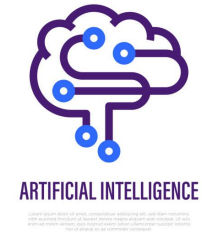


Machine Learning Talks

https://github.com/kaopanboonyuen/mltalks

Reference:

1. https://pytorch.org/tutorials
2. https://stanford.edu/~shervine/teaching/cs-229/
3. http://introtodeeplearning.com/
4. https://www.simplilearn.com/tutorials/deep-learning-tutorial/introduction-to-deep-learning
5. https://www.geeksforgeeks.org/introduction-deep-learning/

About Me

Kao Panboonyuen

kao-panboonyuen

AI Research Scientist

Name: Kao

Contact: teerapong.pa@chula.ac.th

panboonyuen.kao@gmail.com

Education: Ph.D. in Computer Eng., Chula

Position: AI Team Lead, MARS

PostDoc, Chula

Interests: Computer Vision, Deep Learning

Machine Learning

Teerapong Panboonyuen, Ph.D.

FOLLOW

Postdoctoral Researcher, Dept. of Computer Engineering, Chulalongkorn University
Verified email at chula.ac.th - Homepage

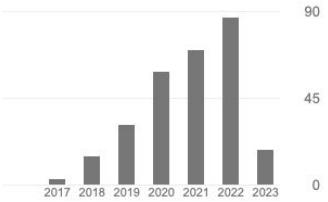
Machine Learning Deep Learning Neural Networks Artificial Intelligence

GET MY OWN PROFILE

TITLE	CITED BY	YEAR
Road segmentation of remotely-sensed images using deep convolutional neural networks with landscape metrics and conditional random fields T Panboonyuen, K Jitkajornwanich, S Lawawirojwong, P Srestasathiern, ... Remote Sensing 9 (7), 680	108	2017
Semantic segmentation on remotely sensed images using an enhanced global convolutional network with channel attention and domain specific transfer learning T Panboonyuen, K Jitkajornwanich, S Lawawirojwong, P Srestasathiern, ... Remote Sensing 11 (1), 83	73	2019
An enhanced deep convolutional encoder-decoder network for road segmentation on aerial imagery T Panboonyuen, P Vateekul, K Jitkajornwanich, S Lawawirojwong Recent Advances in Information and Communication Technology 2017 …	32	2018
Transformer-based decoder designs for semantic segmentation on remotely sensed images T Panboonyuen, K Jitkajornwanich, S Lawawirojwong, P Srestasathiern, ... Remote Sensing 13 (24), 5100	12	2021
Semantic segmentation on medium-resolution satellite images using deep convolutional networks with remote sensing derived indices S Chantharaj, K Pornratthanapong, P Chitsinpchayakun, T Panboonyuen, ... 2018 15th International joint conference on computer science and software …	12	2018
Real-time polyps segmentation for colonoscopy video frames using compressed fully convolutional network I Wichakam, T Panboonyuen, C Udomcharoenchaikit, P Vateekul MultiMedia Modeling: 24th International Conference, MMM 2018, Bangkok …	12	2018
Object detection of road assets using transformer-based YOLOX with feature pyramid decoder on thai highway panorama T Panboonyuen, S Thongbai, W Wongweeranimit, P Santitamnont, ... Information 13 (1), 5	10	2022
Semantic Labeling in Remote Sensing Corpora Using Feature Fusion-Based Enhanced Global Convolutional Network with High-Resolution Representations and Depthwise Atrous Convolution T Panboonyuen, K Jitkajornwanich, S Lawawirojwong, P Srestasathiern, ... Remote Sensing 12 (8), 1233	9	2020

Cited by

	All	Since 2018
Citations	284	280
h-index	8	8
i10-index	7	7



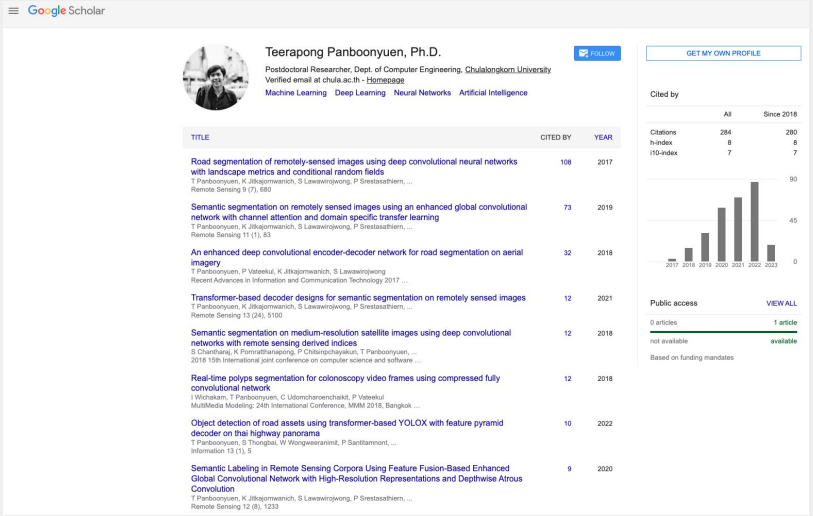
Public access VIEW ALL

0 articles 1 article

not available available

Based on funding mandates

Featured Publications



Google Scholar

Teerapong Panboonyuen, Ph.D.

Postdoctoral Researcher, Dept. of Computer Engineering, Chulalongkorn University
Verified email at chula.ac.th - Homepage

Machine Learning Deep Learning Neural Networks Artificial Intelligence

FOLLOW

GET MY OWN PROFILE

TITLE	CITED BY	YEAR
Road segmentation of remotely-sensed images using deep convolutional neural networks with landscape metrics and conditional random fields T Panboonyuen, K Jitkajornwanich, S Lawawirojwong, P Srestasathiern, ... Remote Sensing 9 (7), 680	108	2017
Semantic segmentation on remotely sensed images using an enhanced global convolutional network with channel attention and domain specific transfer learning T Panboonyuen, K Jitkajornwanich, S Lawawirojwong, P Srestasathiern, ... Remote Sensing 11 (1), 83	73	2019
An enhanced deep convolutional encoder-decoder network for road segmentation on aerial imagery T Panboonyuen, P Vateekul, K Jitkajornwanich, S Lawawirojwong Recent Advances in Information and Communication Technology 2017 ...	32	2018
Transformer-based decoder designs for semantic segmentation on remotely sensed images T Panboonyuen, K Jitkajornwanich, S Lawawirojwong, P Srestasathiern, ... Remote Sensing 13 (24), 5100	12	2021
Semantic segmentation on medium-resolution satellite images using deep convolutional networks with remote sensing derived indices S Chanthorag, K Ponnatthanapong, P Chitsiripchayakun, T Panboonyuen, ... 2018 15th International joint conference on computer science and software ...	12	2018
Real-time polyps segmentation for colonoscopy video frames using compressed fully convolutional network I Wichakam, T Panboonyuen, C Udomcharoenchaikit, P Vateekul MultiMedia Modeling: 24th International Conference, MMM 2018, Bangkok ...	12	2018
Object detection of road assets using transformer-based YOLOX with feature pyramid decoder on thai highway panorama T Panboonyuen, S Thongbai, W Wongweeranimit, P Santitamnont, ... Information 13 (1), 5	10	2022
Semantic Labeling in Remote Sensing Corpora Using Feature Fusion-Based Enhanced Global Convolutional Network with High-Resolution Representations and Depthwise Atrous Convolution T Panboonyuen, K Jitkajornwanich, S Lawawirojwong, P Srestasathiern, ... Remote Sensing 12 (8), 1233	9	2020

Cited by

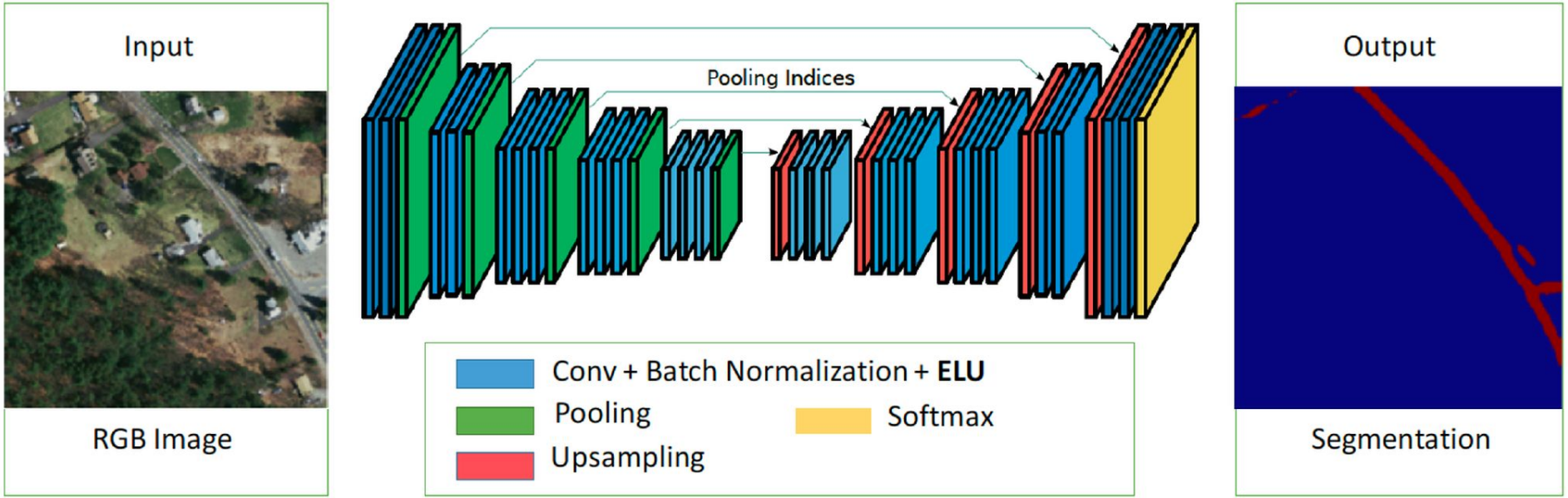
	All	Since 2018
Citations	284	280
h-index	8	8
i10-index	7	7

Public access VIEW ALL

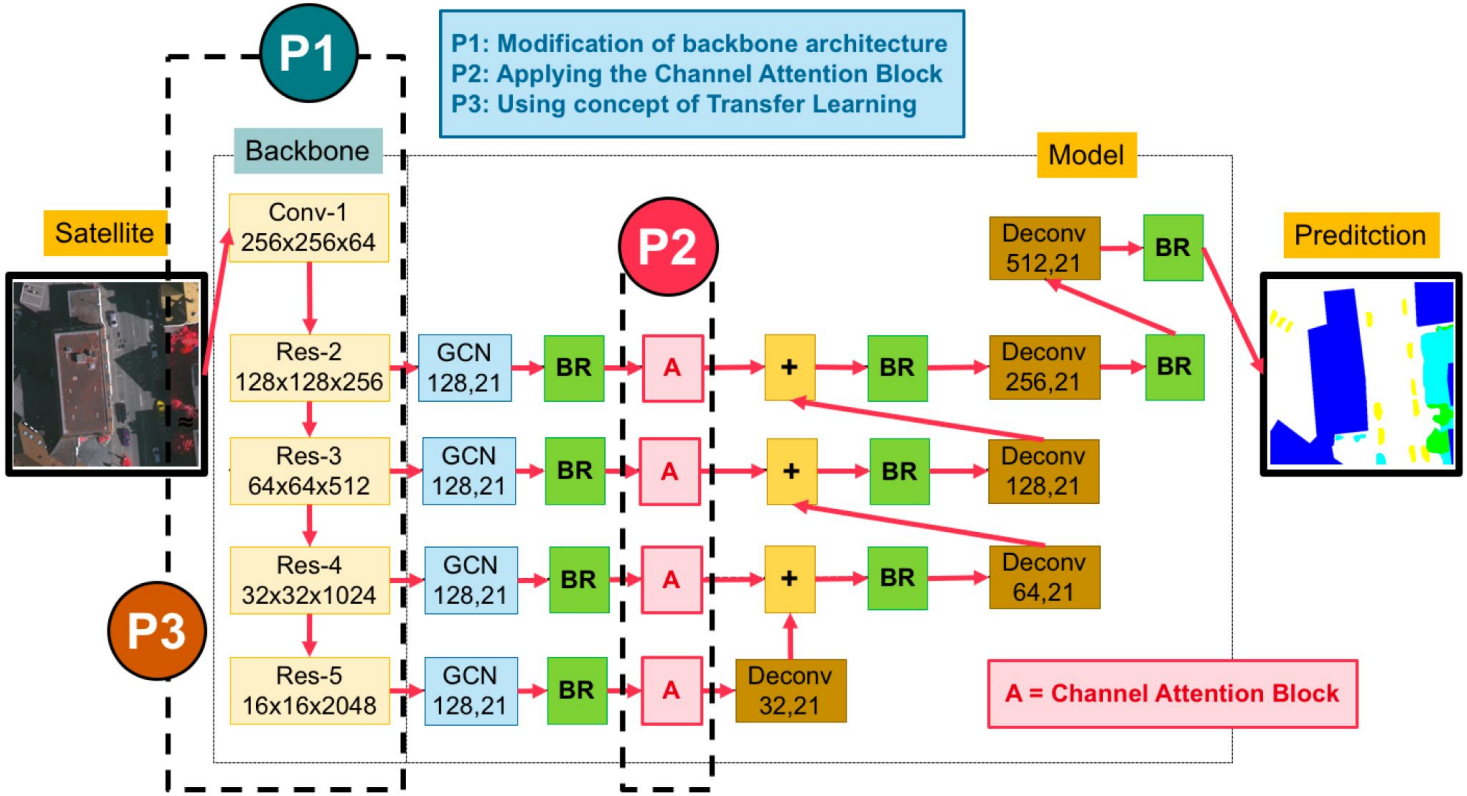
0 articles not available
1 article available

Based on funding mandates

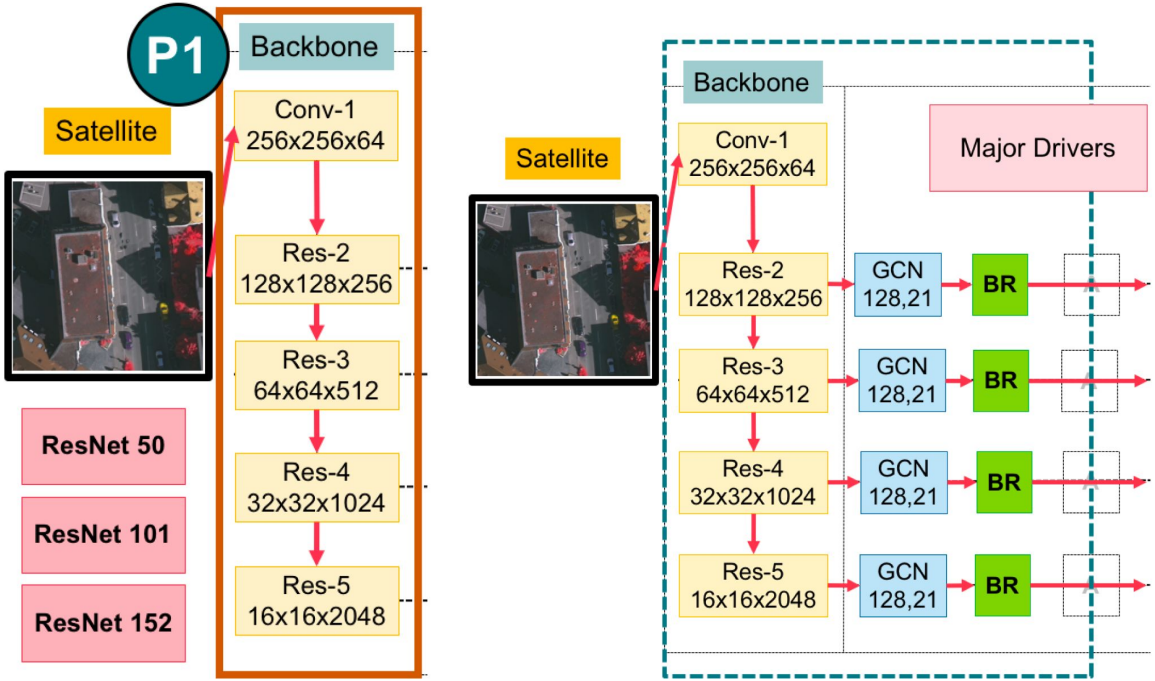
[1] Panboonyuen, Teerapong, et al. **"Road segmentation of remotely-sensed images using deep convolutional neural networks with landscape metrics and conditional random fields."** Remote Sensing 9.7 (2017): 680.



[2] Panboonyuen, Teerapong, et al. **"Semantic segmentation on remotely sensed images using an enhanced global convolutional network with channel attention and domain specific transfer learning."** Remote Sensing 11.1 (2019): 83.

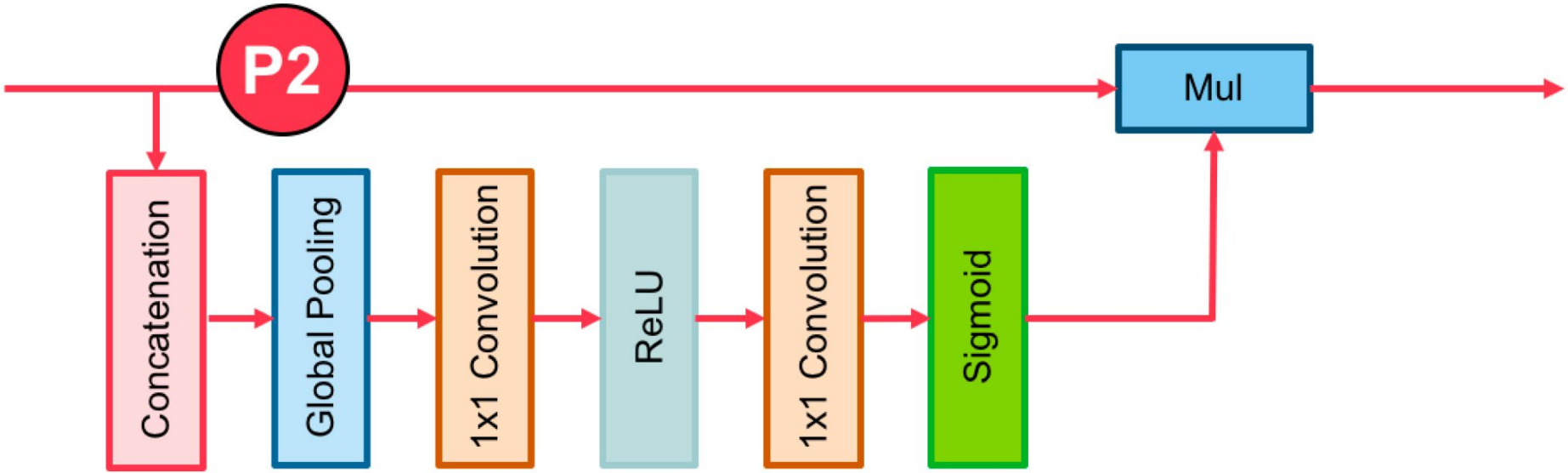


[2] Panboonyuen, Teerapong, et al. **"Semantic segmentation on remotely sensed images using an enhanced global convolutional network with channel attention and domain specific transfer learning."** Remote Sensing 11.1 (2019): 83.



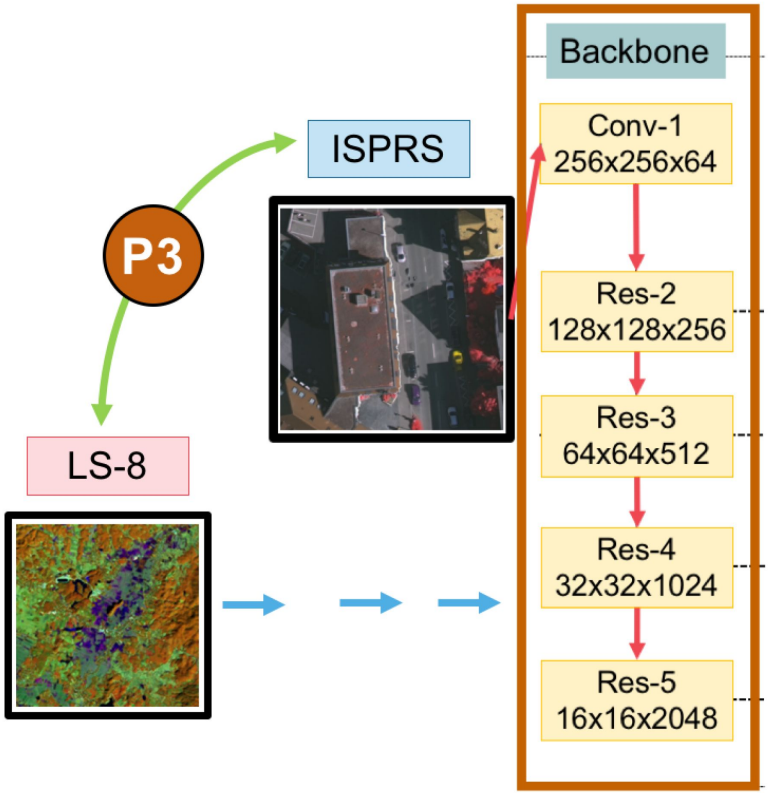
An overview of the whole backbone pipeline in (left) the main backbone with varying by ResNet50, ResNet 101, and ResNet 152; (right) the major drivers of my main classification network (composed of a global convolutional network (GCN) and a boundary refinement (BR) block.

[2] Panboonyuen, Teerapong, et al. **"Semantic segmentation on remotely sensed images using an enhanced global convolutional network with channel attention and domain specific transfer learning."** Remote Sensing 11.1 (2019): 83.



- Components of the channel attention block.
- The red lines represent the downsample operators, respectively.
- The red line cannot change the size of feature maps.
- It is only a path for information passing.

[2] Panboonyuen, Teerapong, et al. **"Semantic segmentation on remotely sensed images using an enhanced global convolutional network with channel attention and domain specific transfer learning."** Remote Sensing 11.1 (2019): 83.



- The domain-specific transfer learning strategy reuses pre-trained weights of models between two datasets—very high (ISPRS) and medium (Landsat-8; LS-8) resolution images.

[2] Panboonyuen, Teerapong, et al. **"Semantic segmentation on remotely sensed images using an enhanced global convolutional network with channel attention and domain specific transfer learning."** Remote Sensing 11.1 (2019): 83.

Figure 6. Sample satellite images from Nan, a province in Thailand (**left**), and corresponding ground truth (**right**). The label of medium resolution dataset includes five categories: agriculture (yellow), forest (green), miscellaneous (brown), urban (red), and water (blue).

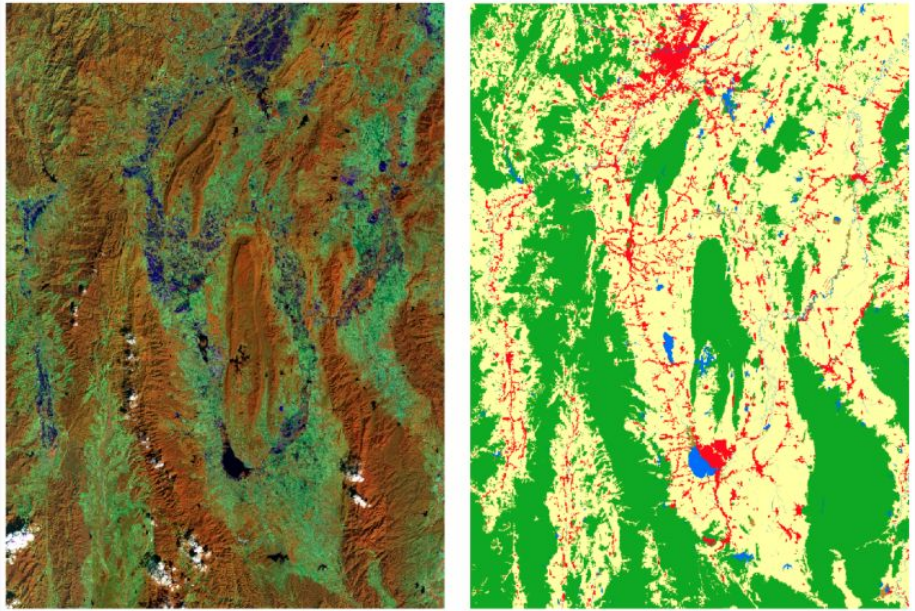
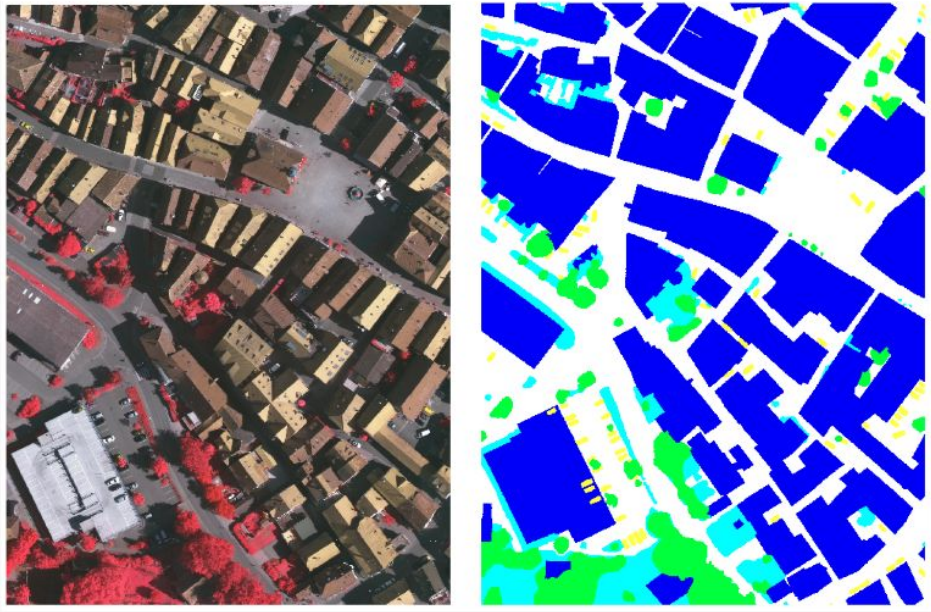
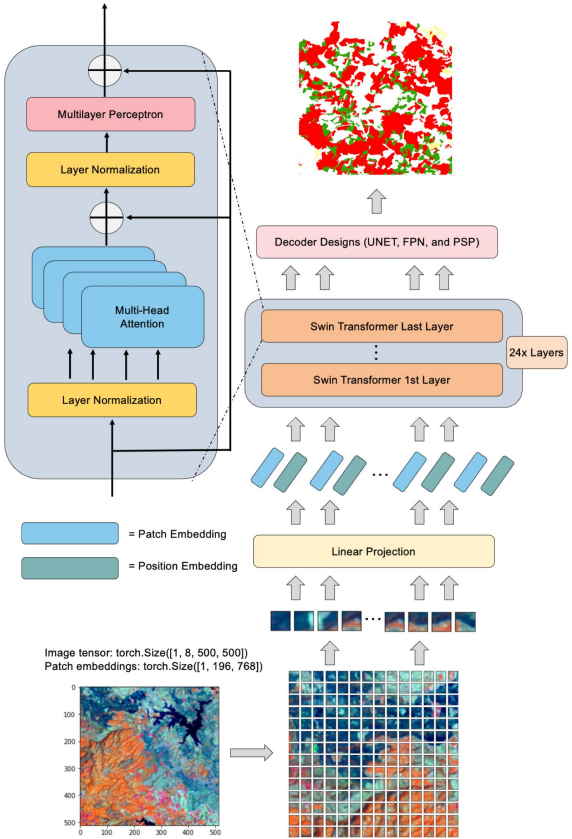
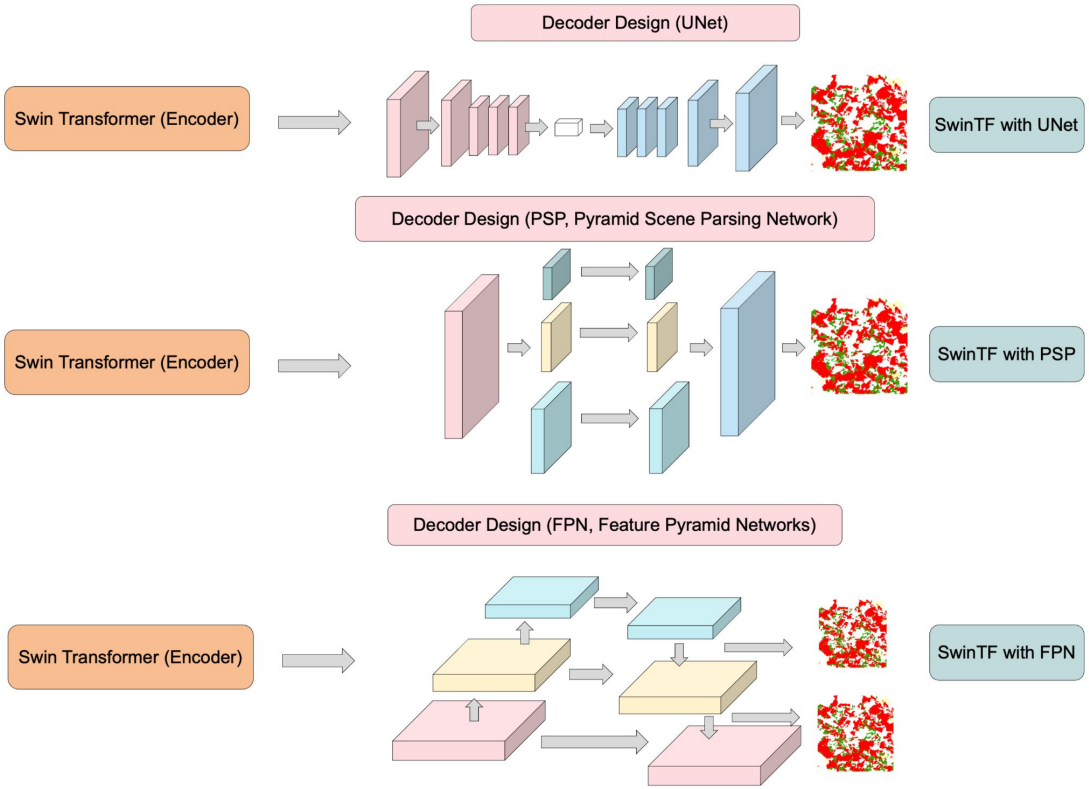


Figure 8. The sample input tile from Figure 7 (**left**) and corresponding ground truth (**right**). The label of the Vaihingen Challenge includes six categories: impervious surface (imp surf, white), building (blue), low vegetation (low veg, cyan), tree (green), car (yellow), and clutter/background (red).

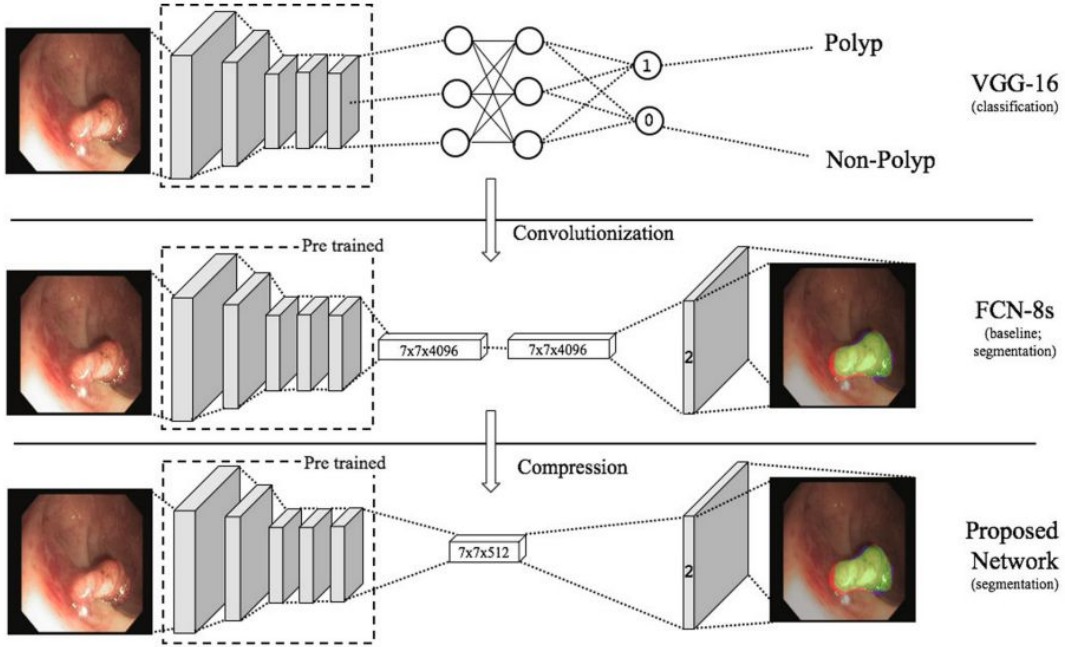


[3] Panboonyuen, Teerapong, et al. **"Transformer-based decoder designs for semantic segmentation on remotely sensed images."** Remote Sensing 13.24 (2021): 5100.





[4] Wichakam, I., **Panboonyuen, T.,** Udomcharoenchaikit, C., & Vateekul, P. (2018). **Real-time polyps segmentation for colonoscopy video frames using compressed fully convolutional network.** In MultiMedia Modeling: 24th International Conference, MMM 2018, Bangkok, Thailand, February 5-7, 2018, Proceedings, Part I 24 (pp. 393-404). Springer International Publishing.



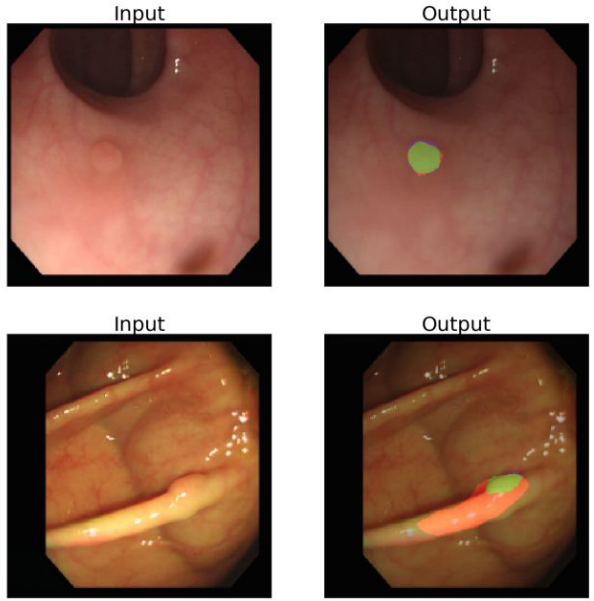
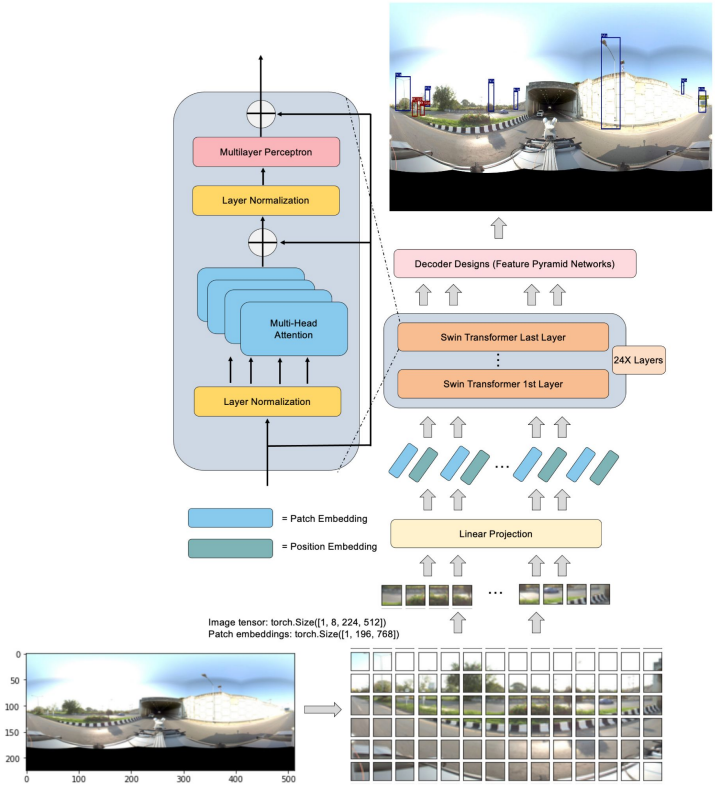
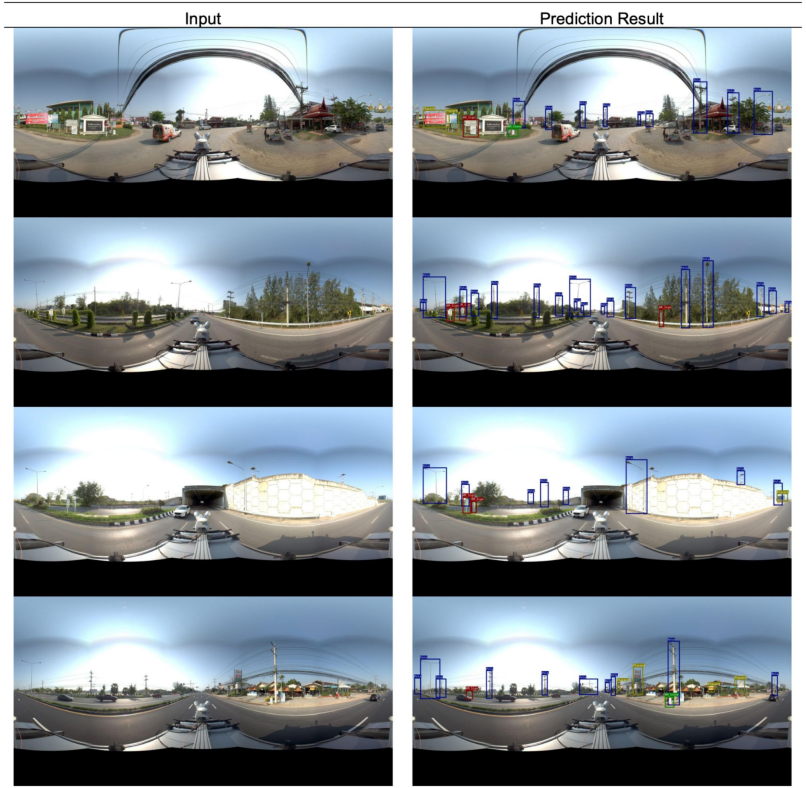


Fig. 1. Overview of our compressed network which is compressed from the original FCN-8s [8] based on VGG-16 [4] architecture.

[5] Panboonyuen, Teerapong, et al. **"Object detection of road assets using transformer-based YOLOX with feature pyramid decoder on thai highway panorama."** Information 13.1 (2022): 5.





[6] Thitisiriwech, K., **Panboonyuen, T.**, Kantavat, P., Iwahori, Y., & Kijsirikul, B. (2022). **The Bangkok Urbanscapes Dataset for Semantic Urban Scene Understanding Using Enhanced Encoder-Decoder With Atrous Depthwise Separable A1 Convolutional Neural Networks.** IEEE Access, 10, 59327-59349.

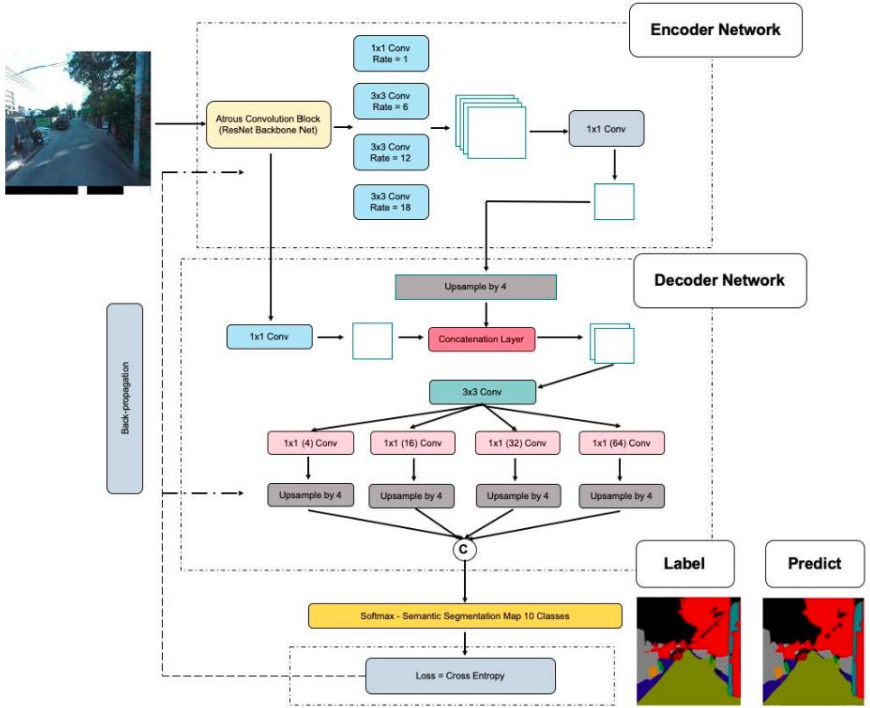


FIGURE 6. An overview of enhanced DeepLab-V3+ (Encoder-Decoder with atrous separable convolutional for semantic segmentation [15]) with ResNet-101 backbone [41] (DeepLab-V3-A1-ResNet-101).

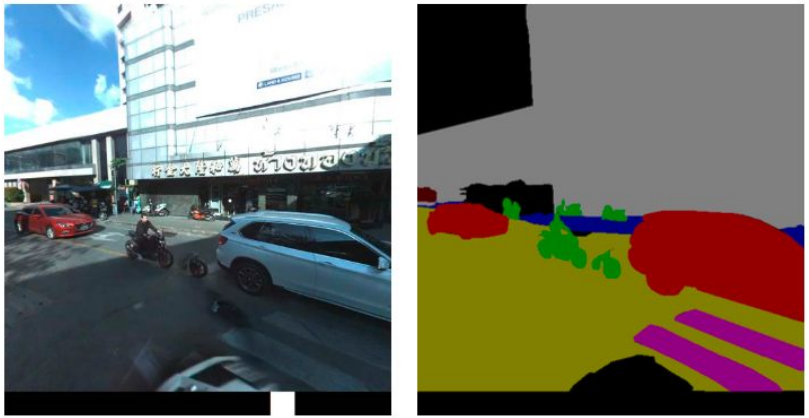


FIGURE 8. Sample 1: The example of Sukhumvit's large road from the training set of the Bangkok Urbanscapes dataset. The input image is shown in (a), and the ground truth is shown in (b).

FIGURE 3. The semantic color codes of the CamVid dataset. Each color is encoded with respect to the semantic class in the ground truth images.

[6] Vajeethaveesin, T., **Panboonyuen, T.**, Lawawironjwong, S., Srestasathiern, P., Jaiyen, S., & Jitkajornwanich, K. (2022). **A performance comparison between GIS-based and neuron network methods for flood susceptibility assessment in ayutthaya province.** Trends in Sciences, 19(2), 2038-2038.

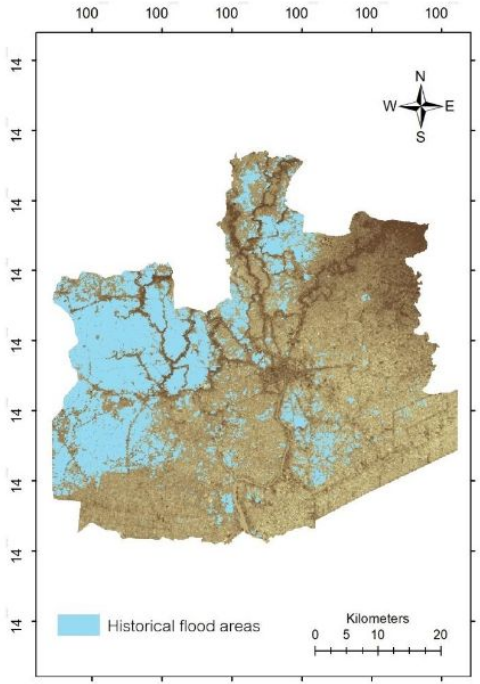


Figure 2 The historical flood zone shows flood events, which were captured in 2016 from GISTDA. Most of the flooding areas are located in the west of the study area.

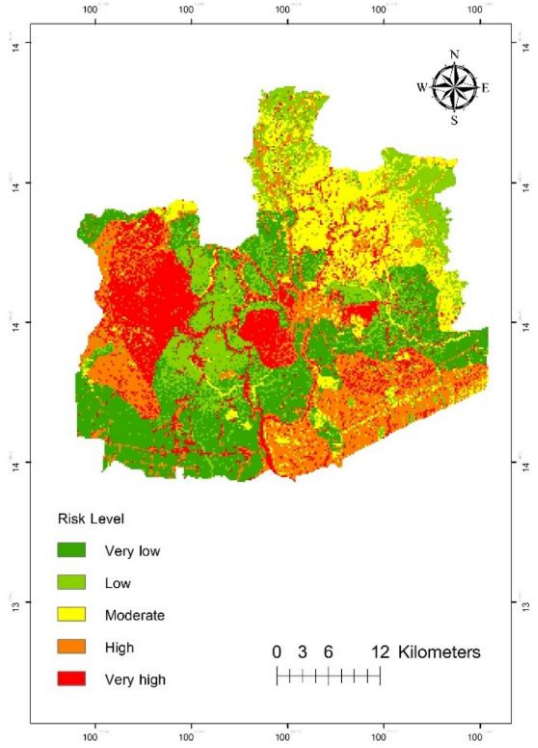
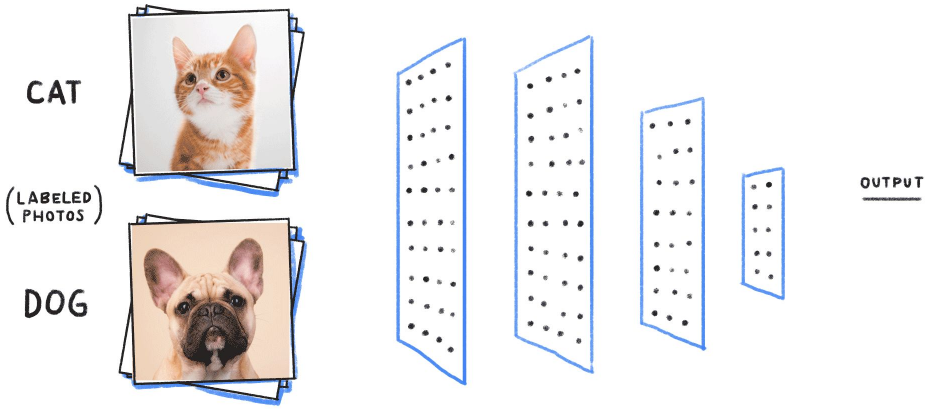


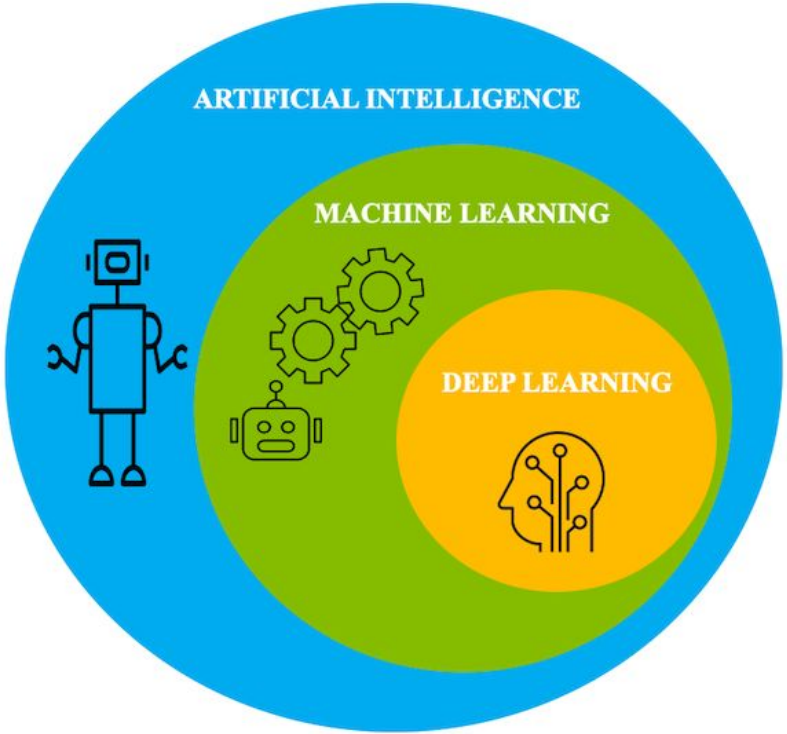
Figure 4 Flood susceptibility map from the flood risk assessment model (FRAM).

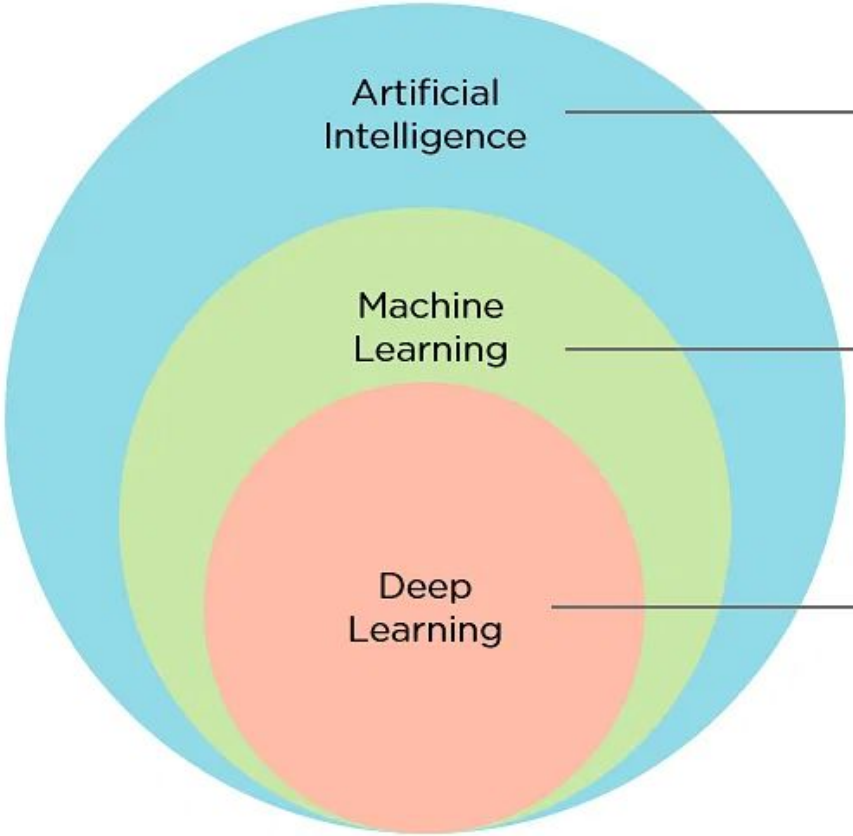
Basic AI



AI vs Machine Learning vs Deep Learning

- **Artificial Intelligence (AI)** is the concept of creating smart intelligent machines.
- **Machine Learning (ML)** is a subset of artificial intelligence that helps you build AI-driven applications.
- **Deep Learning** is a subset of machine learning that uses vast volumes of data and complex algorithms to train a model.





Ability of a machine to imitate intelligent human behavior

Application of AI that allows a system to automatically learn and improve from experience

Application of Machine Learning that uses complex algorithms and deep neural nets to train a model

Applications of Artificial Intelligence

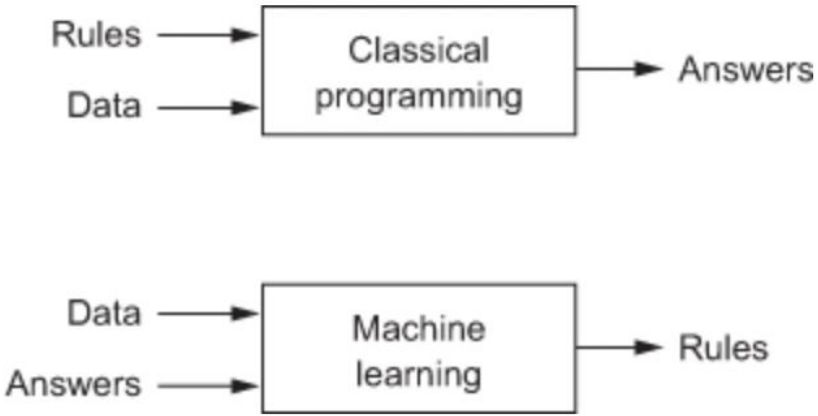
- Machine Translation such as Google Translate
- Self Driving Vehicles such as Google's Waymo
- AI Robots such as Sophia and Aibo
- Speech Recognition applications like Apple's Siri or OK Google

https://www.youtube.com/watch?v=kJoAcEI2PXQ&t=14s

What is Deep Learning?

Deep Learning is a subset of Machine Learning that uses mathematical functions to map the input to the output.

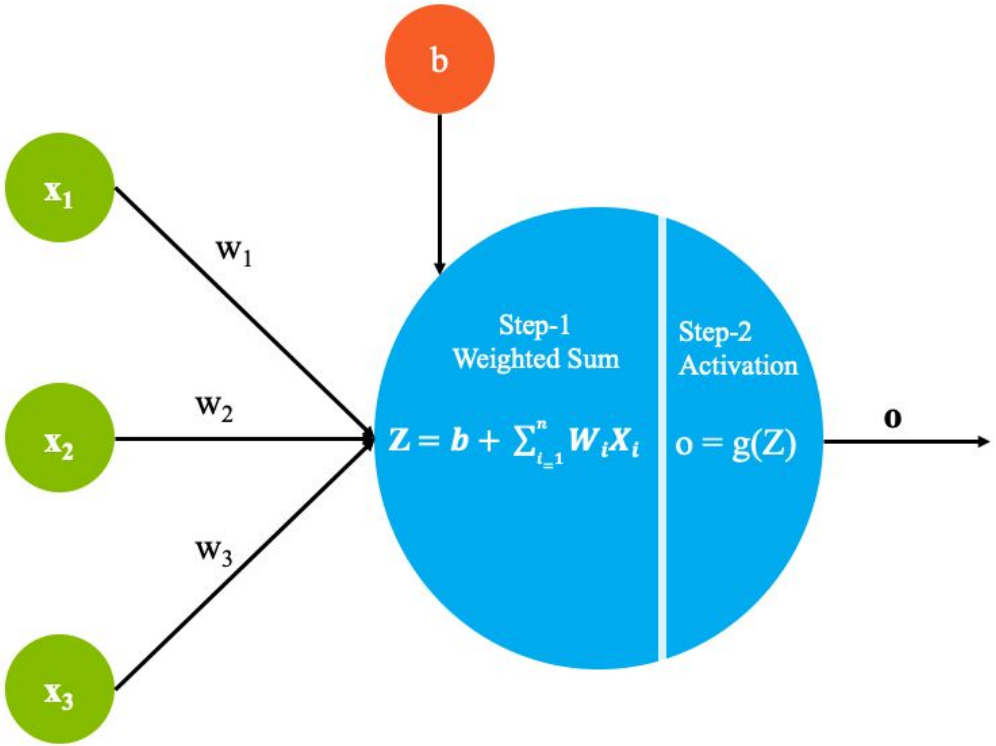
These functions can extract non-redundant information or patterns from the data, which enables them to form a relationship between the input and the output.



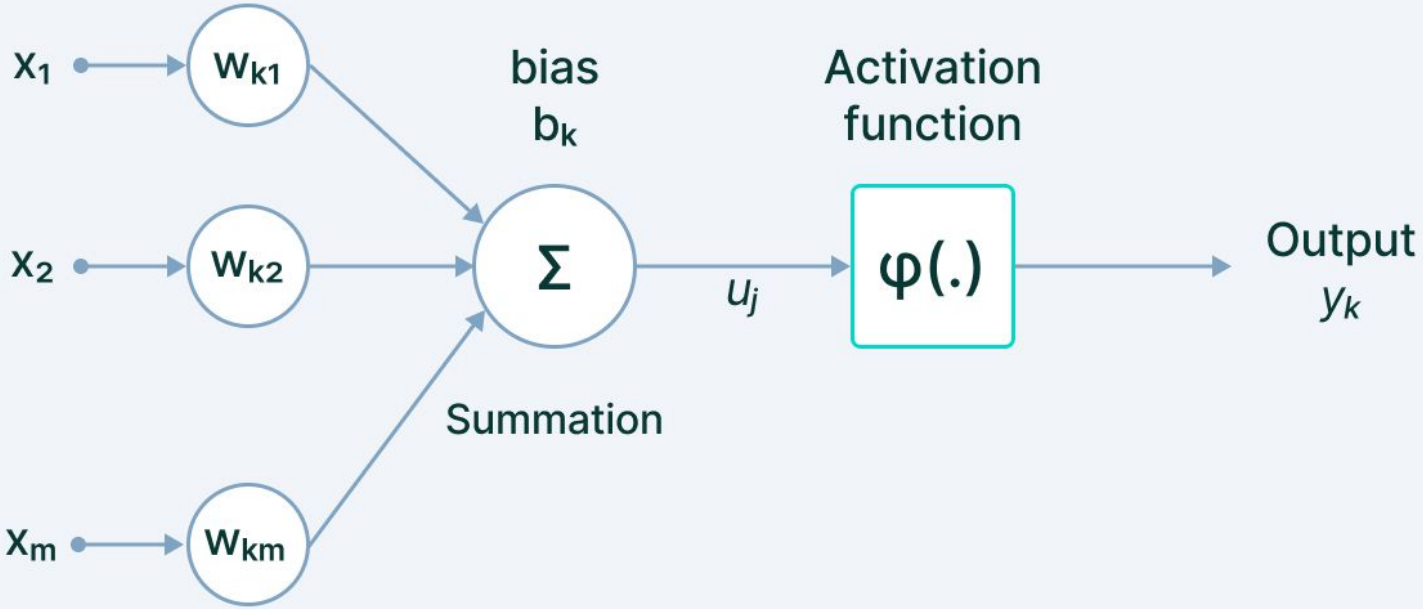
Artificial Neural Network (ANN)

This section provides an overview of the architecture behind deep learning, artificial neural networks (ANN), and discusses some of the key terminology.

As shown in the following figure, each perceptron is made up of the following parts:

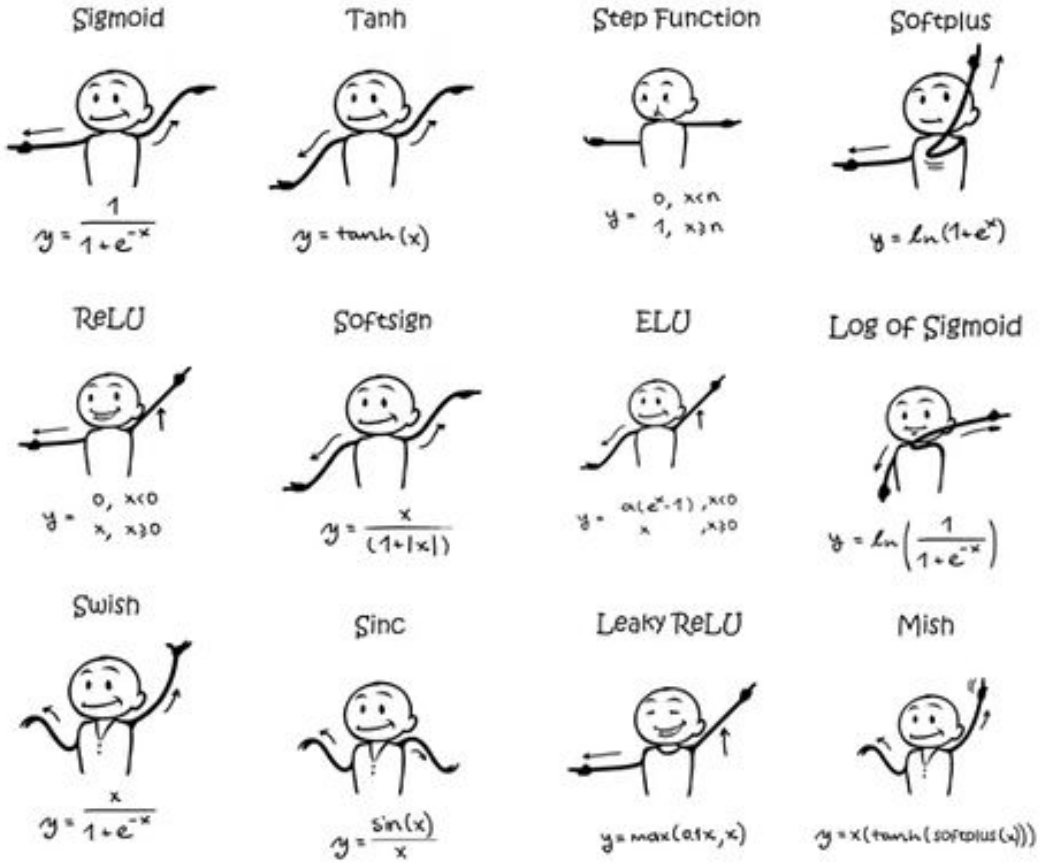


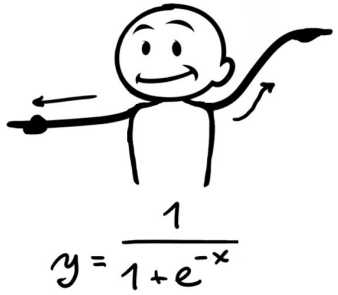
Neuron

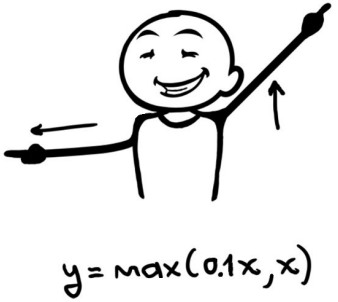


V7 Labs

https://www.theaidream.com/post/an-overview-of-activation-functions-in-deep-learning





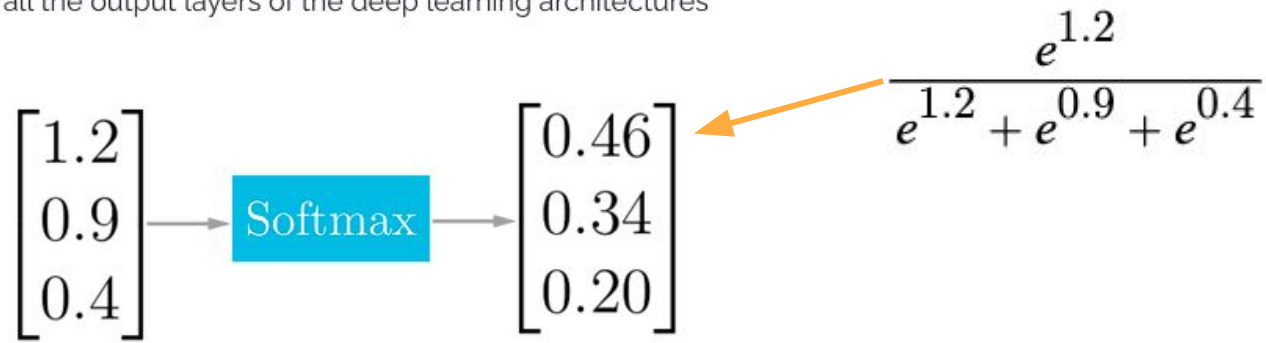


Softmax Function

The softmax function is used to compute probability distribution from a vector of real numbers. The Softmax function produces an output which is a range of values between 0 and 1, with the sum of the probabilities being equal to 1. The Softmax function is computed using the relationship:

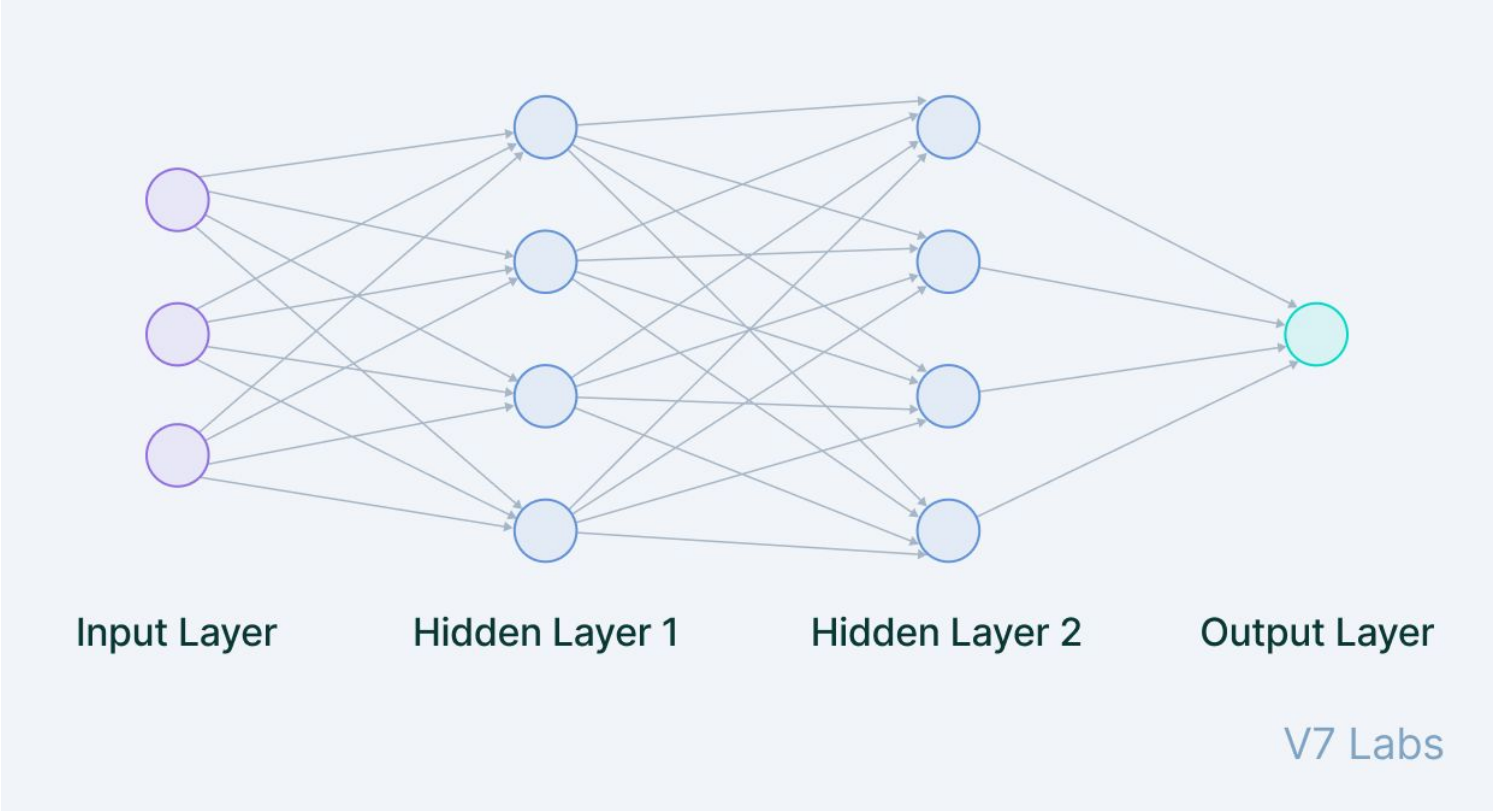
$$f(x_i) = \frac{exp\ (x_i)}{\sum_j exp\ (x_j)}$$

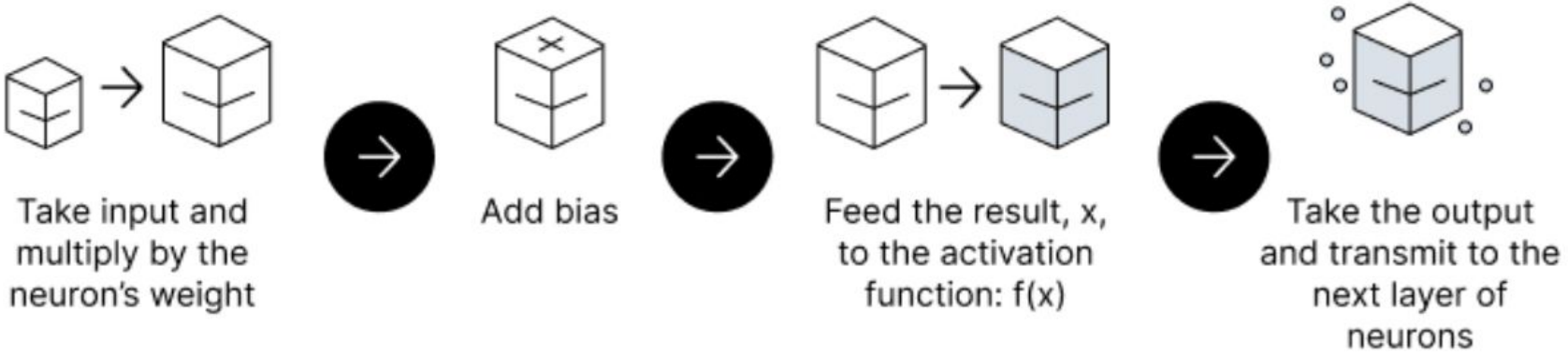
The Softmax function is used in multi-class models where it returns probabilities of each class, with the target class having the highest probability. The Softmax function mostly appears in almost all the output layers of the deep learning architectures



The softmax function squashes the outputs of each unit to be between 0 and 1, just like a sigmoid. It also divides each output such that the total sum of the outputs is equal to 1.
Credits

How does Deep Learning work?





Forward propagation in neural networks

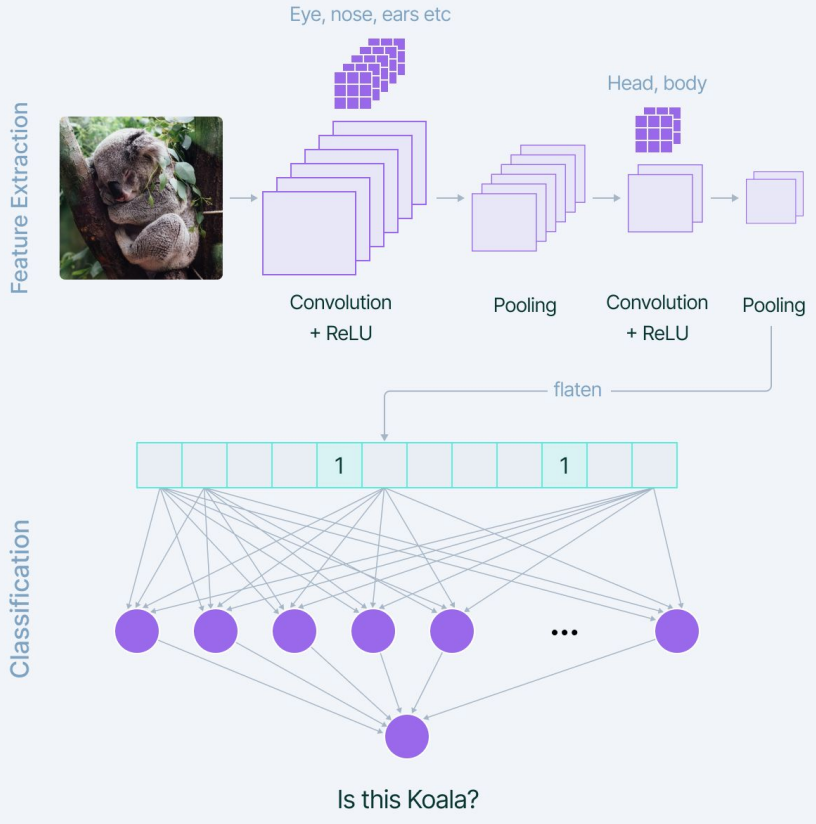
The Convolutional Neural Networks or CNNs

The Convolutional Neural Networks or CNNs are primarily used for tasks related to computer vision or image processing.

CNNs are extremely good in modeling spatial data such as 2D or 3D images and videos.

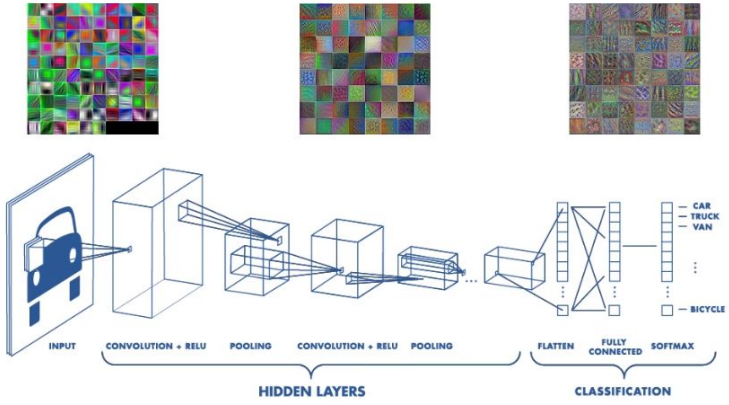
They can extract features and patterns within an image, enabling tasks such as image classification or object detection.

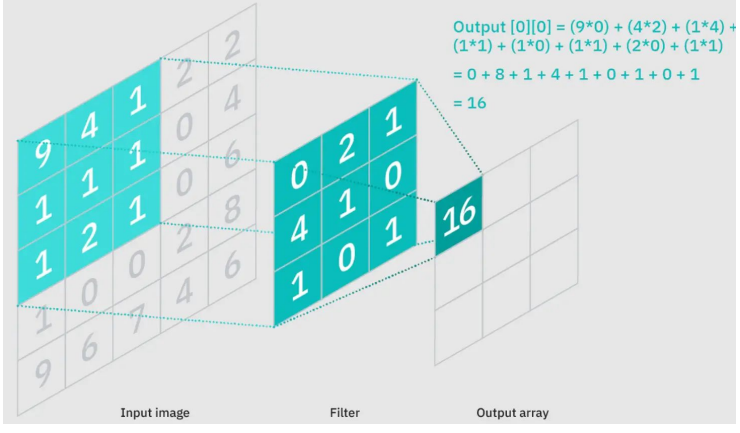
Convolutional Neural Networks



V7 Labs

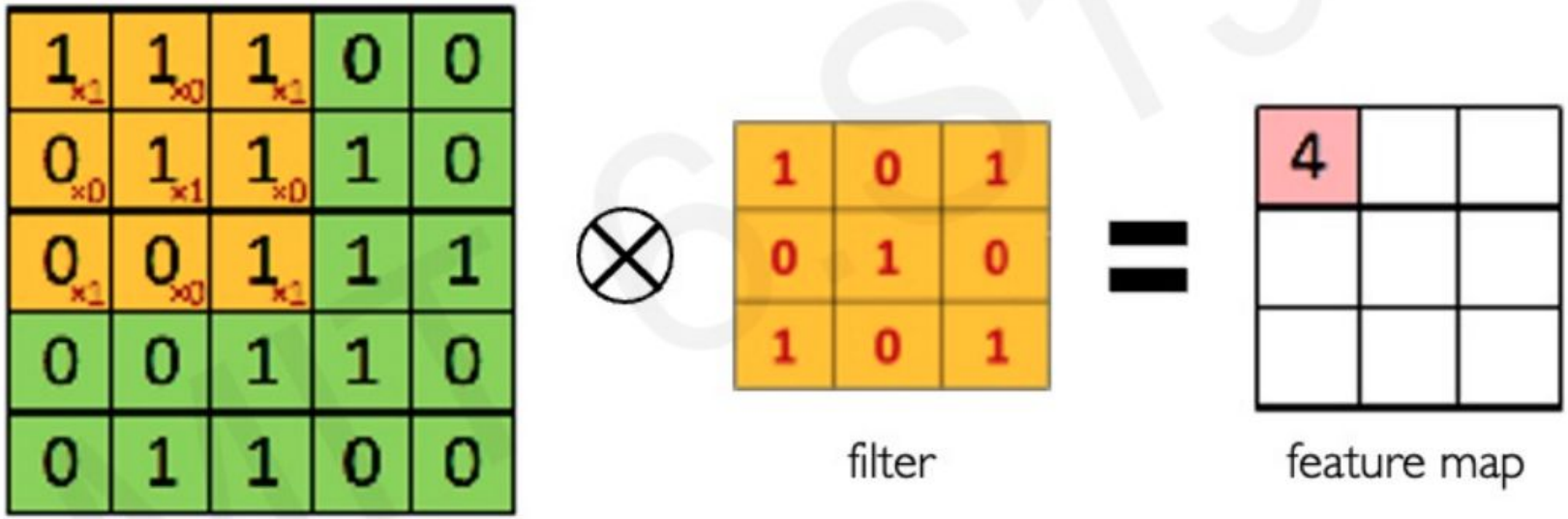
Convolutional Neural Network

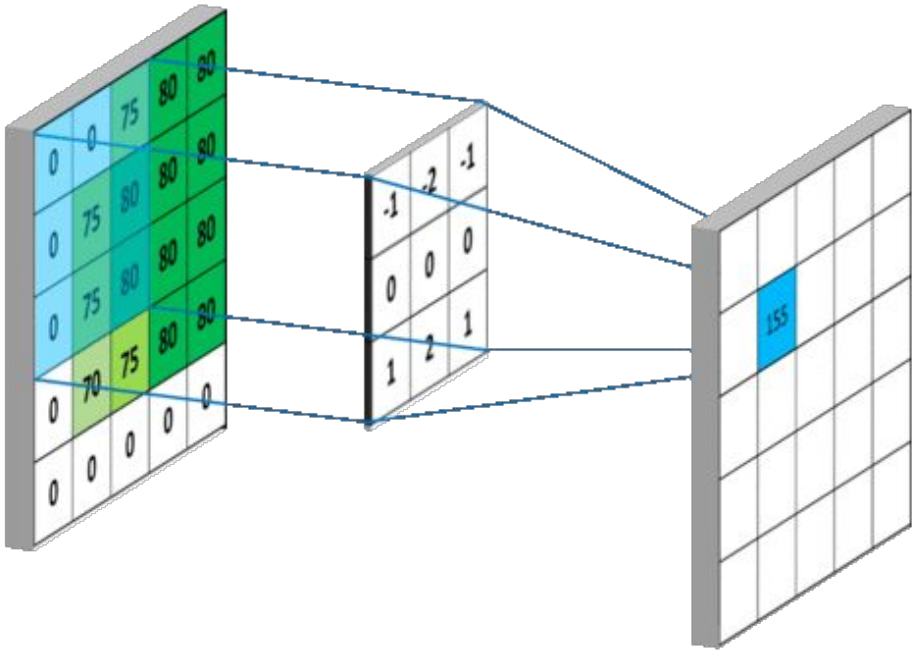




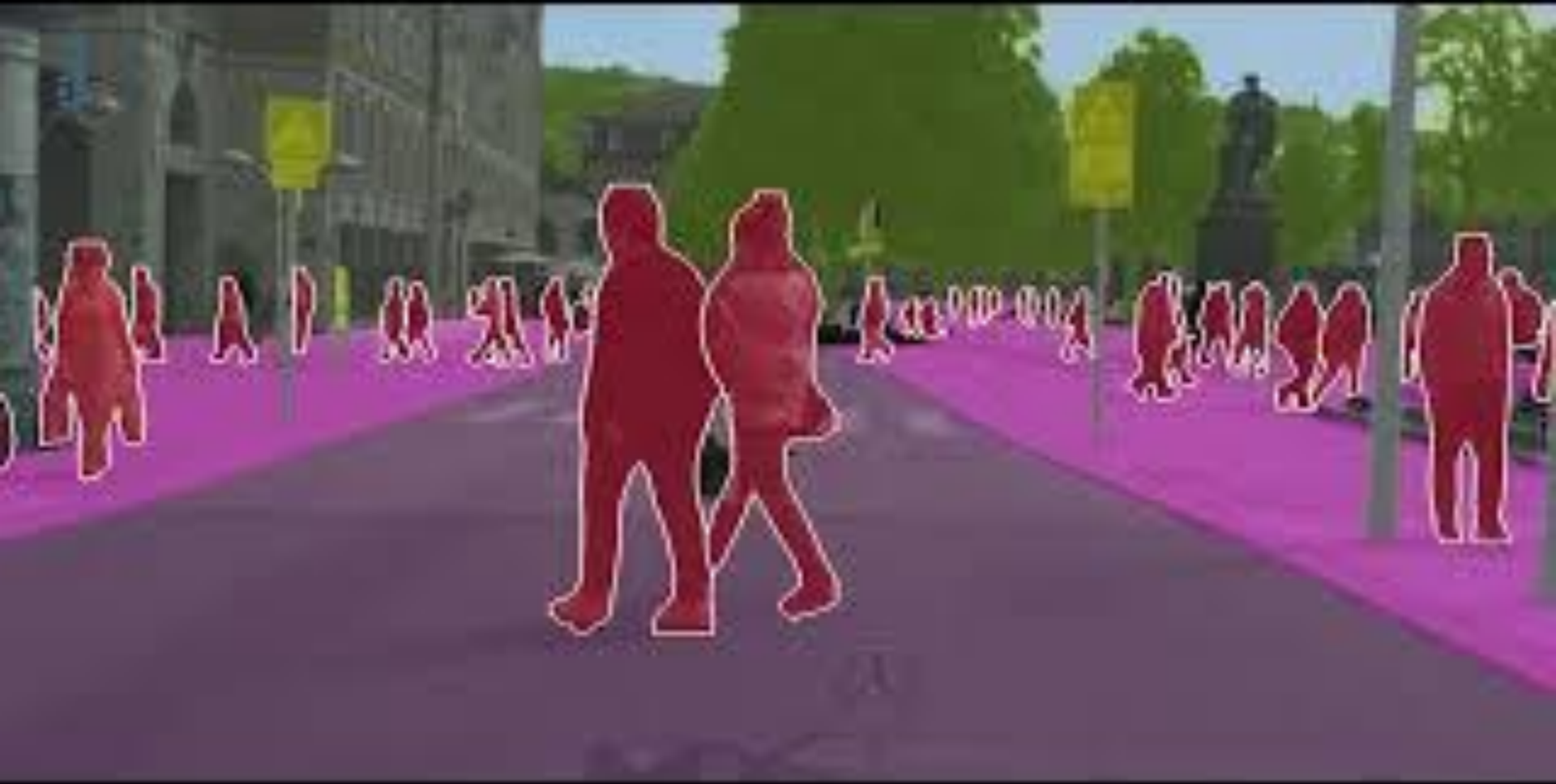
The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:

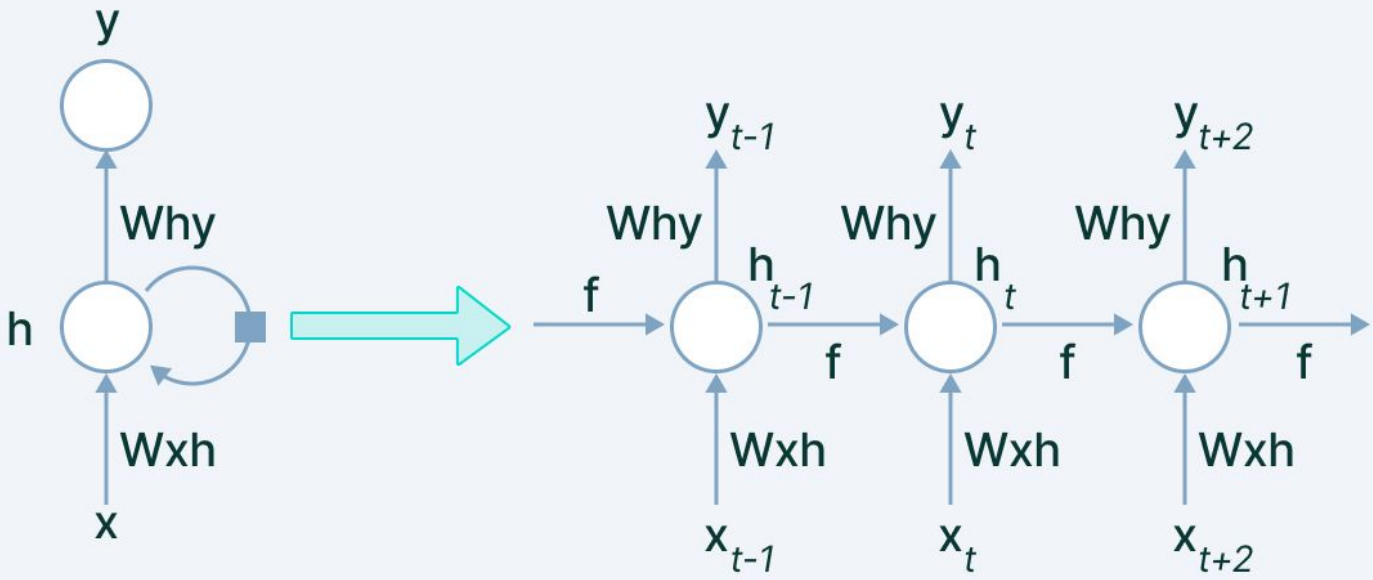



0 0 75 80 80
0 75 80 80 80
0 75 80 80 80
0 70 75 80 80
0 0 0 0 0
-1 -2 -1
0 0 0
1 2 1
155

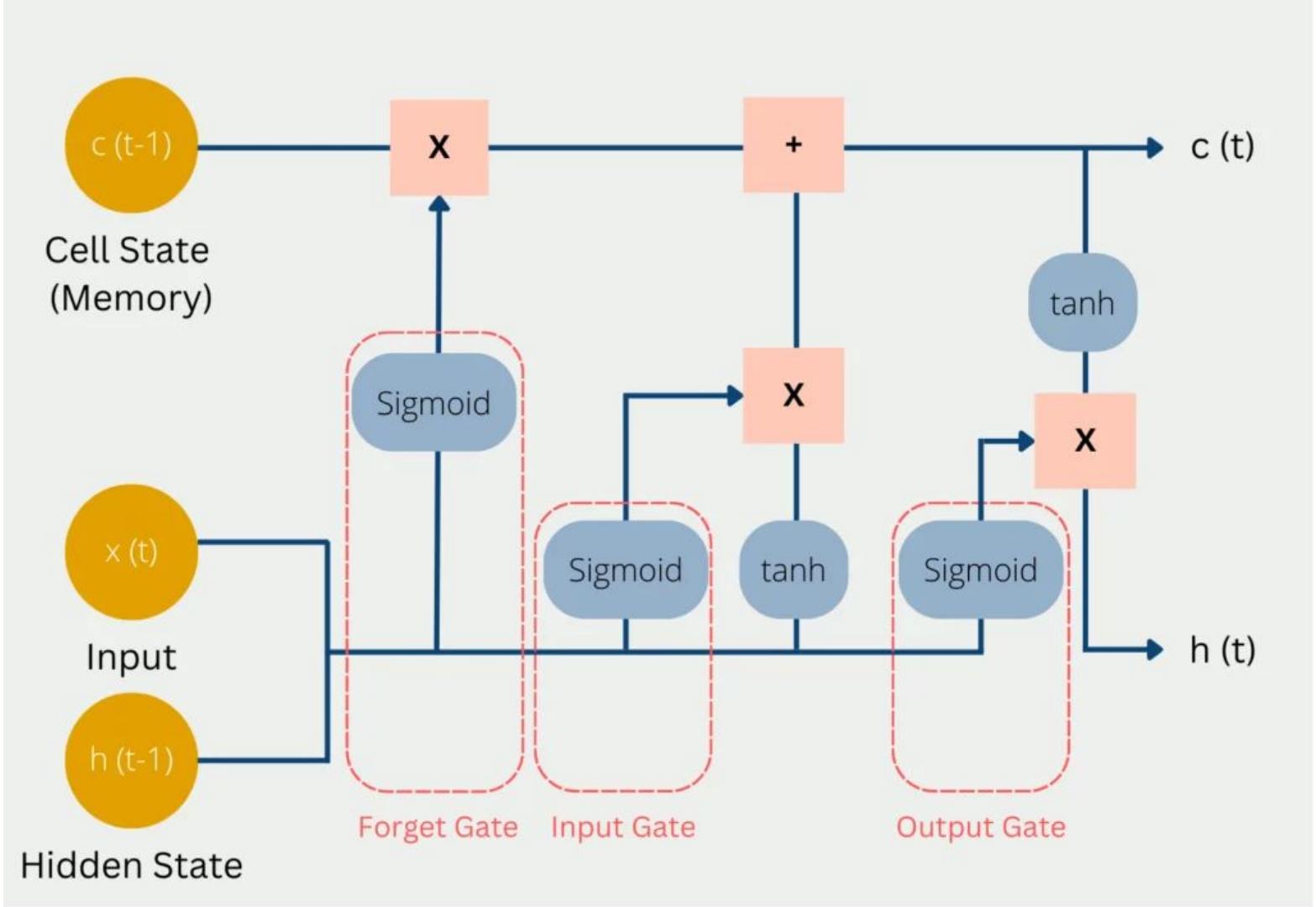

WORLD'S



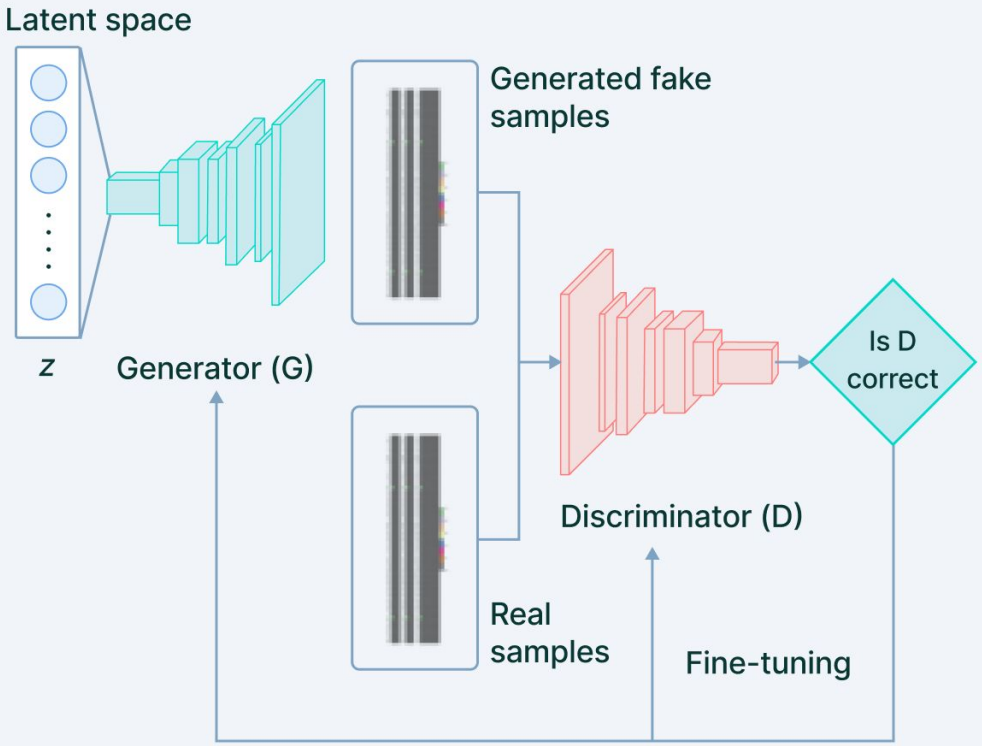
The Recurrent Neural Networks (RNN)

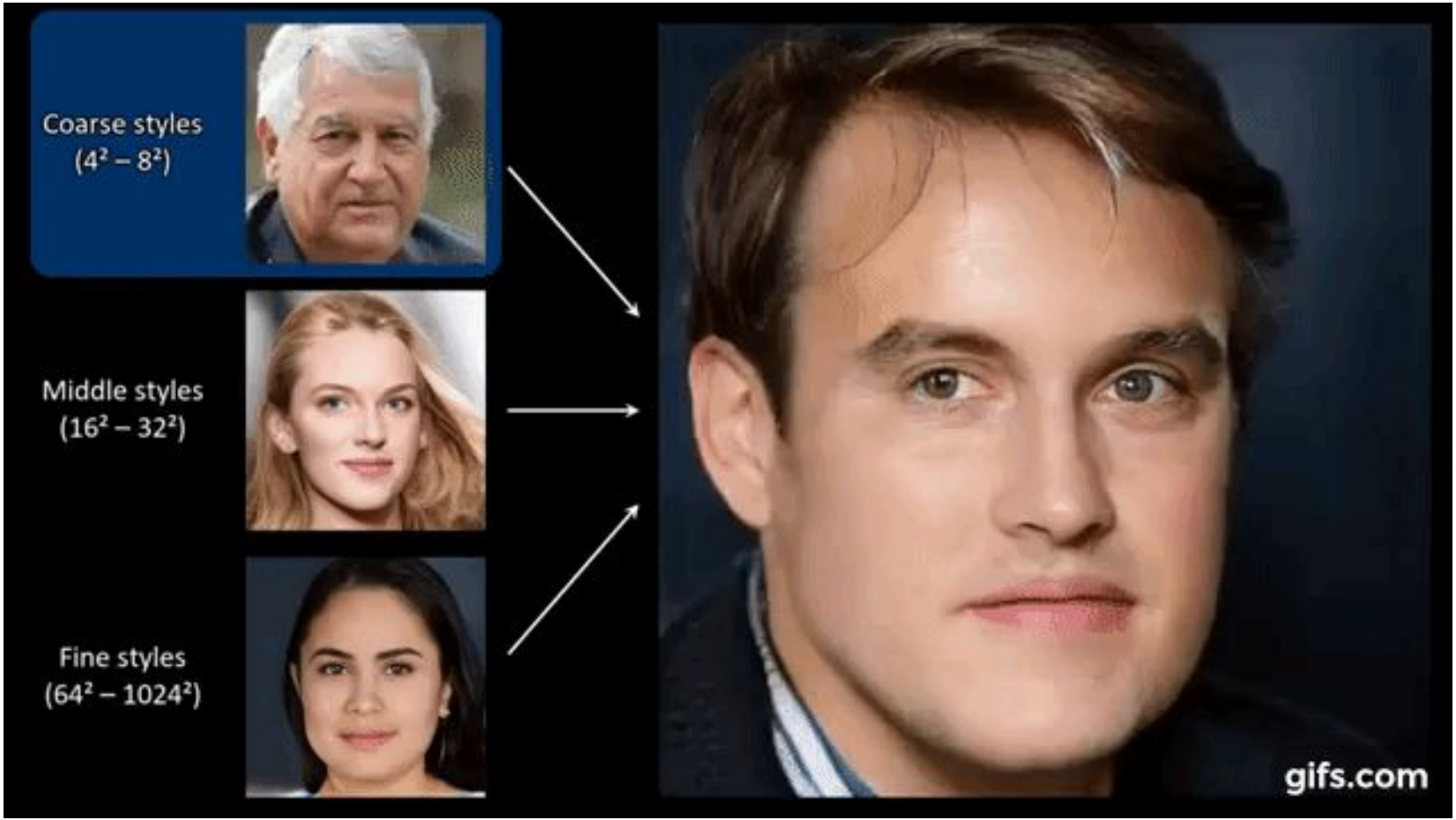


V7 Labs


c (t-1)
Cell State
(Memory)
x
+
c (t)
tanh
Sigmoid
x
x
x (t)
Input
Sigmoid
tanh
Sigmoid
h (t)
h (t-1)
Hidden State
Forget Gate
Input Gate
Output Gate

Generative Adversarial Networks




Coarse styles
(4² – 8²)
Middle styles
(16² – 32²)
Fine styles
(64² – 1024²)
gifs.com


DIEP
NEP.
THIS IS NOT MORGAN FREEMAN.

DALL-E 2 AI IMAGE GENERATORS

TEXT PROMPT a teapot in the shape of an avocado. a teapot imitating an avocado.

AI-GENERATED IMAGES

DALL·E mini

DALL·E mini is an AI model that generates images from any prompt you give!

voldemort as a cat person

Run

▶ Bias and Limitations

DALL·E mini

AI model generating images from any prompt!

lord voldemort as the baby on the cover of nevermind by nirvana

Run

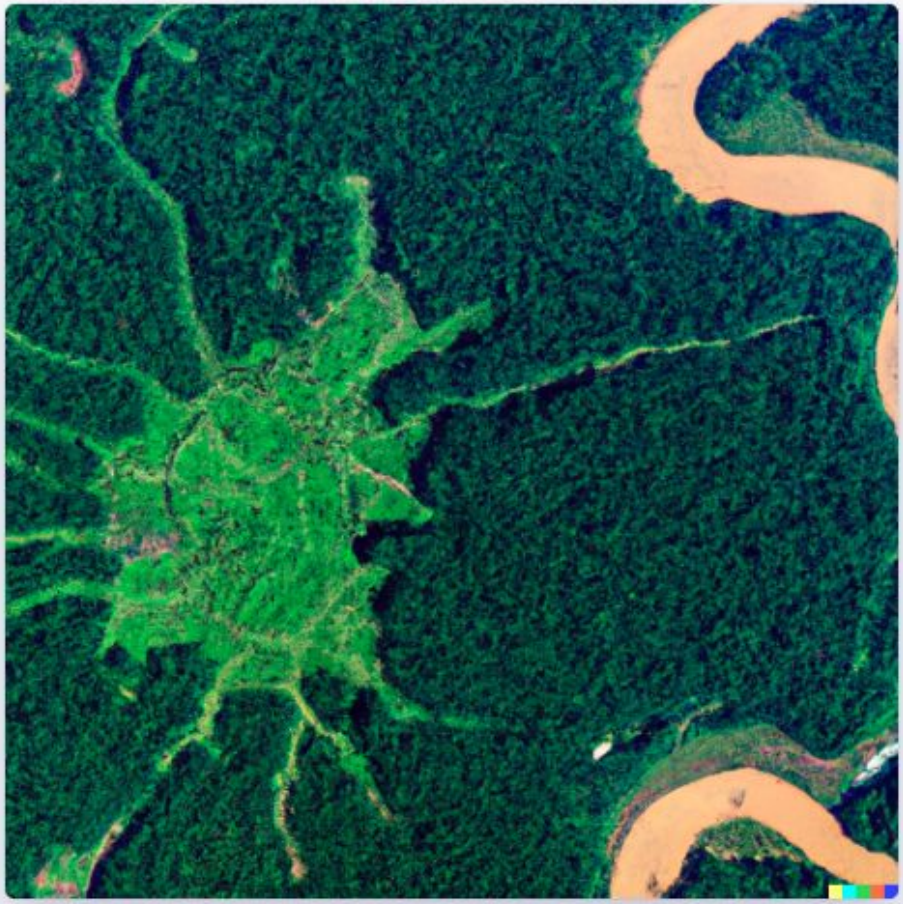
DALL·E mini

AI model generating images from any prompt!

Thanos eating pizza

Run

https://labs.openai.com/s/q74WPnaXW3mdlsfc498zyGma

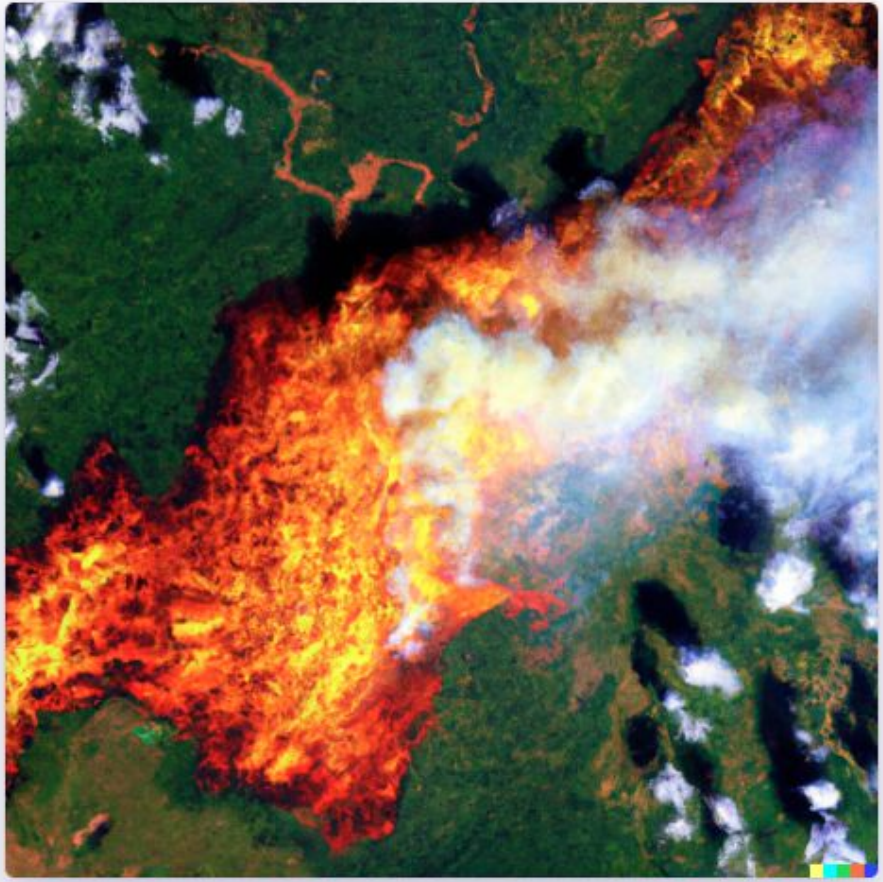


Created with **DALL·E**, an AI system by OpenAI

"A satellite image showing Amazon rainforest after deforestation"

Vinícius × DALL·E

Human & AI



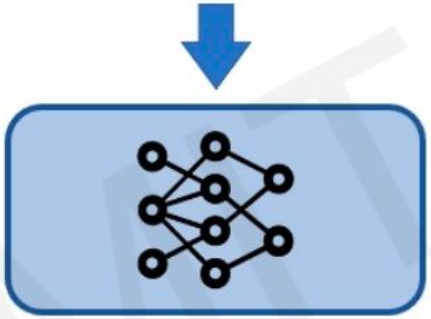
Created with DALL·E, an AI system by OpenAI

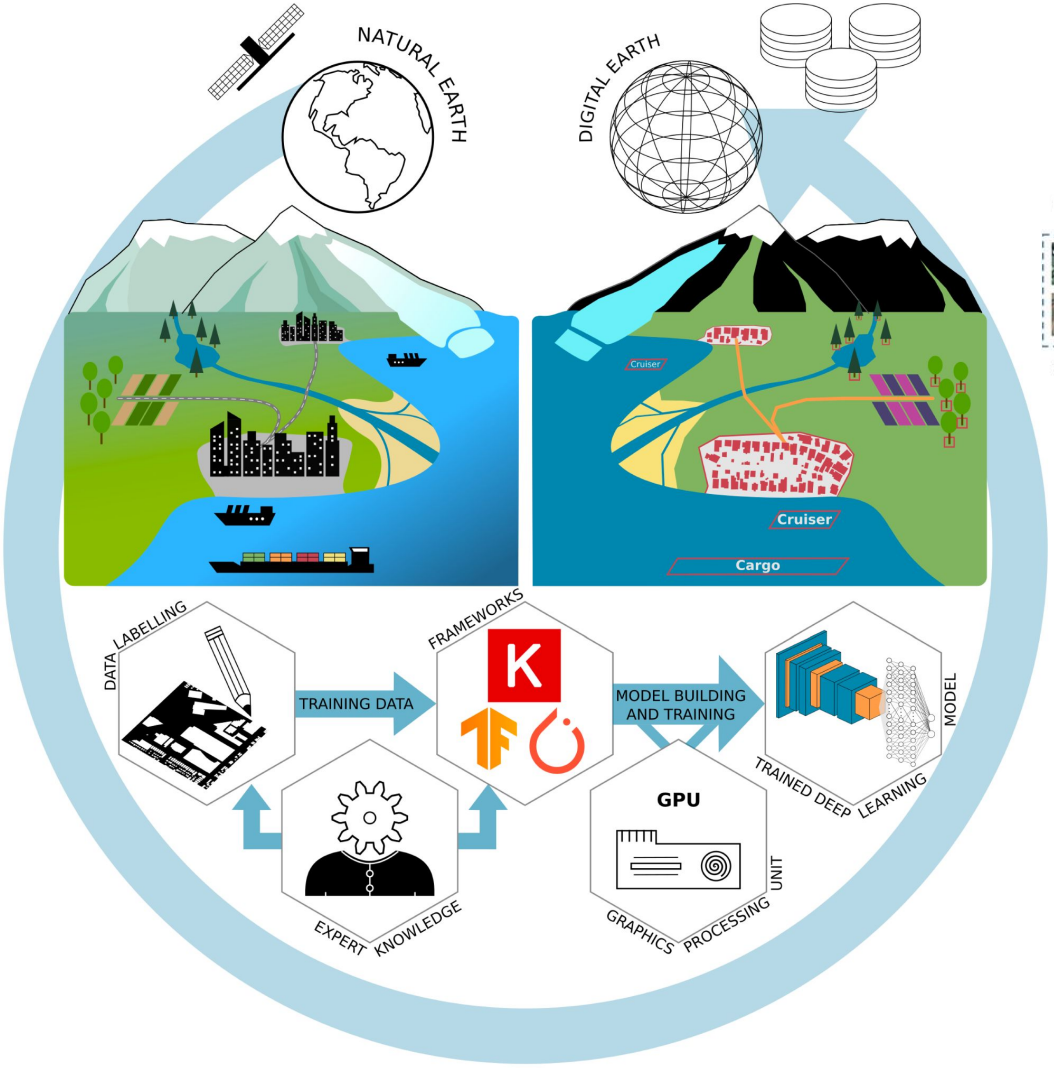
“A satellite image showing Amazon rainforest under fire”

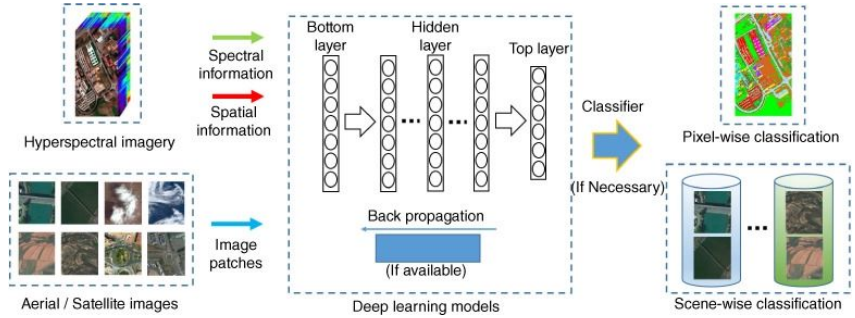
Vinícius × DALL·E
Human & AI

Generating Images from Natural Language

"A photo of an astronaut riding a horse."







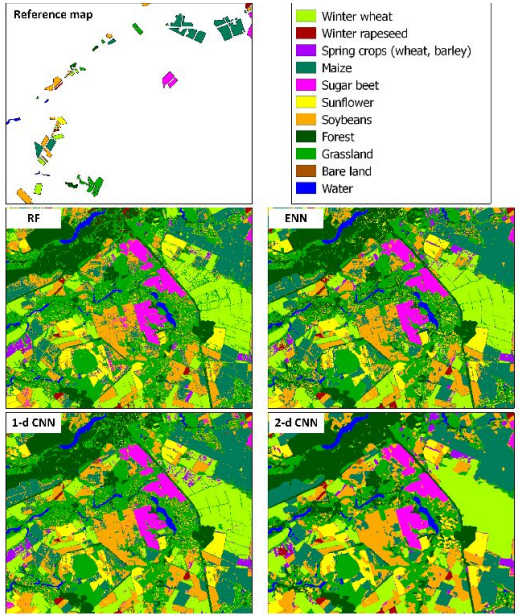
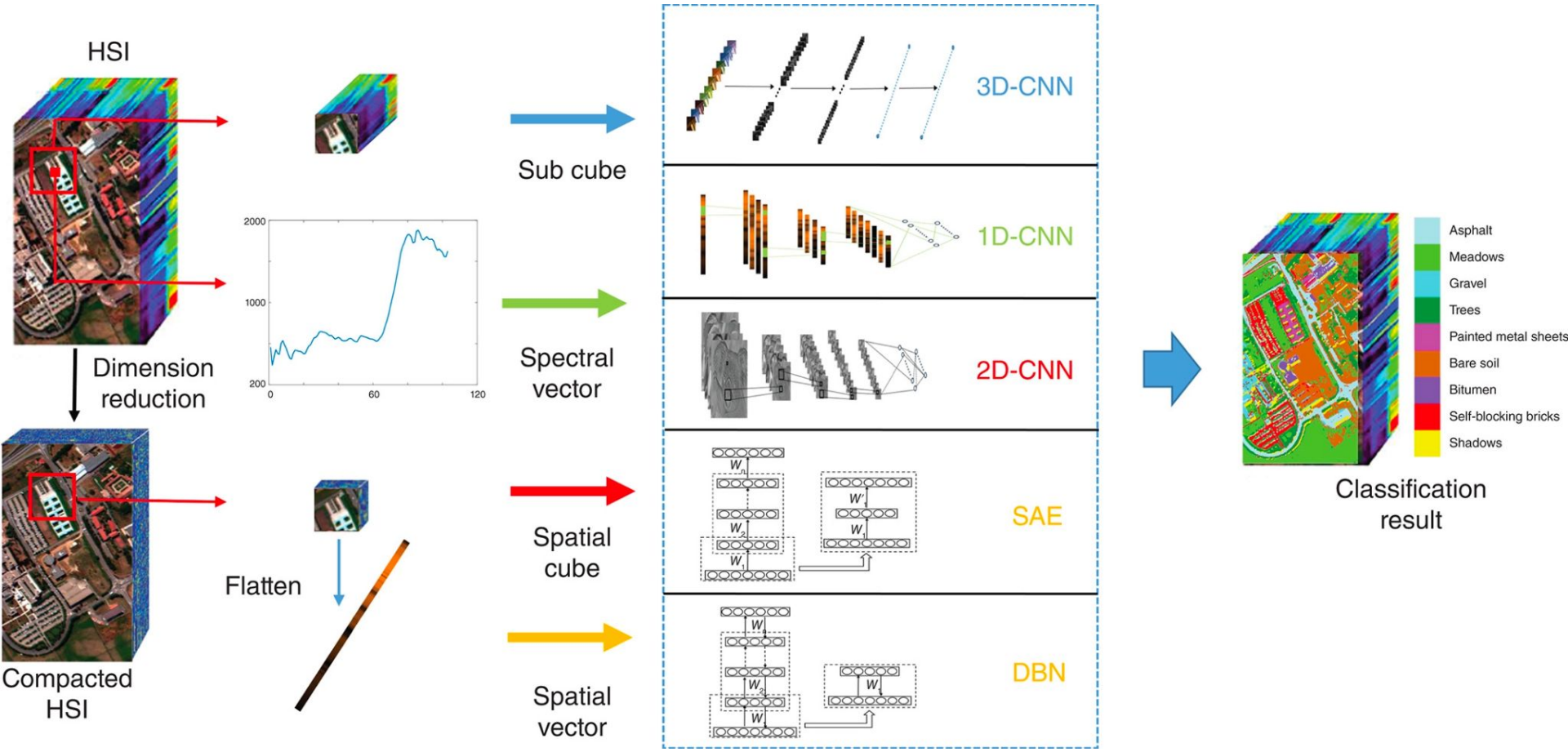


Fig. 3. Example of classification result for the Kyiv region for 2015 based


HSI
Dimension reduction
Compacted HSI
Flatten
Sub cube
Spectral vector
Spatial cube
Spatial vector
3D-CNN
1D-CNN
2D-CNN
SAE
DBN
Classification result
Asphalt
Meadows
Gravel
Trees
Painted metal sheets
Bare soil
Bitumen
Self-blocking bricks
Shadows

Why Now?

Neural Networks date back decades, so why the resurgence?

1. Big Data

- Larger Datasets
- Easier Collection & Storage

IMAGENET

2. Hardware

- Graphics Processing Units (GPUs)
- Massively Parallelizable

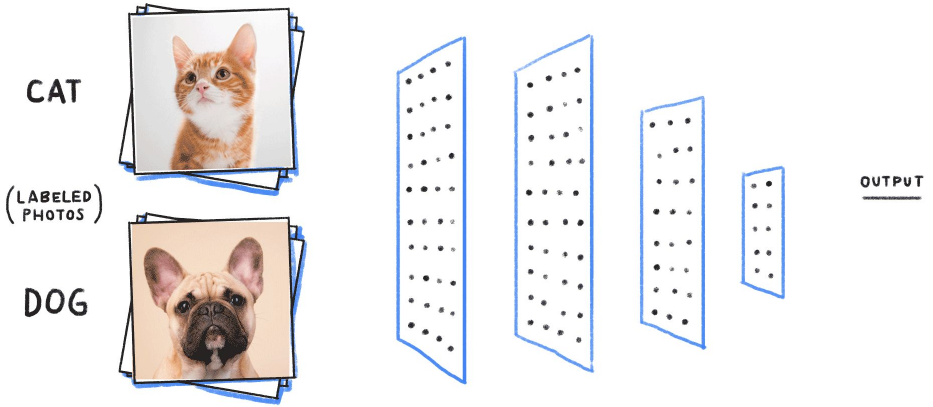
3. Software

- Improved Techniques
- New Models
- Toolboxes

TensorFlow

Credit: https://stanford.edu/~shervine/teaching/

Machine Learning Glossary



Reflex-based models with Machine Learning

By Afshine Amidi and Shervine Amidi

Linear predictors

In this section, we will go through reflex-based models that can improve with experience, by going through samples that have input-output pairs.

❒ **Feature vector** — The feature vector of an input x is denoted $\phi(x)$ and is such that:

$$\boxed{\phi(x) = \begin{bmatrix} \phi_1(x) \\ \vdots \\ \phi_d(x) \end{bmatrix} \in \mathbb{R}^d}$$

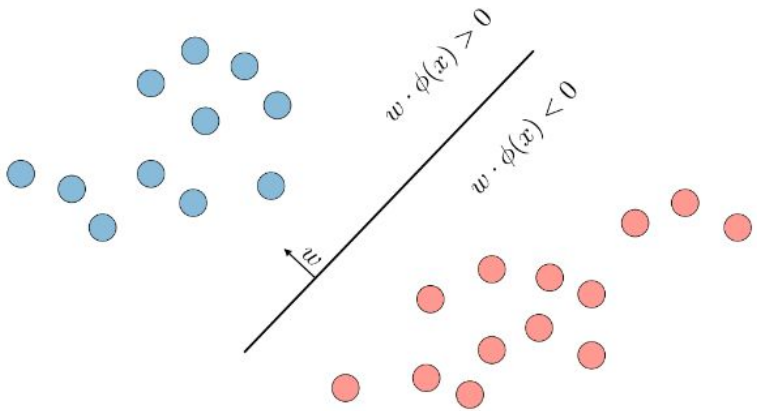
❒ **Score** — The score $s(x, w)$ of an example $(\phi(x), y) \in \mathbb{R}^d \times \mathbb{R}$ associated to a linear model of weights $w \in \mathbb{R}^d$ is given by the inner product:

$$\boxed{s(x, w) = w \cdot \phi(x)}$$

Classification

❒ **Linear classifier** — Given a weight vector $w \in \mathbb{R}^d$ and a feature vector $\phi(x) \in \mathbb{R}^d$, the binary linear classifier f_w is given by:

$$f_w(x) = \text{sign}(s(x, w)) = \begin{cases} +1 & \text{if } w \cdot \phi(x) > 0 \\ -1 & \text{if } w \cdot \phi(x) < 0 \\ ? & \text{if } w \cdot \phi(x) = 0 \end{cases}$$



❒ **Margin** — The margin $m(x, y, w) \in \mathbb{R}$ of an example $(\phi(x), y) \in \mathbb{R}^d \times \{-1, +1\}$ associated to a linear model of weights $w \in \mathbb{R}^d$ quantifies the confidence of the prediction: larger values are better. It is given by:

$$m(x, y, w) = s(x, w) \times y$$

Regression

❒ **Linear regression** — Given a weight vector $w \in \mathbb{R}^d$ and a feature vector $\phi(x) \in \mathbb{R}^d$, the output of a linear regression of weights w denoted as f_w is given by:

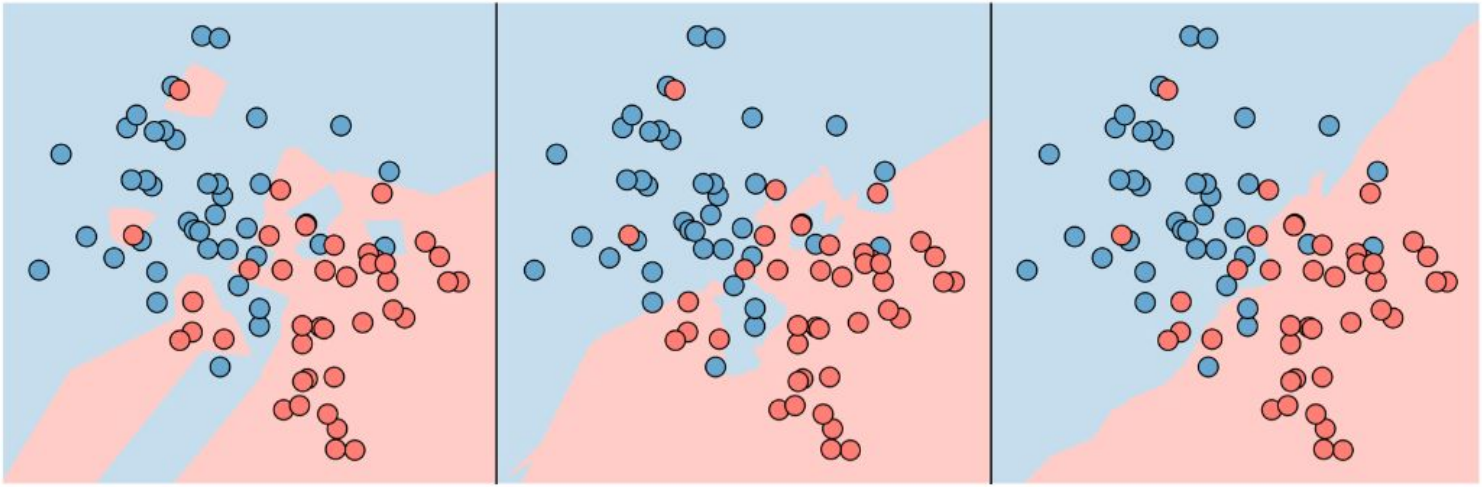
$$\boxed{f_w(x) = s(x, w)}$$

❒ **Residual** — The residual $\mathrm{res}(x, y, w) \in \mathbb{R}$ is defined as being the amount by which the prediction $f_w(x)$ overshoots the target y:

$$\boxed{\mathrm{res}(x, y, w) = f_w(x) - y}$$

Non-linear predictors

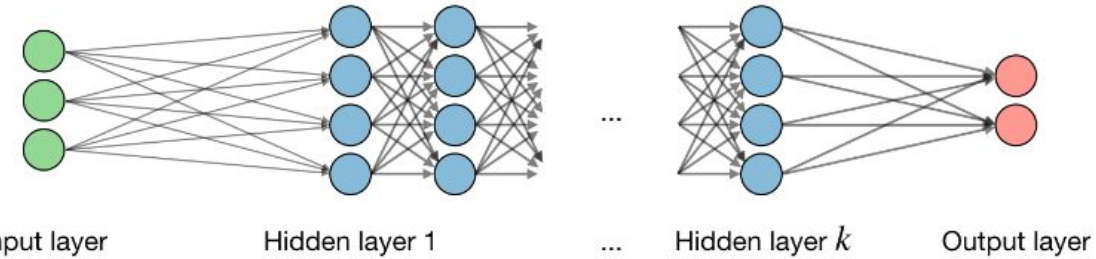
❒ **k-nearest neighbors** — The k-nearest neighbors algorithm, commonly known as k-NN, is a non-parametric approach where the response of a data point is determined by the nature of its k neighbors from the training set. It can be used in both classification and regression settings.



$k = 1$ $k = 3$ $k = 11$

Remark: the higher the parameter k, the higher the bias, and the lower the parameter k, the higher the variance.

❐ **Neural networks** — Neural networks are a class of models that are built with layers. Commonly used types of neural networks include convolutional and recurrent neural networks. The vocabulary around neural networks architectures is described in the figure below:



By noting i the i^{th} layer of the network and j the j^{th} hidden unit of the layer, we have:

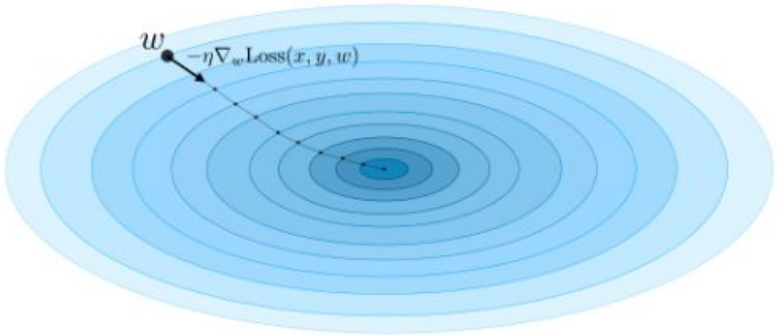
$$\boxed{z_j^{[i]} = {w_j^{[i]}}^T x + b_j^{[i]}}$$

where we note w, b, x, z the weight, bias, input and non-activated output of the neuron respectively.

Stochastic gradient descent

❒ **Gradient descent** — By noting $\eta \in \mathbb{R}$ the learning rate (also called step size), the update rule for gradient descent is expressed with the learning rate and the loss function $\text{Loss}(x, y, w)$ as follows:

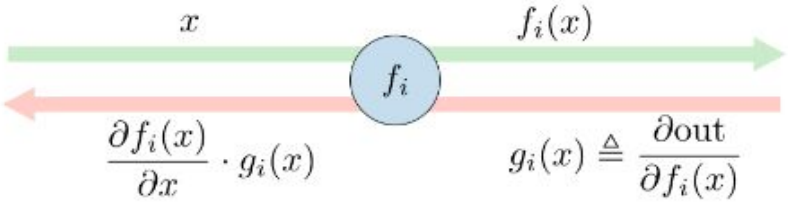
$$\boxed{w \longleftarrow w - \eta \nabla_w \text{Loss}(x, y, w)}$$



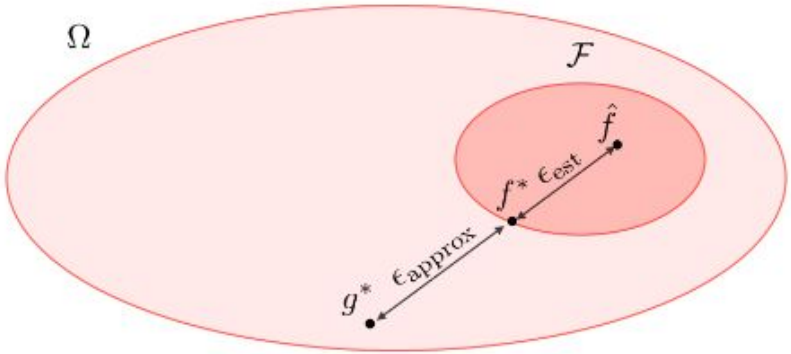
❒ **Stochastic updates** — Stochastic gradient descent (SGD) updates the parameters of the model one training example $(\phi(x), y) \in \mathcal{D}_{\text{train}}$ at a time. This method leads to sometimes noisy, but fast updates.

❒ **Batch updates** — Batch gradient descent (BGD) updates the parameters of the model one batch of examples (e.g. the entire training set) at a time. This method computes stable update directions, at a greater computational cost.

❒ **Backpropagation** — The forward pass is done through f_i, which is the value for the subexpression rooted at i, while the backward pass is done through $g_i = \frac{\partial \text{out}}{\partial f_i}$ and represents how f_i influences the output.



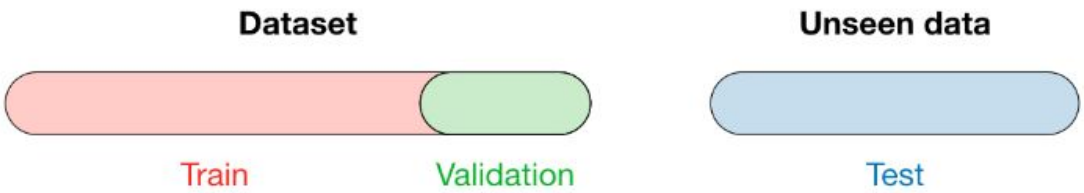
❒ **Approximation and estimation error** — The approximation error ϵ_{approx} represents how far the entire hypothesis class $\mathcal{F}$ is from the target predictor g^*, while the estimation error ϵ_{est} quantifies how good the predictor $\hat{f}$ is with respect to the best predictor f^* of the hypothesis class $\mathcal{F}$.



❒ **Sets vocabulary** — When selecting a model, we distinguish 3 different parts of the data that we have as follows:

Training set	Validation set	Testing set
• Model is trained • Usually 80% of the dataset	• Model is assessed • Usually 20% of the dataset • Also called hold-out or development set	• Model gives predictions • Unseen data

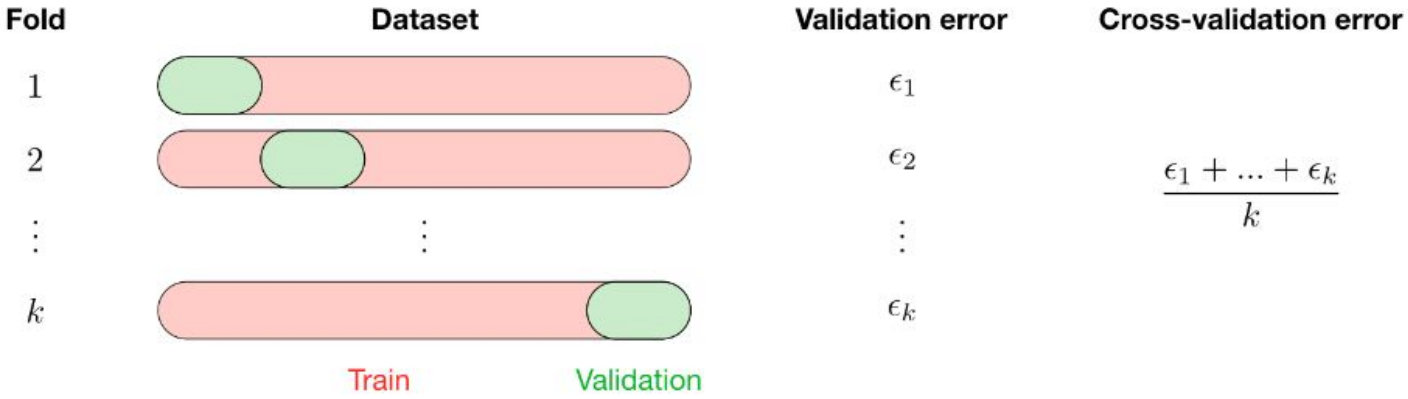
Once the model has been chosen, it is trained on the entire dataset and tested on the unseen test set. These are represented in the figure below:



❒ **Cross-validation** — Cross-validation, also noted CV, is a method that is used to select a model that does not rely too much on the initial training set. The different types are summed up in the table below:

k-fold	Leave-p-out
• Training on $k-1$ folds and assessment on the remaining one • Generally $k=5$ or 10	• Training on $n-p$ observations and assessment on the p remaining ones • Case $p=1$ is called leave-one-out

The most commonly used method is called k-fold cross-validation and splits the training data into k folds to validate the model on one fold while training the model on the $k-1$ other folds, all of this k times. The error is then averaged over the k folds and is named cross-validation error.



Unsupervised Learning

The class of unsupervised learning methods aims at discovering the structure of the data, which may have rich latent structures.

k-means

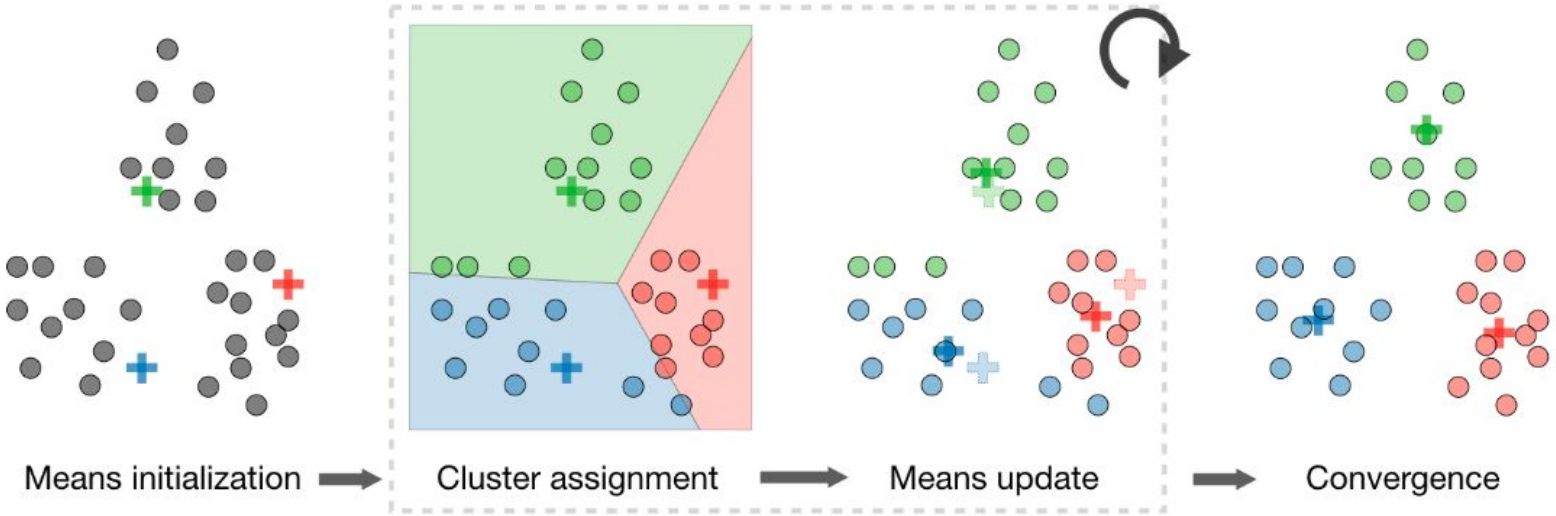
❒ **Clustering** — Given a training set of n input points $\mathcal{D}_{\text{train}}$, the goal of a clustering algorithm is to assign each point $\phi(x_i)$ to a cluster $z_i \in \{1, ..., k\}$.

❒ **Objective function** — The loss function for one of the main clustering algorithms, k-means, is given by:

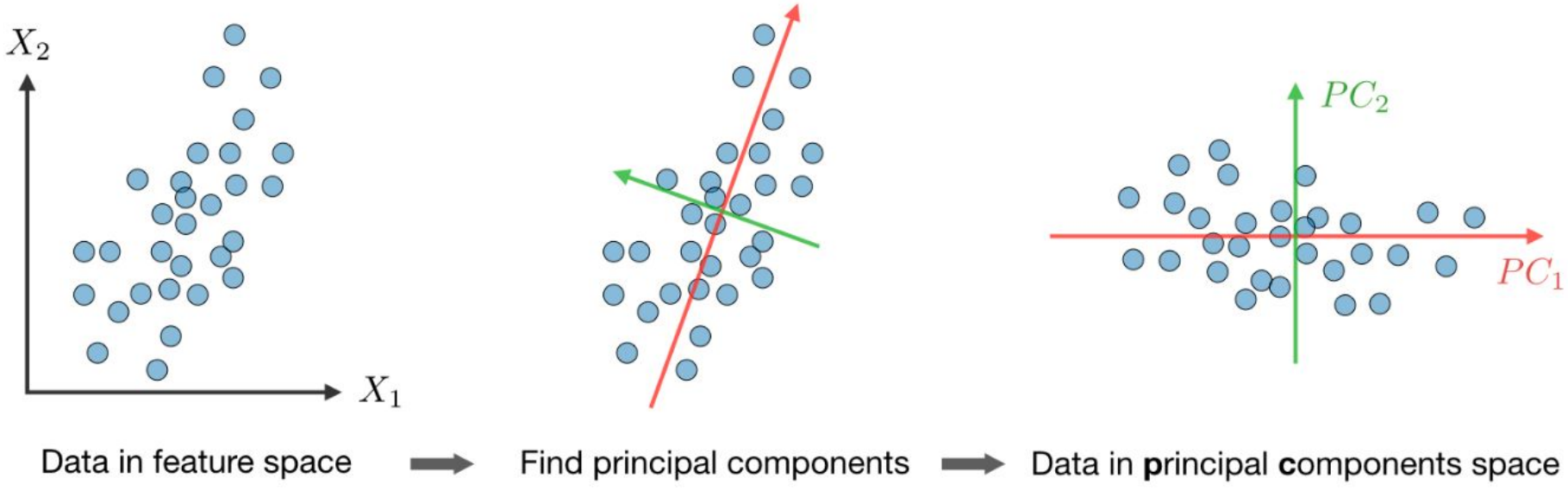
$$\boxed{\text{Loss}_{\text{k-means}}(x, \mu) = \sum_{i=1}^{n} ||\phi(x_i) - \mu_{z_i}||^2}$$

❒ **Algorithm** — After randomly initializing the cluster centroids $\mu_1, \mu_2, ..., \mu_k \in \mathbb{R}^d$, the k-means algorithm repeats the following step until convergence:

$$z_i = \arg\min_j ||\phi(x_i) - \mu_j||^2 \quad \text{and} \quad \mu_j = \frac{\sum_{i=1}^{n} 1_{\{z_i=j\}} \phi(x_i)}{\sum_{i=1}^{n} 1_{\{z_i=j\}}}$$



Principal Component Analysis (PCA)



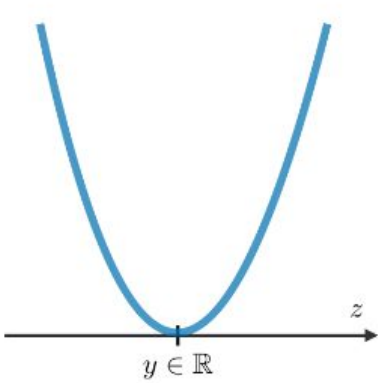
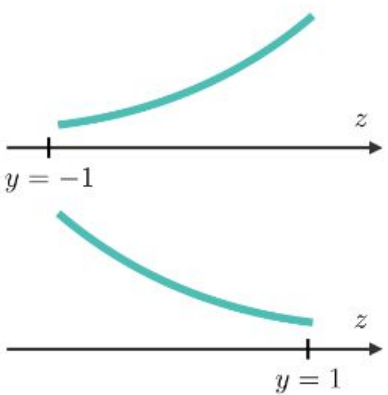
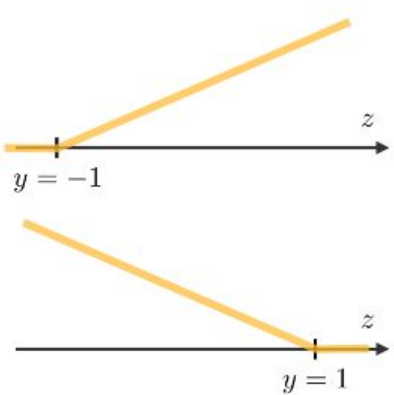
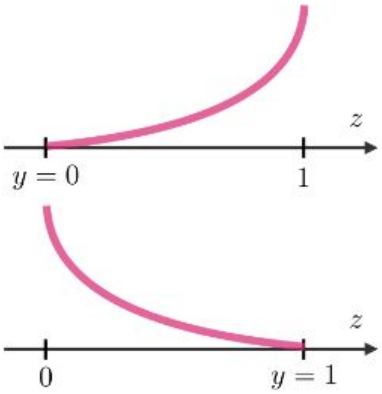
Introduction to Supervised Learning

Given a set of data points $\{x^{(1)}, ..., x^{(m)}\}$ associated to a set of outcomes $\{y^{(1)}, ..., y^{(m)}\}$, we want to build a classifier that learns how to predict y from x.

❒ **Type of prediction** — The different types of predictive models are summed up in the table below:

	Regression	**Classification**
Outcome	Continuous	Class
Examples	Linear regression	Logistic regression, SVM, Naive Bayes

❒ **Loss function** — A loss function is a function $L : (z, y) \in \mathbb{R} \times Y \longmapsto L(z, y) \in \mathbb{R}$ that takes as inputs the predicted value z corresponding to the real data value y and outputs how different they are. The common loss functions are summed up in the table below:

Least squared error	Logistic loss	Hinge loss	Cross-entropy
$\frac{1}{2}(y-z)^2$	$\log(1+\exp(-yz))$	$\max(0, 1-yz)$	$-\left[y\log(z)+(1-y)\log(1-z)\right]$
			
Linear regression	Logistic regression	SVM	Neural Network

❒ **Confusion matrix** — The confusion matrix is used to have a more complete picture when assessing the performance of a model. It is defined as follows:

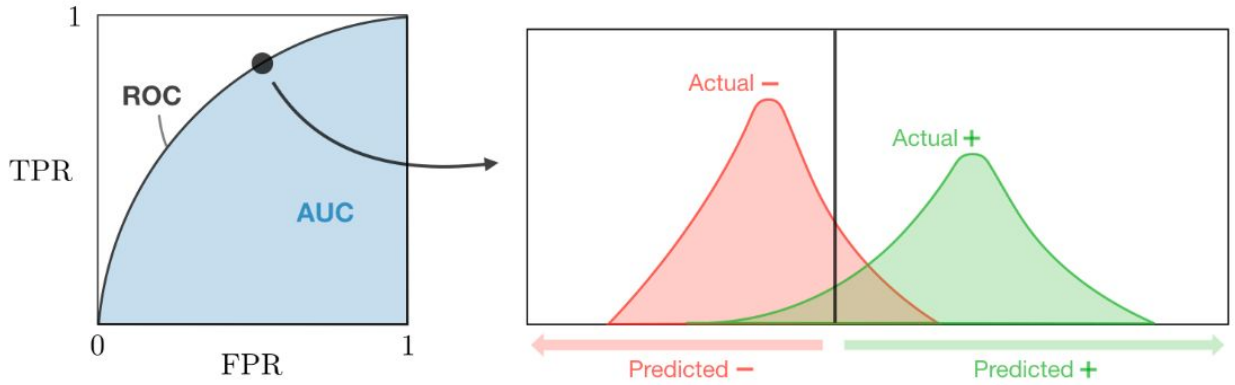
		Predicted class	
		+	-
Actual class	+	**TP** True Positives	**FN** False Negatives Type II error
	-	**FP** False Positives Type I error	**TN** True Negatives

Metric	Formula	Interpretation
Accuracy	$\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$	Overall performance of model
Precision	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$	How accurate the positive predictions are
Recall Sensitivity	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$	Coverage of actual positive sample
Specificity	$\frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}}$	Coverage of actual negative sample
F1 score	$\frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}$	Hybrid metric useful for unbalanced classes

❒ **ROC** — The receiver operating curve, also noted ROC, is the plot of TPR versus FPR by varying the threshold. These metrics are are summed up in the table below:

Metric	Formula	Equivalent
True Positive Rate TPR	$\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$	Recall, sensitivity
False Positive Rate FPR	$\frac{\mathrm{FP}}{\mathrm{TN}+\mathrm{FP}}$	1-specificity

❒ **AUC** — The area under the receiving operating curve, also noted AUC or AUROC, is the area below the ROC as shown in the following figure:



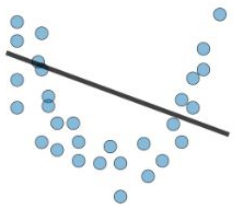
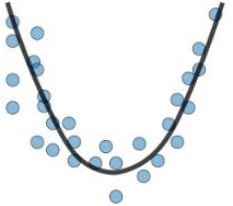
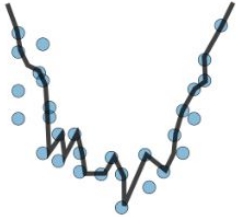
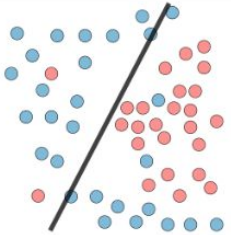
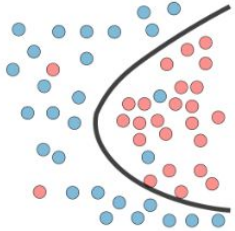
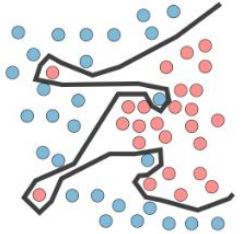
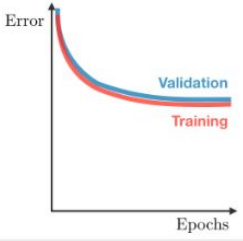
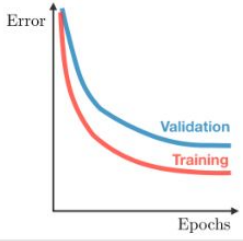
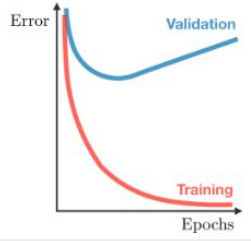
Regression metrics

❒ **Basic metrics** — Given a regression model f, the following metrics are commonly used to assess the performance of the model:

Total sum of squares	Explained sum of squares	Residual sum of squares
$\mathrm{SS}_{\mathrm{tot}} = \sum_{i=1}^{m} (y_i - \overline{y})^2$	$\mathrm{SS}_{\mathrm{reg}} = \sum_{i=1}^{m} (f(x_i) - \overline{y})^2$	$\mathrm{SS}_{\mathrm{res}} = \sum_{i=1}^{m} (y_i - f(x_i))^2$

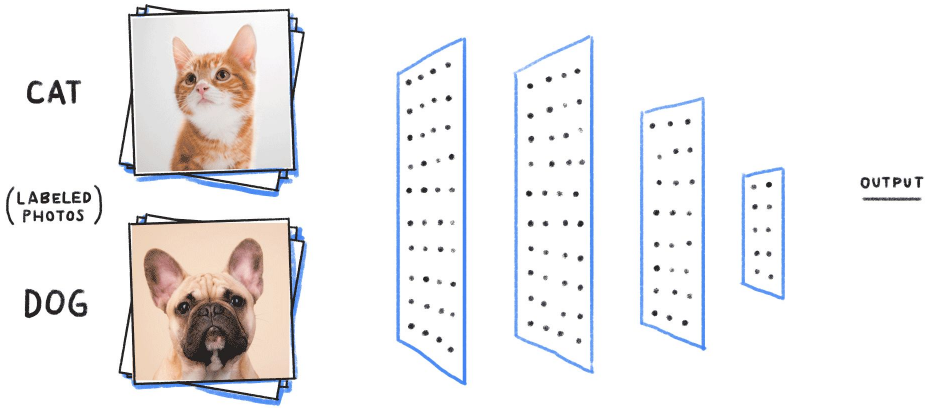
Bias/variance tradeoff - The simpler the model, the higher the bias, and the more complex the model, the higher the variance.

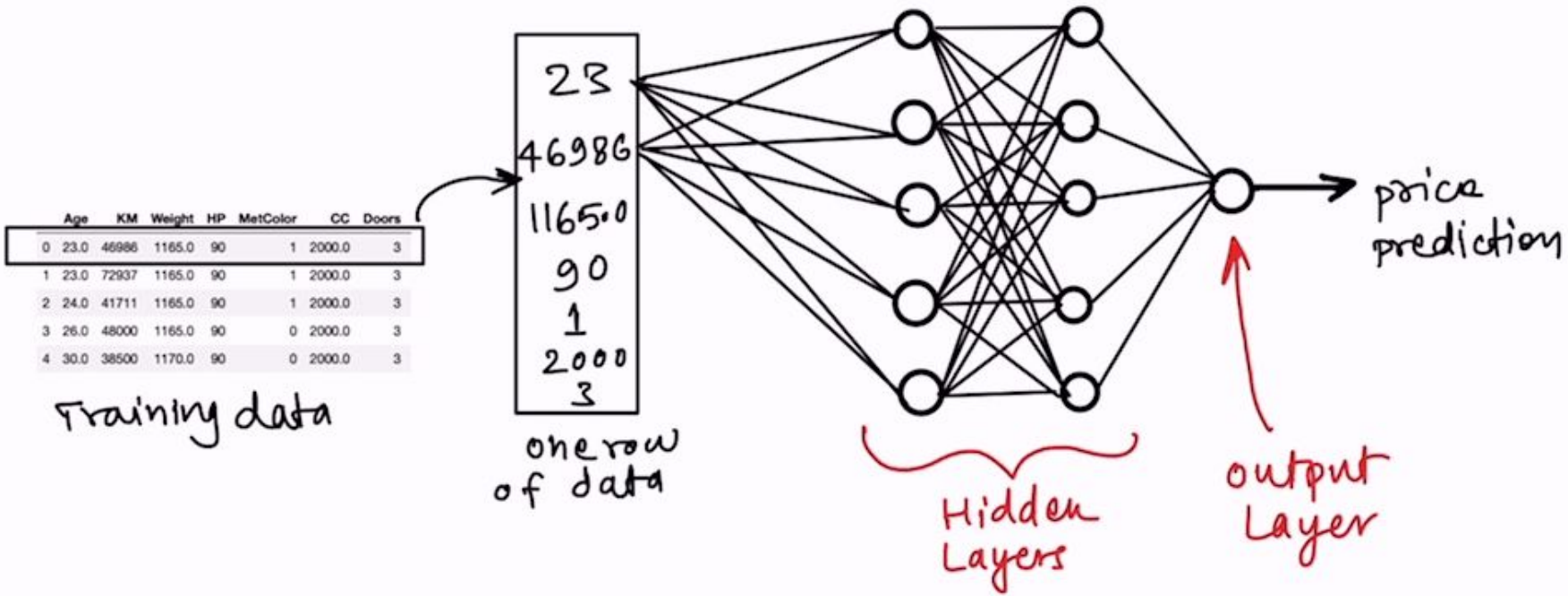
❒ **Bias/variance tradeoff** — The simpler the model, the higher the bias, and the more complex the model, the higher the variance.

	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			

Credit:
https://thinkingneuron.com/using-artificial-neural-networks-for-regression-in-python/

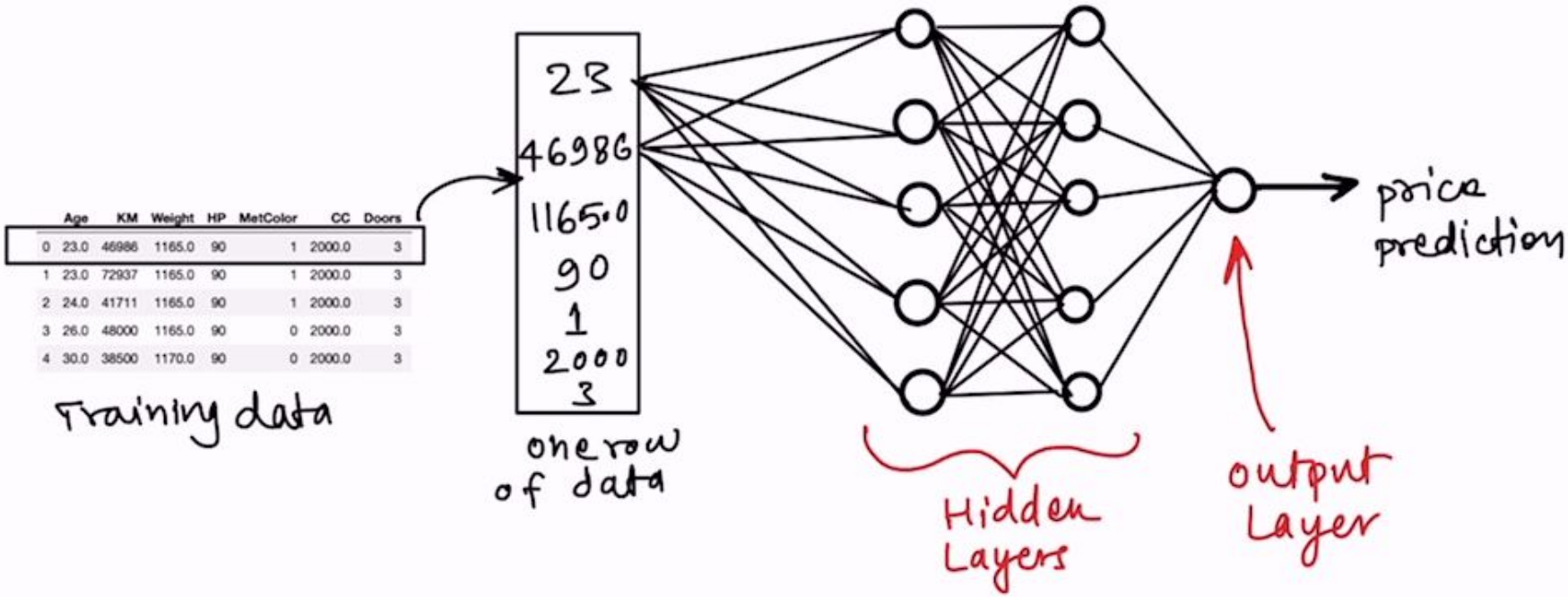
Using Artificial Neural Networks for Regression in Python




Age KM Weight HP MetColor CC Doors
0 23.0 46986 1165.0 90 1 2000.0 3
1 23.0 72937 1165.0 90 1 2000.0 3
2 24.0 41711 1165.0 90 1 2000.0 3
3 26.0 48000 1165.0 90 0 2000.0 3
4 30.0 38500 1170.0 90 0 2000.0 3
Training data
23
46986
1165.0
90
1
2000
3
one row of data
Hidden Layers
output Layer
price prediction

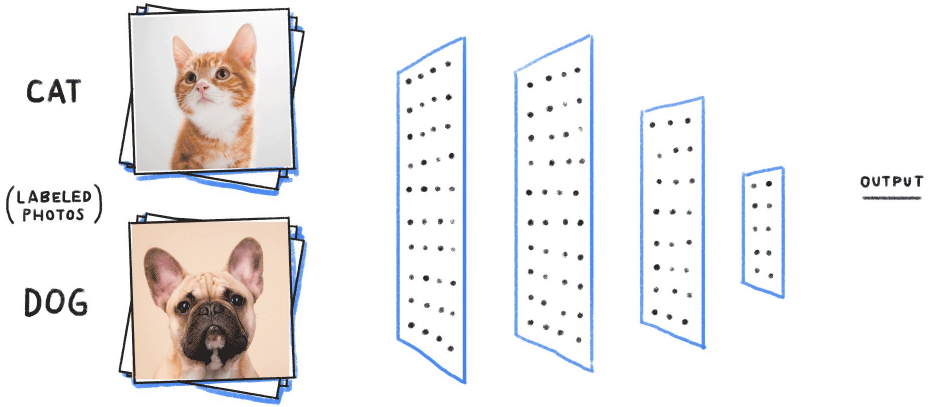
- **Price**: The Price of the car in dollars
- **Age**: The age of the car in months
- **KM**: How many KMS did the car was used
- **HP**: Horsepower of the car
- **MetColor**: Whether the car has a metallic color or not
- **CC**: The engine size of the car
- **Doors**: The number of doors in the car
- **Weight**: The weight of the car

	Age	KM	Weight	HP	MetColor	CC	Doors	Price
0	23.0	46986	1165.0	90	1	2000.0	3	13500
1	23.0	72937	1165.0	90	1	2000.0	3	13750
2	24.0	41711	1165.0	90	1	2000.0	3	13950
3	26.0	48000	1165.0	90	0	2000.0	3	14950
4	30.0	38500	1170.0	90	0	2000.0	3	13750


Age KM Weight HP MetColor CC Doors
0 23.0 46986 1165.0 90 1 2000.0 3
1 23.0 72937 1165.0 90 1 2000.0 3
2 24.0 41711 1165.0 90 1 2000.0 3
3 26.0 48000 1165.0 90 0 2000.0 3
4 30.0 38500 1170.0 90 0 2000.0 3
Training data
23
46986
1165.0
90
1
2000
3
one row of data
Hidden Layers
output Layer
price prediction

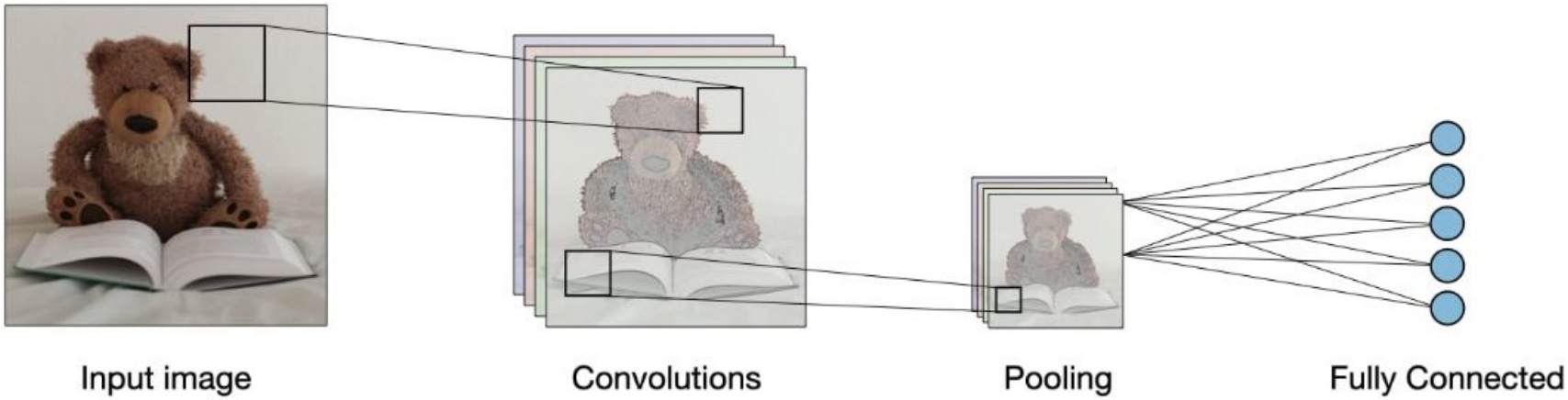
Credit:
https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

Convolutional Neural Networks



Overview

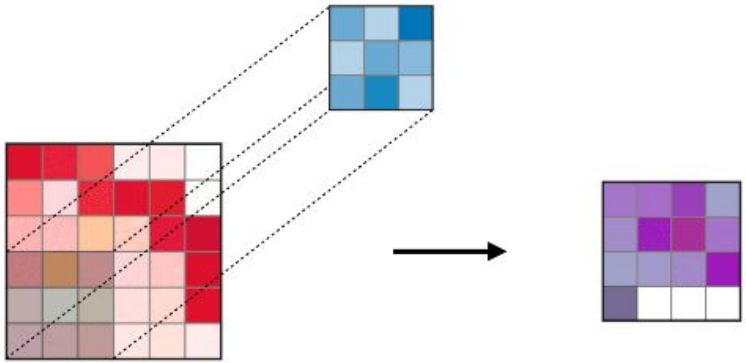
❒ **Architecture of a traditional CNN** — Convolutional neural networks, also known as CNNs, are a specific type of neural networks that are generally composed of the following layers:



The convolution layer and the pooling layer can be fine-tuned with respect to hyperparameters that are described in the next sections.

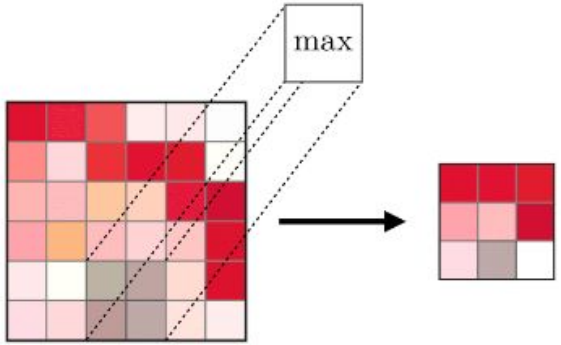
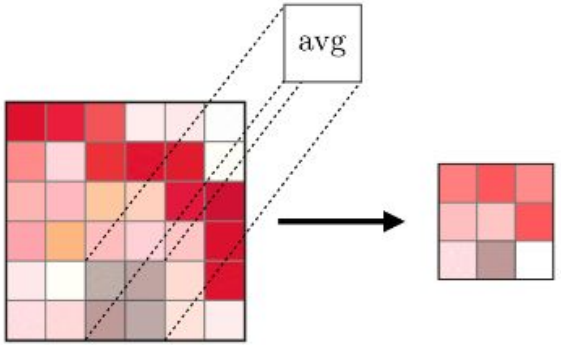
Types of layer

❒ **Convolution layer (CONV)** — The convolution layer (CONV) uses filters that perform convolution operations as it is scanning the input I with respect to its dimensions. Its hyperparameters include the filter size F and stride S. The resulting output O is called *feature map* or *activation map*.

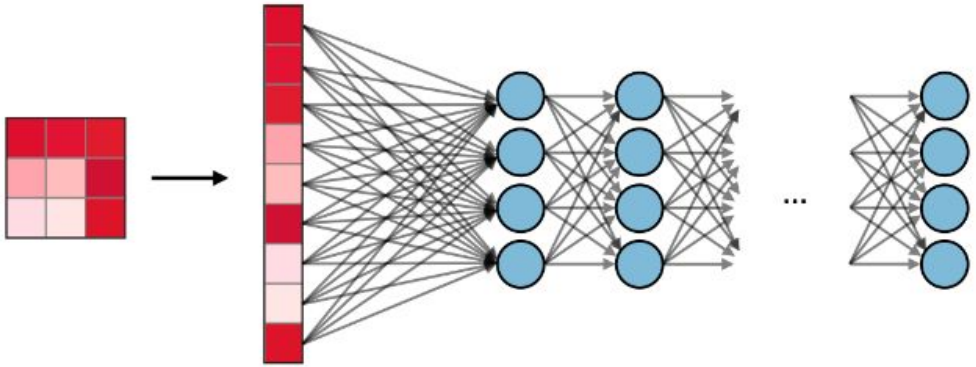


Remark: the convolution step can be generalized to the 1D and 3D cases as well.

❐ **Pooling (POOL)** — The pooling layer (POOL) is a downsampling operation, typically applied after a convolution layer, which does some spatial invariance. In particular, max and average pooling are special kinds of pooling where the maximum and average value is taken, respectively.

Type	Max pooling	Average pooling
Purpose	Each pooling operation selects the maximum value of the current view	Each pooling operation averages the values of the current view
Illustration		
Comments	• Preserves detected features • Most commonly used	• Downsamples feature map • Used in LeNet

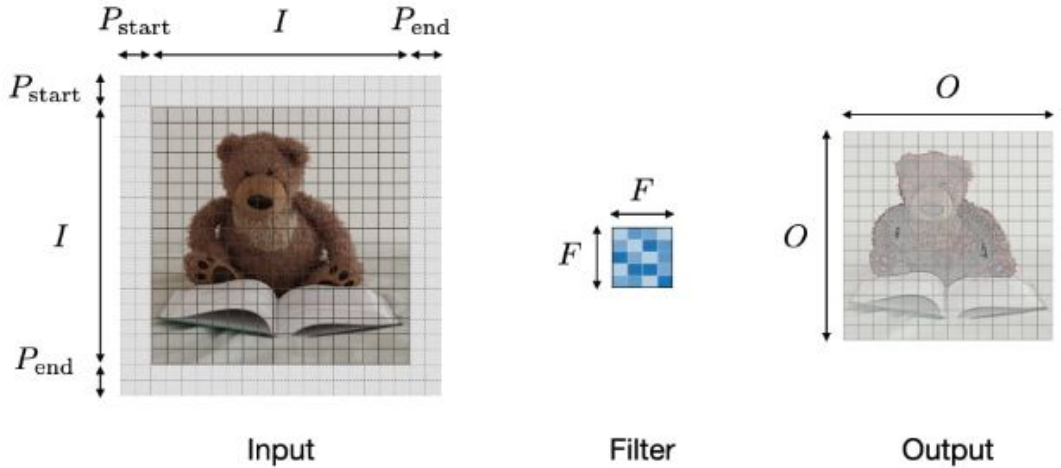
❒ **Fully Connected (FC)** — The fully connected layer (FC) operates on a flattened input where each input is connected to all neurons. If present, FC layers are usually found towards the end of CNN architectures and can be used to optimize objectives such as class scores.



Tuning hyperparameters

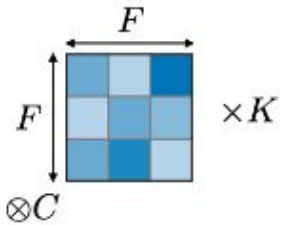
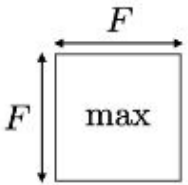
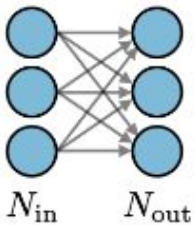
❒ **Parameter compatibility in convolution layer** — By noting I the length of the input volume size, F the length of the filter, P the amount of zero padding, S the stride, then the output size O of the feature map along that dimension is given by:

$$\boxed{O = \frac{I - F + P_{\text{start}} + P_{\text{end}}}{S} + 1}$$



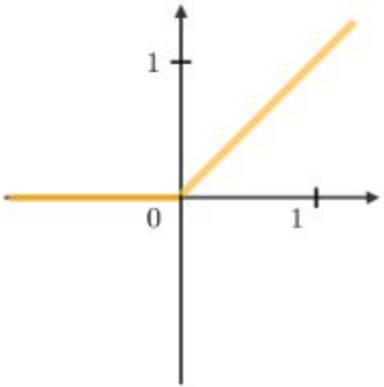
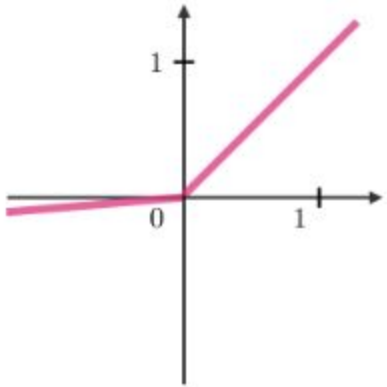
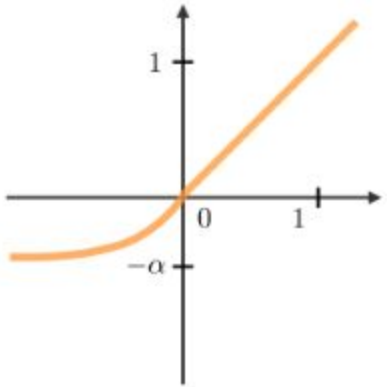
Remark: often times, $P_{\text{start}} = P_{\text{end}} \triangleq P$, in which case we can replace $P_{\text{start}} + P_{\text{end}}$ by $2P$ in the formula above.

❒ **Understanding the complexity of the model** — In order to assess the complexity of a model, it is often useful to determine the number of parameters that its architecture will have. In a given layer of a convolutional neural network, it is done as follows:

	CONV	POOL	FC
Illustration			
Input size	$I \times I \times C$	$I \times I \times C$	N_{in}
Output size	$O \times O \times K$	$O \times O \times C$	N_{out}
Number of parameters	$(F \times F \times C + 1) \cdot K$	0	$(N_{\text{in}} + 1) \times N_{\text{out}}$
Remarks	• One bias parameter per filter • In most cases, $S < F$ • A common choice for K is $2C$	• Pooling operation done channel-wise • In most cases, $S = F$	• Input is flattened • One bias parameter per neuron • The number of FC neurons is free of structural constraints

Commonly used activation functions

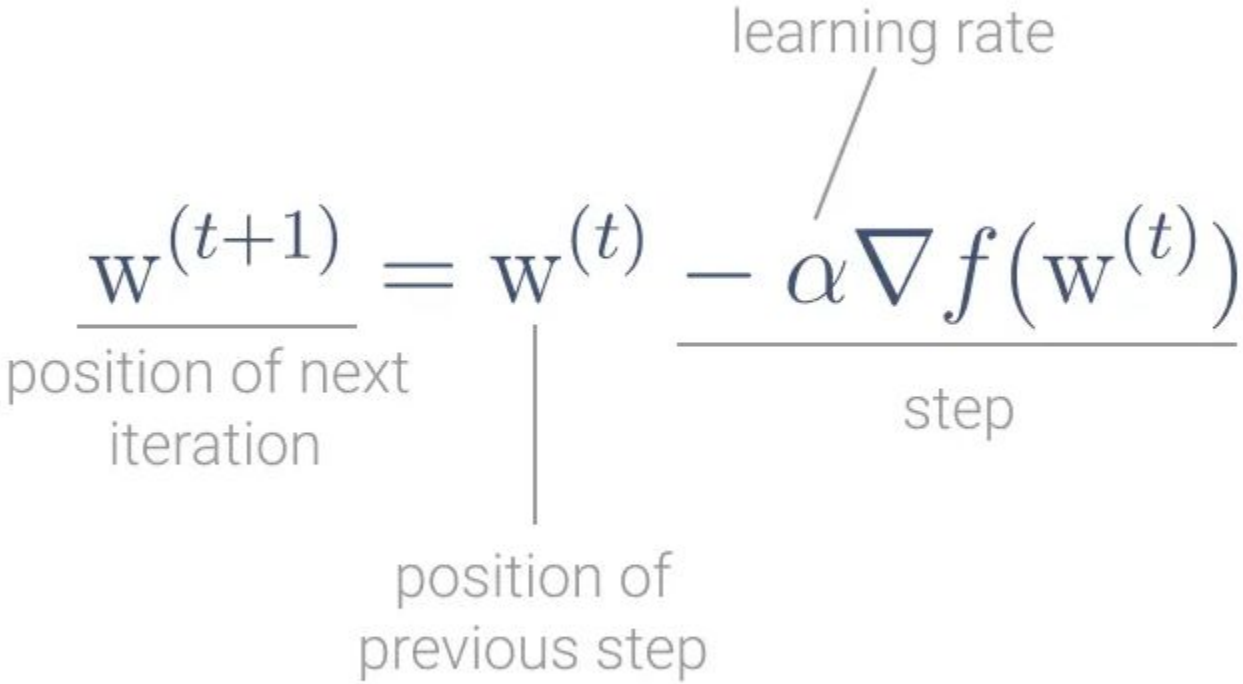
❒ **Rectified Linear Unit** — The rectified linear unit layer (ReLU) is an activation function g that is used on all elements of the volume. It aims at introducing non-linearities to the network. Its variants are summarized in the table below:

ReLU	Leaky ReLU	ELU
$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$	$g(z) = \max(\alpha(e^z - 1), z)$ with $\alpha \ll 1$
		
• Non-linearity complexities biologically interpretable	• Addresses dying ReLU issue for negative values	• Differentiable everywhere

❒ **Softmax** — The softmax step can be seen as a generalized logistic function that takes as input a vector of scores $x \in \mathbb{R}^n$ and outputs a vector of output probability $p \in \mathbb{R}^n$ through a softmax function at the end of the architecture. It is defined as follows:

$$\boxed{p = \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix}} \quad \text{where} \quad \boxed{p_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}}$$

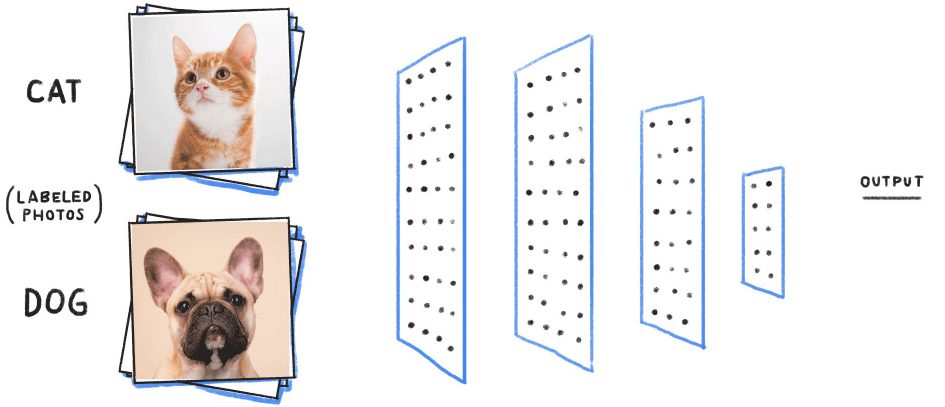
https://towardsdatascience.com/stochastic-gradient-descent-explained-in-real-life-predicting-your-pizzas-cooking-time-b7639d5e6a32



Gradient Descent update rule for step t+1.

```
model_CNNLSTM = tf.keras.Sequential([

    tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(100, return_sequences=True),
                                  input_shape=(win_length, num_features)),

    tf.keras.layers.Dropout(0.5),

    tf.keras.layers.Conv1D(filters=128, kernel_size=2,
                           activation=tf.keras.layers.LeakyReLU(alpha=0.7)),
    tf.keras.layers.MaxPooling1D(pool_size=1),

    tf.keras.layers.Flatten(),

    tf.keras.layers.Dense(targets.shape[1])
])

model_CNNLSTM.compile(optimizer=tf.keras.optimizers.Nadam(learning_rate=0.001),
                      loss=tf.losses.MeanSquaredLogarithmicError())
```

Exercises



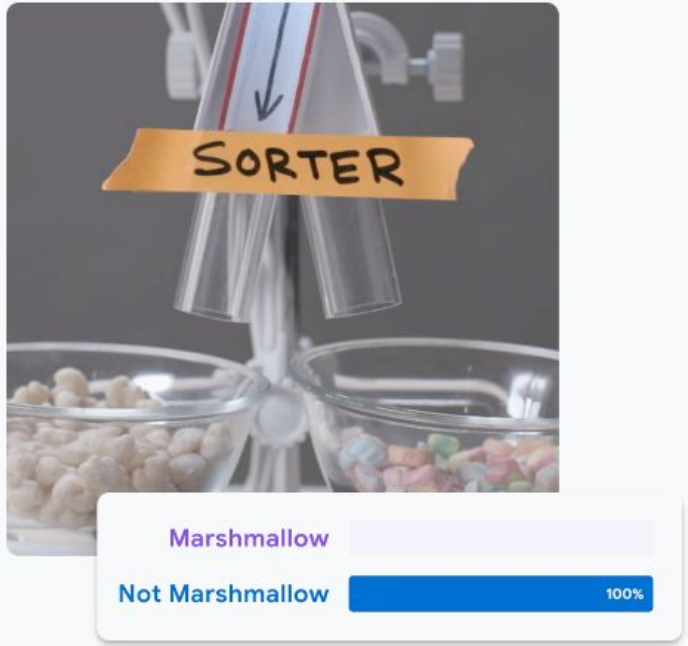
https://teachablemachine.withgoogle.com/

Teachable Machine

Train a computer to recognize your own images, sounds, & poses.

A fast, easy way to create machine learning models for your sites, apps, and more – no expertise or coding required.

Get Started



Teachable Machine

Annual Crop

20 Image Samples

Webcam Upload

Forest

20 Image Samples

Webcam Upload

River

20 Image Samples

Webcam Upload

Add a class

Training

Model Trained

Advanced

Epochs: 50

Batch Size: 16

Learning Rate: 0.001

Reset Defaults

Under the hood

Preview

Export Model

Input ON File

Choose images from your files, or drag & drop here

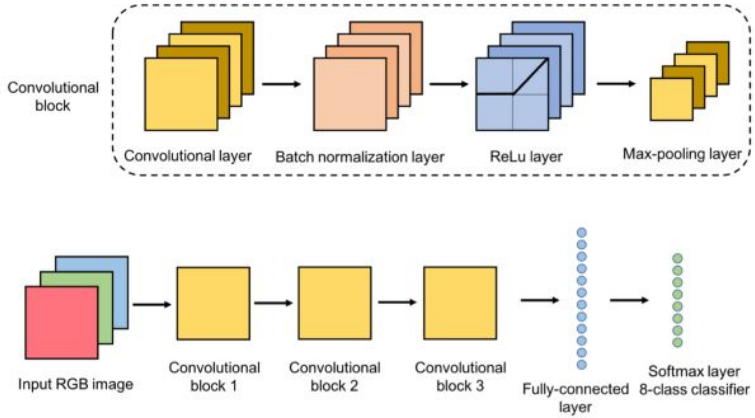
Import images from Google Drive

Output

Annual Crop 100%

Deep Learning (Sample Code):

- Basic Convolution Neural Networks: 



Deep Learning (Sample Code):

- Sample BiLSTM with Conv1D Code (Air Pollution Data Set): Github Notebook