of Theano, Tensorflow and CNTK. It is now one of most popular Deep Learning libraries.
- **pytorch:** created by facebook AI, now one of the main library used in industry and research.
- **fastai: high level libraries that sits on top of pytorch with the goal of making Deep Learning accessible to all. Developed at the University of San Francisco.**

Training a deep learning model



Compiling model

```
#NOTE INPUT SHOULD BE THE NUMBER OF VAR
#### Test with less number of input nodes: pruning
model_dnn = Sequential()
model_dnn.add(Dense(4, input_dim=3, activation='relu'))
model_dnn.add(Dense(10, activation='relu'))
model_dnn.add(Dense(10, activation='relu'))
model_dnn.add(Dense(1)) #scalar regression, end DNN without activation function as we are predicting continuous values

model_dnn.compile(optimizer="adam",
                  loss="mse",
                  metrics=["mae"])

model_dnn.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 4)                 16
_________________________________________________________________
dense_1 (Dense)              (None, 10)                50
_________________________________________________________________
dense_2 (Dense)              (None, 10)                110
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 11
=================================================================
Total params: 187
Trainable params: 187
Non-trainable params: 0
```

- **Once a model is created, you need to compile the model and select the optimizer, loss function and metrics.**

Loss Function

- A loss function is an error function that is optimized in the model.
- It is function of the weights and it is used to find weight values to update.
- Gradient descent is used to find the minimum of the loss function and the optimal weights.
- Loss function types varies in function of input data: continuous (e.g. MSE) or categorical (e.g. binary cross entropy).

Loss Function

- MSE: Mean Square Error is used as loss function for neural networks with a continuous target (regression).
- Binary cross entropy is used for binary classification or multilabel cases.
- Categorical cross-entropy is used for multi class cases (mutually exclusive classes).

$$L_{CE} = -\sum_{i=1}^{n} t_i \log(p_i), \text{ for n classes,}$$

where t_i is the truth label and p_i is the Softmax probability for the i^{th} class.

https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e

Optimizer

- It is the method used to update the weights so that we decrease the loss (error).
- It is usually a variation of the the gradient descent method to reach the minimum of the loss function.
- There are many optimizers. Some common ones are SGD, RMSprop and Adam.

Optimizers

- **SGD**: gradient descent that is updated by batch (mini-batch gradient descent).
- **SGD with momentum**: consider previous gradient values to carry out current weight update.
- **SGD with scheduled learning rate**: pre-defined decrease in rate of learning.
- **Adagrad:** scheduled learning called Adaptive learning rates for each input features based on update frequency for each input features.
- **Rmsprop**: improvement of Adagrad with better update of learning rates.
- **Adams**: consider both scheduled learning rate and momentum trying to combine all the improvements together in one algorithm.

The importance of Labeling

- Deep Learning requires a large amount of labeled data training.
- These are usually hand-labeled datasets that are expensive and time consuming to generate: https://www.bbc.com/news/technology-56414491
- There are softwares to help.
- New paradigm: weak learning.

LABELING BY HUMAN

AI: Ghost workers demand to be seen and heard

By Jane Wakefield
Technology reporter

4 days ago

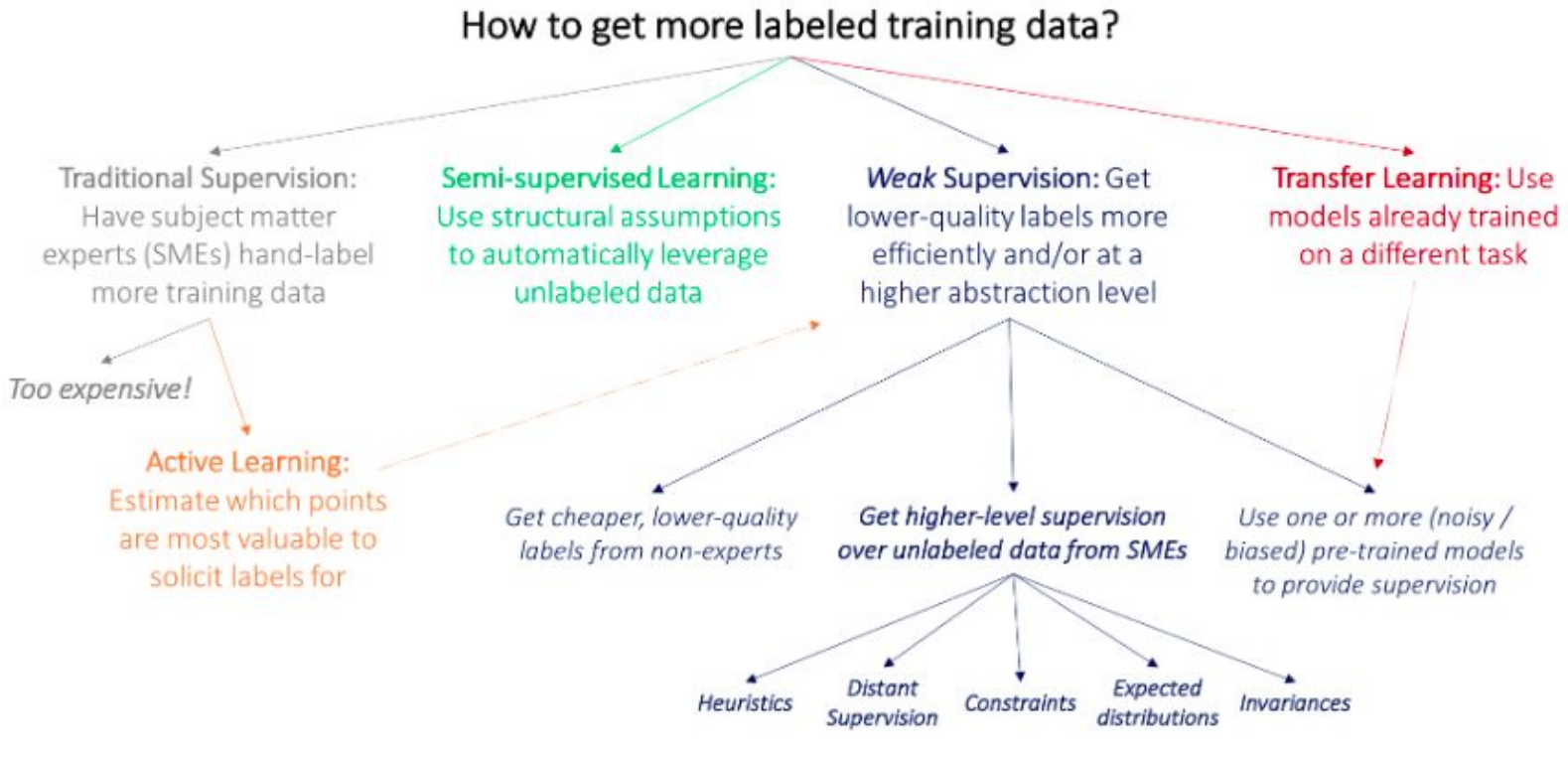
Artificial intelligence and machine learning exist on the back of a lot of hard work from humans.

Alongside the scientists, there are thousands of low-paid workers whose job it is to classify and label data - the lifeblood of such systems.

But increasingly there are questions about whether these so-called ghost workers are being exploited.

- https://www.bbc.com/news/technology-56414491

The importance of Labeling



Transfer Learning and Pre-training

- Use model trained on another related task to predict on the task at hand.
- Can leverage networks that were trained on a large training dataset.
- Often requires removing the ‘head’ of the model to use the ‘base’.
- This method has proven very successful in many instances.

Introduction to Deep Learning

AAG Hawai meeting

April 18, 2024

Benoit Parmentier

University of Mary Washington, VA

benoit.parmentier@gmail.com

https://www.benoitparmentier.org/

https://github.com/bparment1

Introduction to Deep Learning

University of Mary and Washington

April 11, 2024

Benoit Parmentier

University of Mary Washington, VA

benoit.parmentier@gmail.com

https://www.benoitparmentier.org/

https://github.com/bparment1

References

- https://ai.facebook.com/blog/billion-scale-semi-supervised-learning/
- https://www.azavea.com/blog/2020/03/24/labeling-satellite-imagery-for-machine-learning/
- http://ai.stanford.edu/blog/weak-supervision/
- https://www.d2l.ai/chapter_preface/index.html