Unit-2

Operational Deep Learning

edunet foundation

Objective

- Gradient Descent
- Cost Function
- Learning rule
- Accuracy Metrics
- Overfitting Vs Underfitting
- Regularization
- TensorFlow
- Keras

Gradient Descent

- Gradient Descent is known as one of the most used optimization algorithms to train machine learning models by means of minimizing errors between actual and expected results.
- Gradient descent was initially discovered by "Augustin-Louis Cauchy" in the mid 18th century.
- Gradient Descent is defined as one of the most used iterative optimization algorithms of machine learning to train the machine learning and deep learning models. It helps in finding the local minimum of a function.

edunet foundation

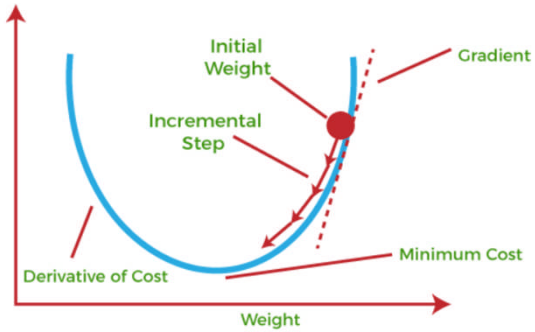
Gradient Descent

- The best way to define the local minimum or local maximum of a function using gradient descent is as follows:
- If we move towards a negative gradient or away from the gradient of the function at the current point, it will give the local minimum of that function.
- Whenever we move towards a positive gradient or towards the gradient of the function at the current point, we will get the local maximum of that function.

Cost Function

- The main objective of using a gradient descent algorithm is to minimize the cost function using iteration.

Gradient Descent



https://www.javatpoint.com/gradient-descent-in-machine-learning

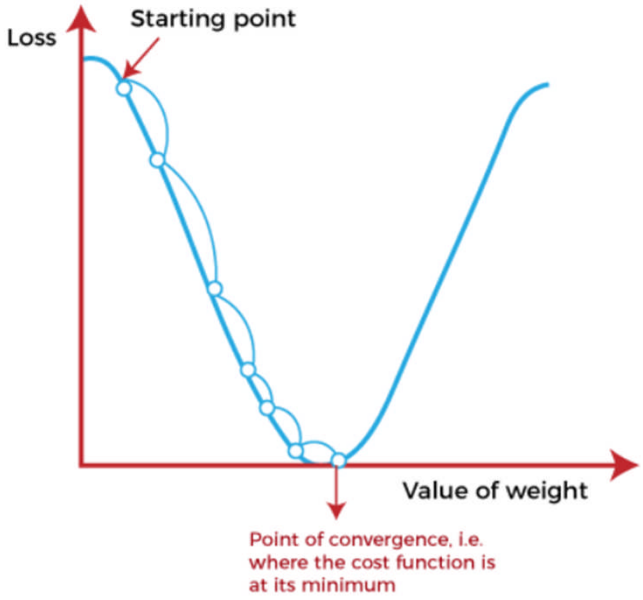
- The cost function is defined as the measurement of difference or error between actual values and expected values at the current position and present in the form of a single real number.

How does Gradient Descent work?

Before starting the working principle of gradient descent, we should know some basic concepts to find out the slope of a line from linear regression. The equation for simple linear regression is given as:

$$y = mx + c$$

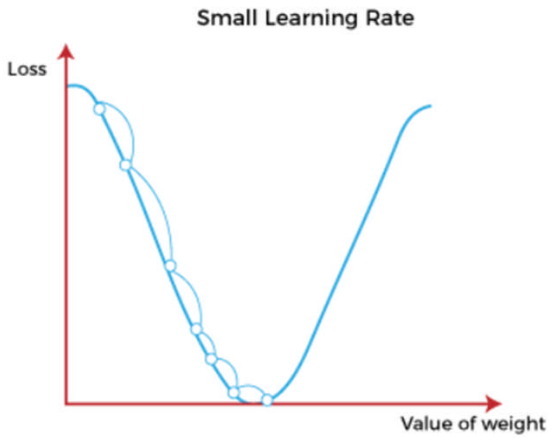
where 'm' represents the slope of the line, and 'c' represents the intercepts on the y-axis.

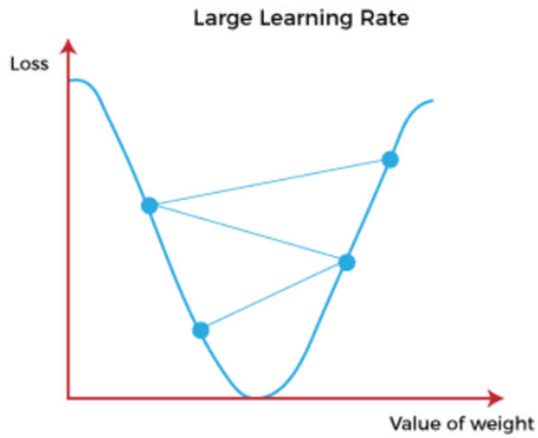


https://www.javatpoint.com/gradient-descent-in-machine-learning

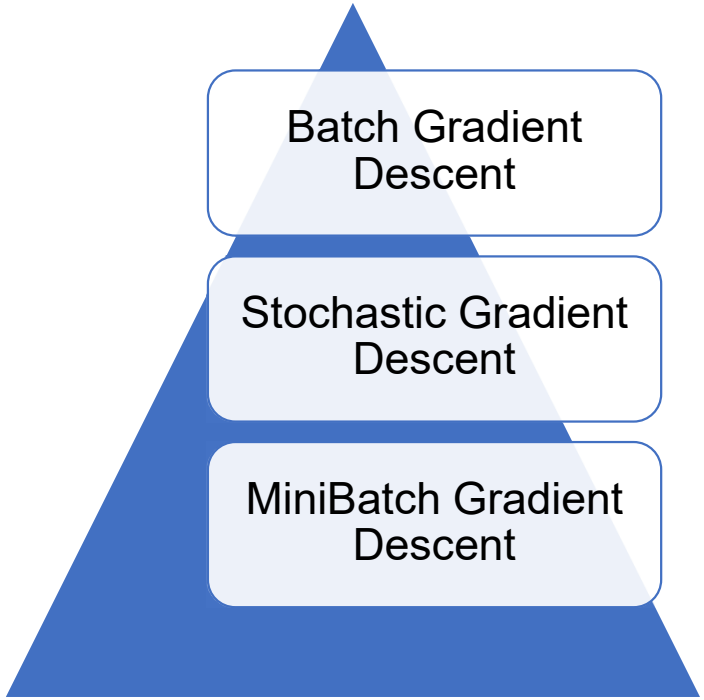
Learning Rate

- It is the step size taken to reach the lowest point.
- Small value evaluated and updated based on the behavior of the cost function.
- High learning rate results in larger steps with risks of overshooting the minimum.
- Low learning rate shows the small step sizes, compromising overall efficiency but gives more precision.



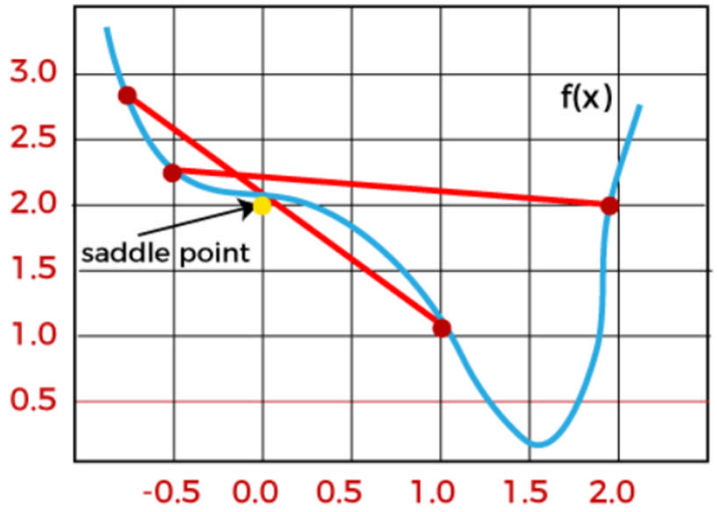


Types of Gradient Descent



Challenges with Gradient Descent

1. Local Minima & Saddle Point
2. Vanishing & Exploding Gradient



edunet foundation

Loss Functions & Accuracy Metrics

Regression Metrics

- Mean Squared Error
- Mean Absolute Error
- 3. Root Mean Squared Error
- R^2 Coefficient of Determination
- Adjusted R^2

Classification Metrics

- Accuracy
- Confusion Matrix
- Precision
- Recall
- F-1 Score

Mean Squared Error

Mean squared error is perhaps the most popular metric used for regression problems. It essentially finds the average of the squared difference between the target value and the value predicted by the regression model.

$$MSE = \frac{1}{N}\sum_{j=1}^{N}(y_j - \check{y}_j)^2$$

where:

- y_j: ground-truth value
- $\hat{y}$: predicted value from the regression model
- N: number of datums

edunet foundation

Mean Absolute Error

Mean Absolute Error is the average of the difference between the ground truth and the predicted values. Mathematically, its represented as

$$MAE = \frac{1}{N}\sum_{j=1}^{N}|y_j - \check{y}_j|$$

where:

y_j: ground-truth value

$\hat{y}$: predicted value from the regression model

N: number of datums

edunet foundation

Root Mean Squared Error

Root Mean Squared Error corresponds to the square root of the average of the squared difference between the target value and the value predicted by the regression model. Basically, sqrt (MSE).

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(y_j - \check{y}_j)^2}$$

Few key points related to RMSE:

- It retains the differentiable property of MSE.
- It handles the penalization of smaller errors done by MSE by square rooting it.

edunet foundation

R^2 Coefficient of Determination

The point of even calculating this coefficient is to answer the question “How much (what %) of the total variation in Y(target) is explained by the variation in X (regression line)”

$$coeff(R^2) = 1 - \frac{SE(line)}{SE(\bar{Y})}$$

Adjusted R^2

The Vanilla R^2 method suffers from some demons, like misleading the researcher into believing that the model is improving when the score is increasing but in reality, the learning is not happening. To rectify this, R^2 is adjusted with the number of independent variables.

$$R_a^2 = 1 - [(\frac{n-1}{n-k-1}).(1-R^2)]$$

where:

- n = number of observations
- k = number of independent variables
- Ra^2 = adjusted R^2

edunet foundation

Accuracy

Classification accuracy is perhaps the simplest metric to use and implement and is defined as the number of correct predictions divided by the total number of predictions, multiplied by 100.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

Confusion Matrix

Confusion Matrix is a tabular visualization of the ground-truth labels versus model predictions.

		Predicted	
		Has Cancer	Doesn't Have Cancer
Ground Truth	Has Cancer	TP	FP
	Doesn't Have Cancer	FN	TN

True Positive (TP) : how many positive class samples your model predicted correctly.
True Negative (TN): how many negative class samples your model predicted correctly.
False Positive (FP) :how many negative class samples your model predicted incorrectly.
False Negative (FN):how many positive class samples your model predicted incorrectly.

edunet foundation

Precision

Precision is the ratio of true positives and total positives predicted.

$$P = \frac{TP}{TP+FP}$$

$0<P<1$

edunet foundation

Recall/Sensitivity/Hit-Rate

A Recall is essentially the ratio of true positives to all the positives in ground truth.

$$R = \frac{TP}{TP+FN}$$

$0<R<1$

F1-score

The F1-score metric uses a combination of precision and recall. In fact, the F1 score is the harmonic mean of the two. The formula of the two essentially is:

$$F_1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}}$$

Now, a high F1 score symbolizes a high precision as well as high recall. It presents a good balance between precision and recall and gives good results on imbalanced classification problems.

Cross Entropy & MSE

Mathematically we can represent cross-entropy as below

Cross Entropy vs. Loss Function

When optimizing classification models, cross-entropy is commonly employed as a loss function. The logistic regression technique and artificial neural network can be utilized for classification problems.

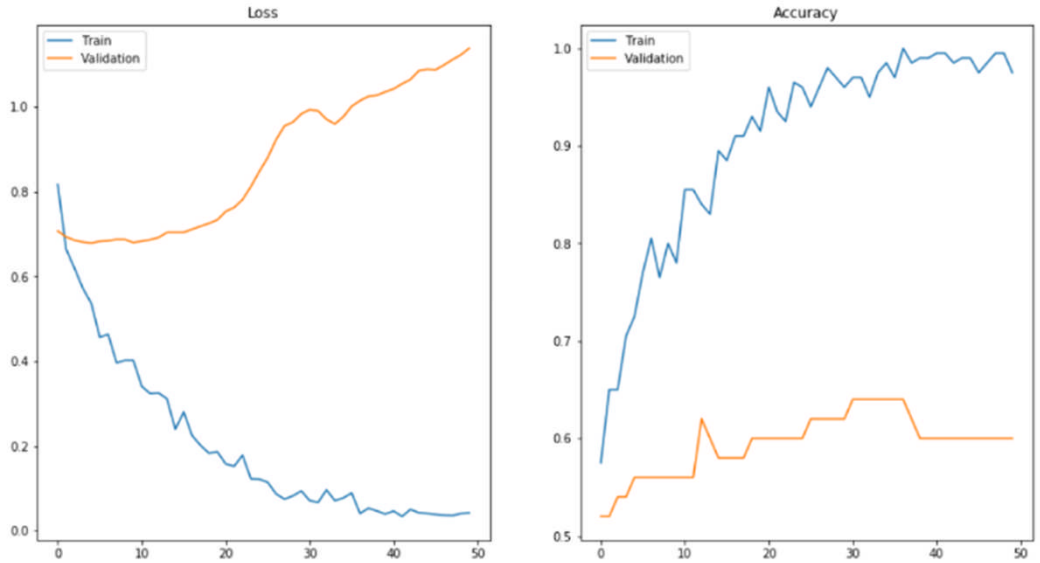
$$H(p, q) = -\sum_{x \in \mathcal{X}} p(x) \log q(x)$$

- Class 1 = 1 (originally predicted)
- Class 0 = 1 – originally predicted

Overfitting and Regularization

When can overfitting occurs ?

The high variance of the model performance is an indicator of an overfitting problem.

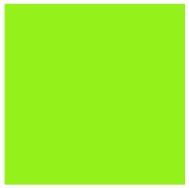


Signs of Overfitting

https://www.v7labs.com/blog/overfitting#:~:text=It%20is%20a%20common%20pitfall,the%20noise%20and%20random%20fluctuations.

Overfitting : Key Concepts

Bias : Bias measures the difference between the model's prediction and the target value.



Variance - Variance is the measure of the inconsistency of different predictions over varied datasets.

Bias-Variance Trade-off - A simple linear model is expected to have a high bias and low variance due to less complexity of the model and fewer trainable parameters.

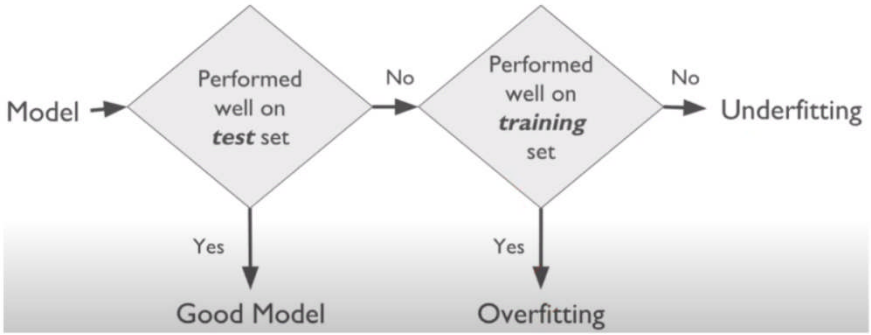
Model generalization - Model generalization means how well the model is trained to extract useful data patterns and classify unseen data samples.

Feature selection - It involves selecting a subset of features from all the extracted features that contribute most towards the model performance.

Overfitting vs. Underfitting

Underfitting occurs when we have a high bias in our data, i.e., we are oversimplifying the problem, and as a result, the model does not work correctly in the training data.

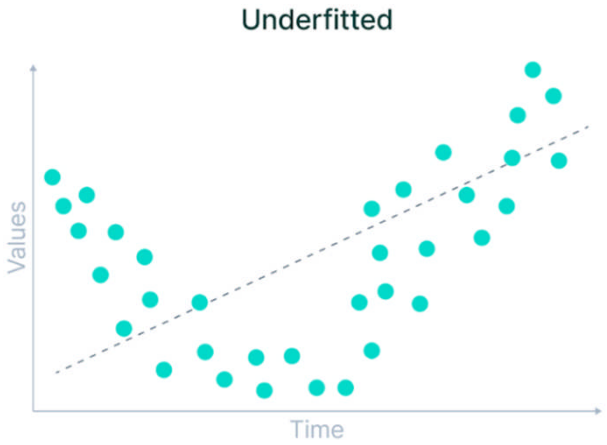
Overfitting occurs when the model has a high variance

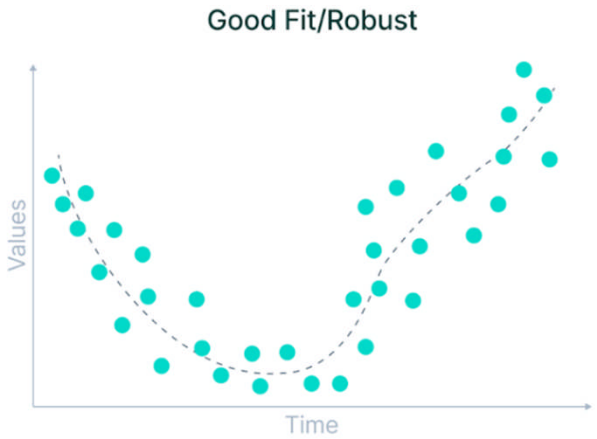


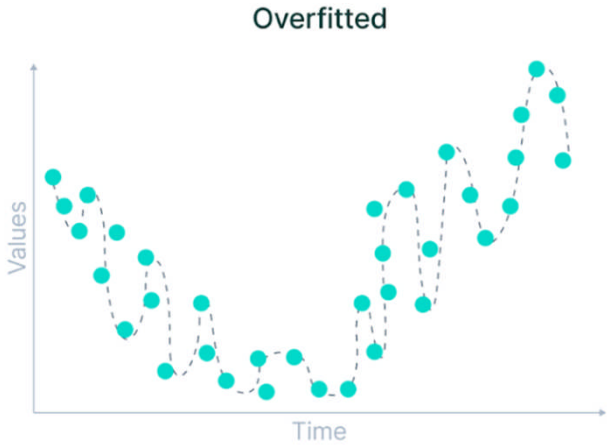
Overfitting vs. Underfitting vs. Good Model

https://www.v7labs.com/blog/overfitting#:~:text=It%20is%20a%20common%20pitfall,the%20noise%20and%20random%20fluctuations.

Visual comparison

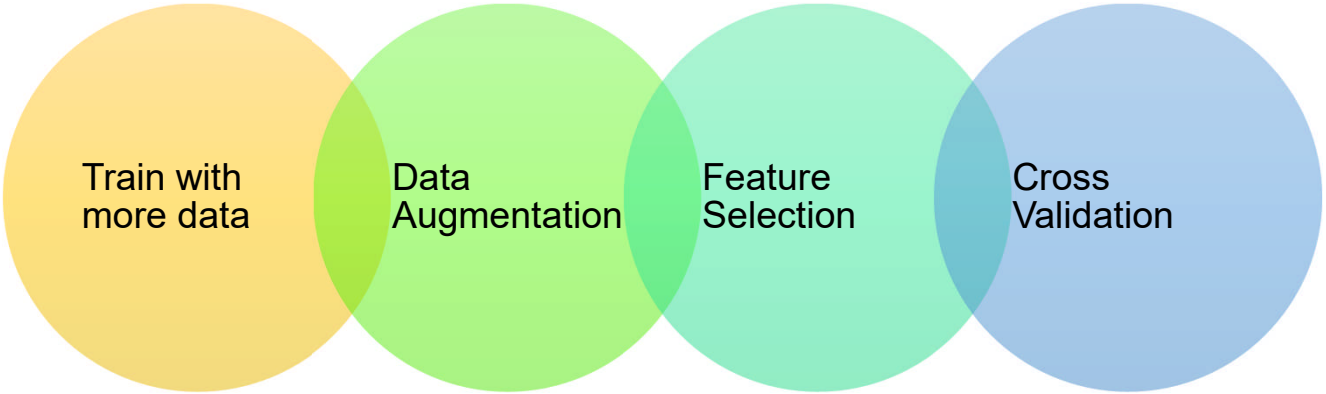






https://www.v7labs.com/blog/overfitting#:~:text=It%20is%20a%20common%20pitfall,the%20noise%20and%20random%20fluctuations.

Techniques to avoid Overfitting



Regularization

Regularization is one of the most important concepts of machine learning. It is a technique to prevent the model from overfitting by adding extra information to it.

How does regularization work ?

$$y = \beta 0 + \beta 1x1 + \beta 2x2 + \beta 3x3 + \cdots + \beta nxn + b$$

In the above equation, Y represents the value to be predicted.

The equation for the cost function for the linear model is given below:

$$\sum_{i=1}^{M} (y_i - y'_i)^2 = \sum_{i=1}^{M} (y_i - \sum_{j=0}^{n} \beta_j * Xij)^2$$

Techniques for Regularization

1. Ridge Regression : The equation for the cost function in ridge regression will be

$$\sum_{i=1}^{M}(y_i - y'_i)^2 = \sum_{i=1}^{M}\left(y_i - \sum_{j=0}^{n}\beta_j * x_{ij}\right)^2 + \lambda\sum_{j=0}^{n}\beta_j{}^2$$

2. Lasso Regression : It is also called as L1 regularization. The equation for the cost function of Lasso regression will be

$$\sum_{i=1}^{M}(y_i - y'_i)^2 = \sum_{i=1}^{M}\left(y_i - \sum_{j=0}^{n}\beta_j * x_{ij}\right)^2 + \lambda\sum_{j=0}^{n}|\beta_j|^{\square}$$

Key Difference between Ridge Regression and Lasso Regression

- Ridge regression is mostly used to reduce the overfitting in the model, and it includes all the features present in the model. It reduces the complexity of the model by shrinking the coefficients.

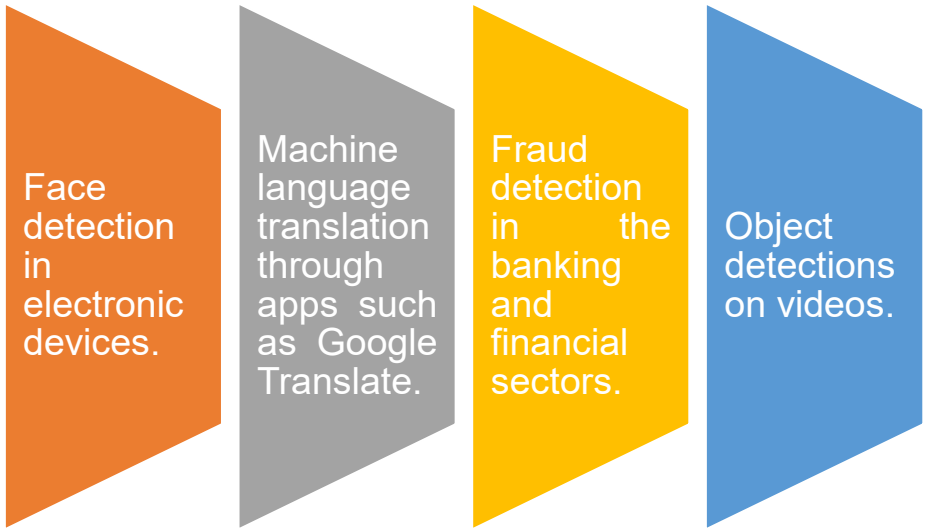
- Lasso regression helps to reduce the overfitting in the model as well as feature selection.

TensorFlow 2.0 & Keras API

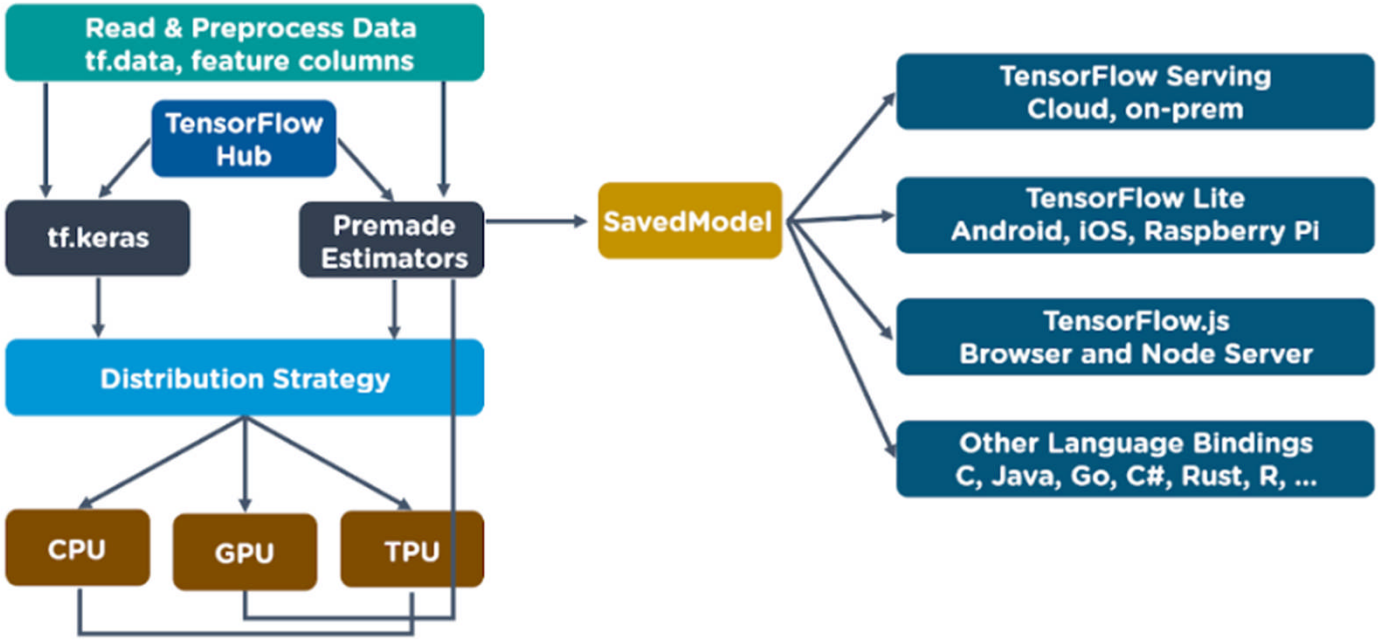
Tensor

TensorFlow 2.0 is a library that provides a comprehensive ecosystem of tools for developers, researchers, and organizations who want to build scalable Machine Learning and Deep Learning applications.

TensorFlow Applications



TensorFlow Architecture



Keras API

Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow.

https://www.simplilearn.com/tutorials/deep-learning-tutorial/tensorflow-2

Building a model in Keras

Dense Layers and Flatten Layers

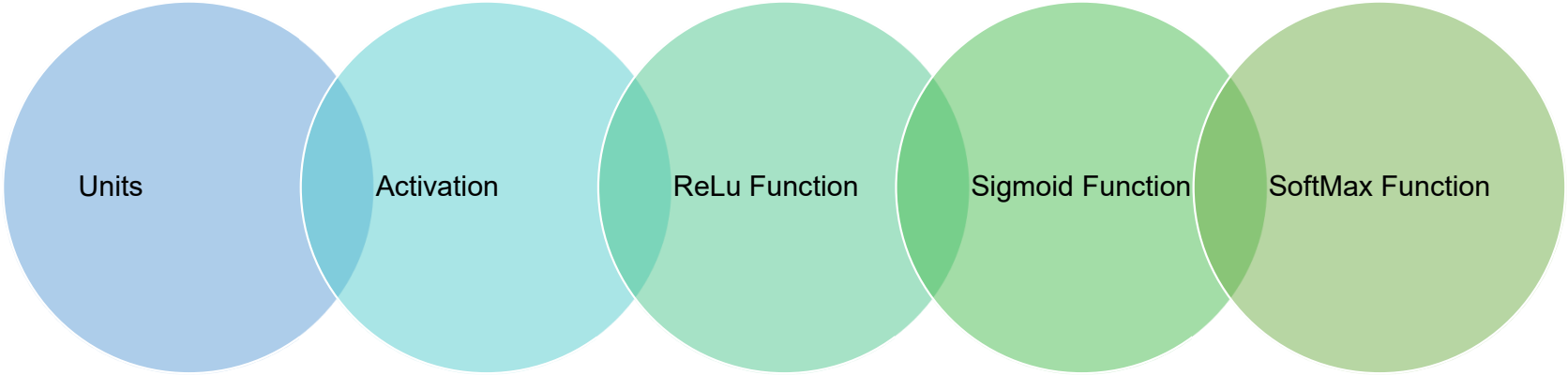
In any neural network, a dense layer is a layer that is deeply connected with its preceding layer which means the neurons of the layer are connected to every neuron of its preceding layer.

The general formula for a matrix-vector product is:

$$\mathbf{Ax} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{bmatrix}.$$

https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/

Dense Layer Hyperparameters



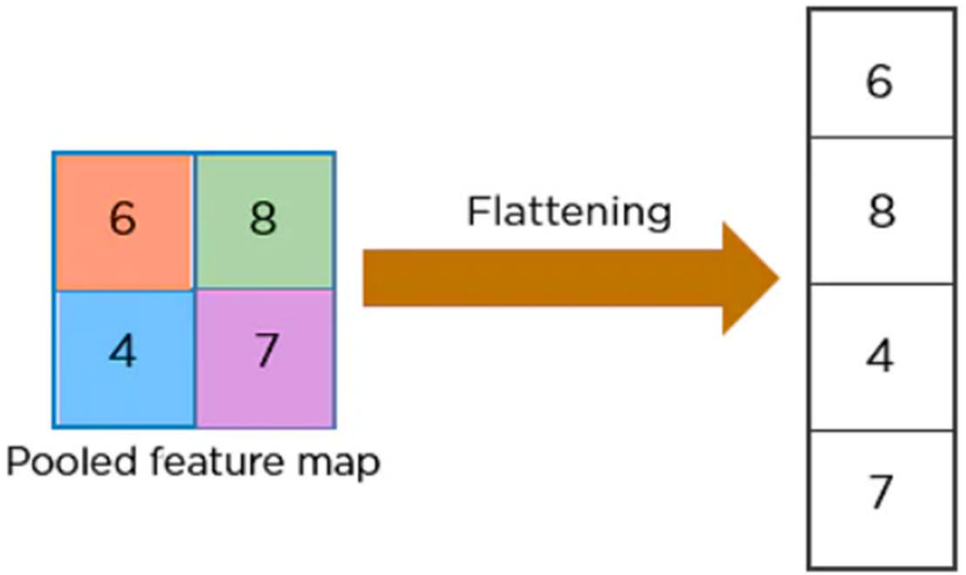
Basic Operations with Dense Layer

As we have seen in the parameters, we have three main attributes: activation function, weight matrix, and bias vector. Using these attributes, a dense layer operation can be represented as:

Output = activation (dot (input, kernel) + bias)

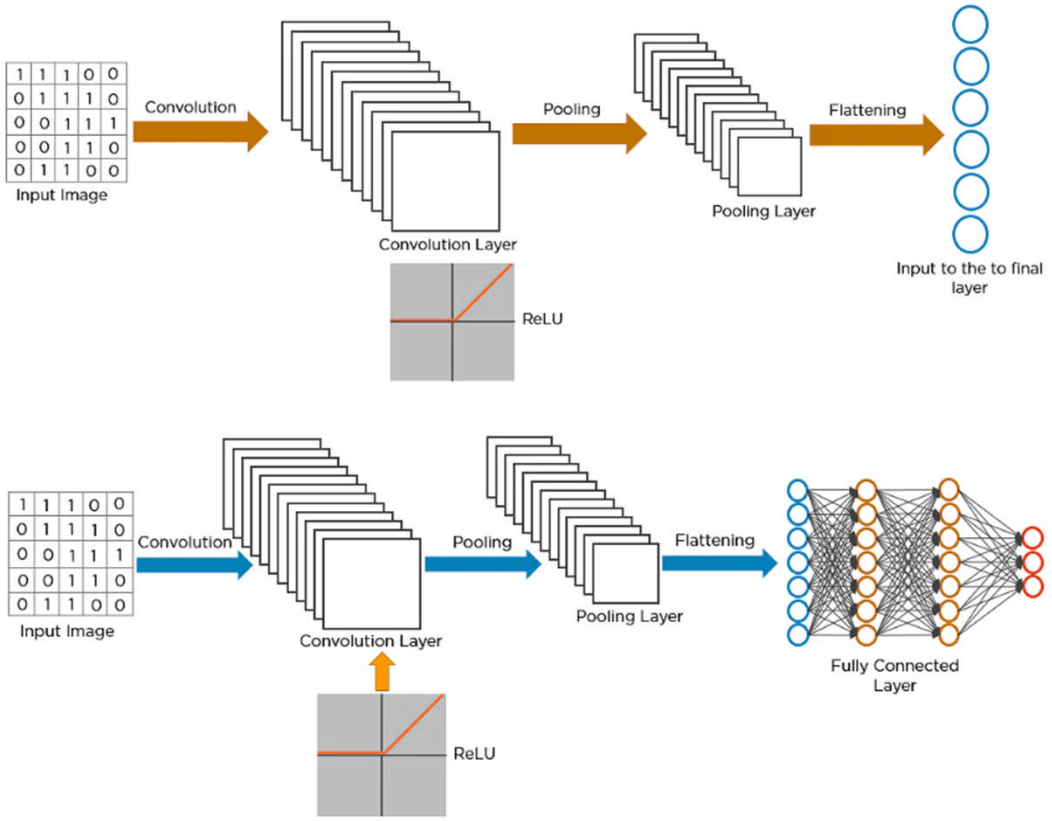
Flatten Layers

Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector.



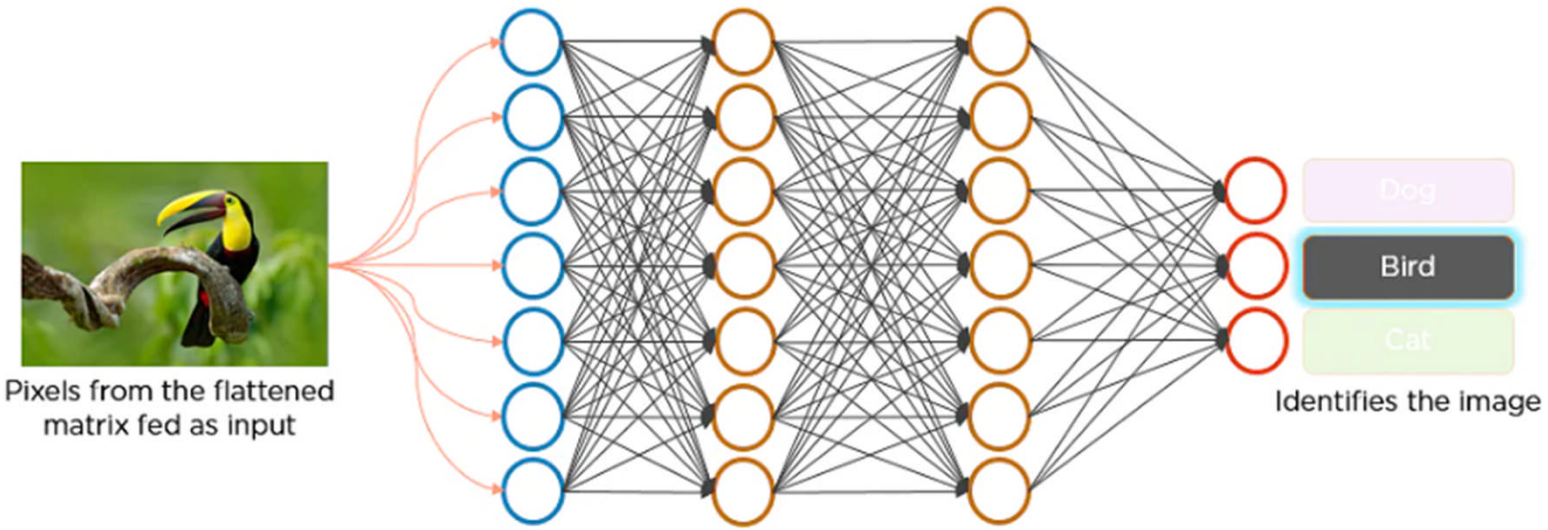
https://www.simplilearn.com/tutorials/deep-learning-tutorial/convolutional-neural-network#:~:text=Flattening%20is%20used%20to%20convert,layer%20to%20classify%20the%20image.

Example of using Flattening layer



https://www.simplilearn.com/tutorials/deep-learning-tutorial/convolutional-neural-network#:~:text=Flattening%20is%20used%20to%20convert,layer%20to%20classify%20the%20image.

How Image classification done using Neural network



https://www.simplilearn.com/tutorials/deep-learning-tutorial/convolutional-neural-network#:~:text=Flattening%20is%20used%20to%20convert,layer%20to%20classify%20the%20image.

edunet foundation

Tensor

Tensors are simply mathematical objects that can be used to describe physical properties, just like scalars and vectors. In fact, tensors are merely a generalisation of scalars and vectors; a scalar is a zero-rank tensor, and a vector is a first rank tensor.

Tensor Operations

Tensor operations are simple. Consider the following tensors:

```
a = np.array([[[4,1,2], [3,5,2], [1,6,7]]
 [[2,1,0], [5,4,3], [6,8,9]]
 [[1,2,3], [2,3,4], [5,5,5]]])
b = np.array([[[1,1,1], [2,2,2], [3,3,3]]
 [[4,4,4], [5,5,5], [6,6,6]]
[[7,7,7], [8,8,8], [9,9,9]]])
```

edunet foundation

Tensor Operations

You can check a tensor's dimension by the following code in Python

```
a.ndim
>>> 3
```

Addition of the discussed two tensors result in a tensor in which each scalar value is the element-wise addition of scalars in the parent tensors.

```
print(a+b)
>>> [[[5,2,3], [5,7,4], [4,9,10]]
 [[6,5,4], [10,9,8], [12,14,15]]
 [[8,10,11], [10,11,12], [14,14,14]]]
```

edunet foundation

References

1. https://www.javatpoint.com/gradient-descent-in-machine-learning
2. https://neptune.ai/blog/performance-metrics-in-machine-learning-complete-guide
3. https://www.section.io/engineering-education/understanding-loss-functions-in-machine-learning/
4. https://analyticsindiamag.com/a-beginners-guide-to-cross-entropy-in-machine-learning/#:~:text=The%20average%20number%20of%20bits,of%20actual%20and%20expected%20results.
5. https://www.v7labs.com/blog/overfitting#:~:text=It%20is%20a%20common%20pitfall,the%20noise%20and%20random%20fluctuations.

edunet foundation

References

6. https://www.javatpoint.com/regularization-in-machine-learning
7. https://www.simplilearn.com/tutorials/deep-learning-tutorial/tensorflow-2
8. https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-keras
9. https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks/
10. https://www.simplilearn.com/tutorials/deep-learning-tutorial/convolutional-neural-network#:~:text=Flattening%20is%20used%20to%20convert,layer%20to%20classify%20the%20image
11. https://towardsdatascience.com/quick-ml-concepts-tensors-eb1330d7760f