Handwriting Comparison using Machine Learning Algorithms

Yashankit Shikhar
UBID: yshikhar

November 2, 2018

Abstract

The purpose of this paper is to propose Machine Learning models such as Linear Regression, Logistic Regression, and Neural Network to solve the handwriting comparison task in forensics. The feature vectors of the handwriting images are obtained from two different sources, Human Observed feature dataset and Gradient Structural Concavity feature dataset. Several hyper-parameters, such as the learning rate, number of basis function, regularization term, etc. are also analyzed. Based on the analysis, the program is executed with obtained values of hyper-parameters and feature vectors.

1 Introduction

In this project, several Machine Learning methods to solve the handwriting comparison task is discussed. For this project, the benchmark CEDAR Letter data set is used from which "AND" images samples have been extracted. The feature vectors are obtained for each image and are compared for same and different writers by two methods, concatenation of feature vectors and subtraction of feature vectors. The target vector takes two values, 1 for same writer and 0 for different writers. First 80% of the data matrix created by feature concatenation or subtraction is used as training data set and last 10% of the data matrix is used as testing data set for both feature concatenation and subtraction. The data sets contain approximately equal numbers of each class. The program is tested on the testing data set.

2 Experiment

The aim of this project is to train a Machine Learning model to predict whether two handwriting images are from the same writers or different writers. The following Machine learning Models have been implemented:

1. **Linear Regression**
   Two approach to train a Linear Regression model is discussed:
   (a) **Closed Form Solution**
   (b) **Stochastic Gradient Descent**

   The following hyper-parameters are analyzed and their effect on the accuracy of the Linear Regression model:
   (a) **Regularization Term** λ
   (b) **Number of Basis Functions M**
   (c) **Learning Rate** η^{τ}
2. **Logistic Regression**
   Learning Rate η^{τ} is analyzed for its effect on the accuracy of the Logistic Regression model
3. **Neural Network**
   The following hyper-parameters are analyzed and their effect on the accuracy of Neural Network:

(a) **Dropout rate of the Neural Network**

(b) **Number of nodes in the hidden layer**

(c) **Input data batch size**

(d) **Optimizer function**

(e) **Activation function**

3 Linear Regression Model

A linear regression function $\mathbf{y}(\mathbf{x}, \mathbf{w})$ is defined as,

$$y(x, w) = w^\top \phi(x) \tag{1}$$

where $\phi_j(x)$ is Gaussian radial basis function given by

$$\phi_j(x) = exp(-1/2(x - \mu_j)^\top (\sum_j)^{-1}(x - \mu_j)) \tag{2}$$

where $\mathbf{w} = (\mathbf{w}_0, \mathbf{w}_1, .., \mathbf{w}_{m-1})$ is a weight vector to be learned from training samples and $\phi = (\phi_0, \phi_1, .., \phi_{m-1})^\top$ is a vector of M basis functions. μ_j is the center of the basis function and $\sum_j$ decides how broadly the basis function spreads. The objective is to minimize the sum-of-squares error,

$$E_D(w) = 1/2 \sum_{n=1}^{N} (t_n - w^\top \phi(x_n))^2 \tag{3}$$

where $\mathbf{t} = (\mathbf{t}_0, \mathbf{t}_1, .., \mathbf{t}_n)$ is the vector of outputs in the training data and ϕ is the design matrix.

3.1 Closed Form Solution

In closed form solution, derivative of objective function is set to zero and required parameters are obtained without resorting to iterative algorithm [1]. Closed form solution is calculated by,

$$w_{ML} = \lambda \mathbf{I} + (\phi^\top \phi)^{-1} \phi^\top t \tag{4}$$

The RMS Error is calculated for training, validation, and test data set using the following formula:

$$E_{RMS} = \sqrt{E(w*)/N_V} \tag{5}$$

3.2 Stochastic Gradient Descent

Gradient Descent is the process of minimizing a function by following the gradients of the cost function [2]. Stochastic Gradient Descent works on a single data point at a time. We start with a single random value of solution. We modify the solution for each data sample using:

$$w^{\tau+1} = w^\tau + \Delta w^\tau \tag{6}$$

where,

$$\Delta w^\tau = -\eta^\tau \nabla E \tag{7}$$

is called weight updates. η^τ is the Learning Rate. Due to linearity of differentiation, we have

$$\nabla E = \nabla E_D + \lambda \nabla E_W \tag{8}$$

in which,

$$\nabla E_D = -(t_n - w^{(\tau)\top} \phi(x_n))\phi(x_n) \tag{9}$$

$$\nabla E_W = w^{(\tau)} \tag{10}$$

4 Logistic Regression Model

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. Like all regression analyses, the logistic regression is a predictive analysis [3]. A logistic regression function $\mathbf{y}(\mathbf{x}, \mathbf{w})$ is defined as,

$$y(x, w) = \sigma(w^\top X) \tag{11}$$

where $\mathbf{w} = (\mathbf{w}_0, \mathbf{w}_1, .., \mathbf{w}_{m-1})$ is a weight vector to be learned from training samples and X is the data matrix.The Loss Function for Logistic Regression is given as,

$$E_D(w) = \frac{1}{m}(-t^\top log(y(x, w)) - (1-t)^\top log(1 - y(x, w))) \tag{12}$$

where $\mathbf{t} = (\mathbf{t}_0, \mathbf{t}_1, .., \mathbf{t}_n)$ is the vector of outputs. The derivative of the loss function with respect to weights is defined as,

$$\nabla E_D = \frac{1}{m} X^\top (y(x, w) - t) \tag{13}$$

We modify the solution for each data sample using:

$$w^{\tau+1} = w^\tau + \Delta w^\tau \tag{14}$$

where,

$$\Delta w^\tau = -\eta^\tau \nabla E_D(w) \tag{15}$$

is called weight updates. η^τ is the Learning Rate.

5 Hyper-parameters

In machine learning, a hyper-parameter is a parameter whose value is set before the learning process begins. Given these hyper-parameters, the training algorithm learns the other parameters from the data [4].

5.1 Regression Model

1. **Regularization Term λ:** Regularization refers to the act of modifying a learning algorithm to favor "simpler" prediction rules to avoid over-fitting. Most commonly, regularization refers to modifying the loss function to penalize certain values of the weights being learned. λ is a hyper-parameter that adjusts the trade-off between having low training loss and having low weights [5].

2. **Number of Basis Functions M:** The number of basis function defines the number of clusters in k-means clustering. The centroid of these clusters form the μ_j matrix.

3. **Learning Rate η^τ:** Learning rate is defined in the context of optimization, and minimizing the loss function of a neural network. For this optimization problem, stochastic gradient descent is used where the model parameters are updated in a way to decrease the cost function [6].

5.2 Neural Network

1. **Dropout rate:** Dropout refers to ignoring neurons during the training phase of a certain set of neurons which is chosen at random. At each training stage, individual nodes are either dropped out of the net with probability **1-P** or kept with probability **P** [7].

2. **Hidden Layer:** In traditional neural networks, a hidden layer neuron is a neuron whose output is connected to the inputs of other neurons and is therefore not visible as a network output. Hidden nodes only receive input from the other nodes and only output to other nodes [8].

3. **Input data batch size:** Input data batch size is defined as the number of samples per gradient update. If unspecified, batch size will be set to the default value of 32 [9].

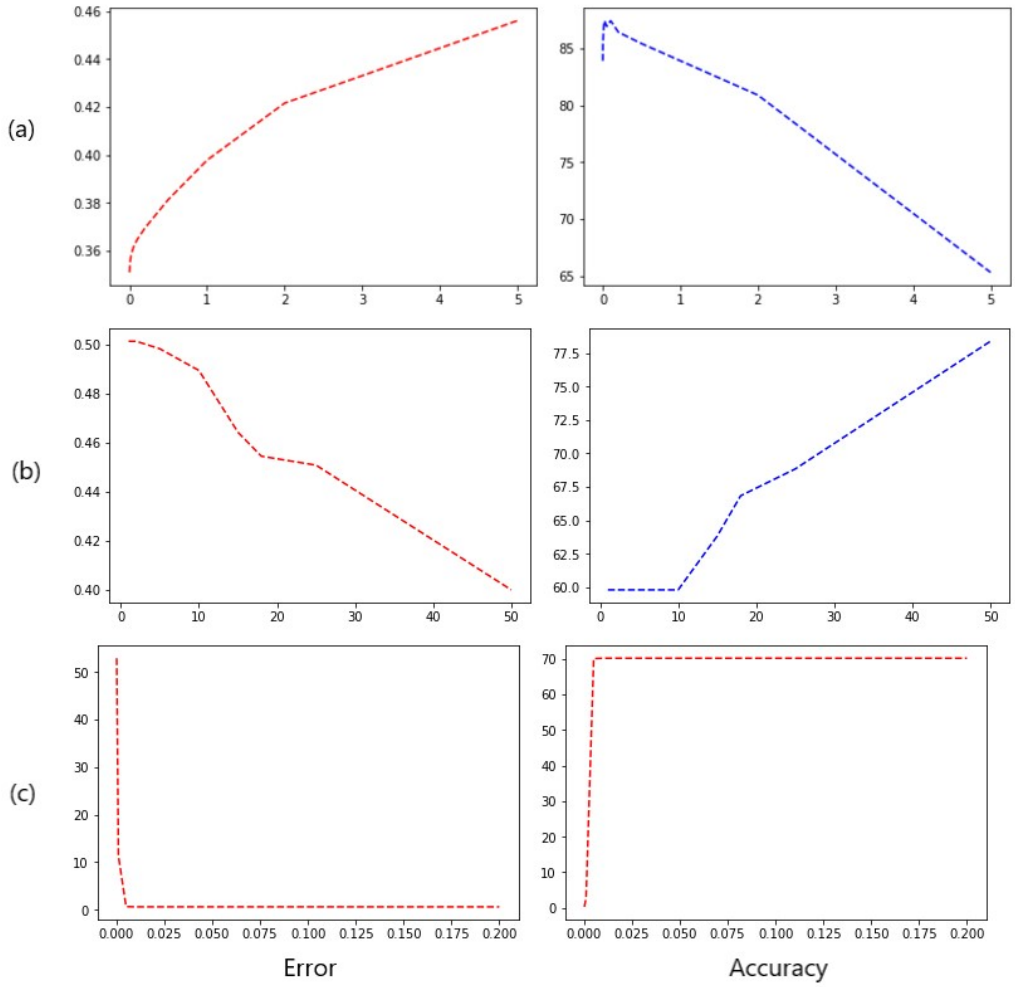

Figure 1: (a) graph of accuracy percentage and error (y-axis) vs λ (x-axis) respectively (b) graph of accuracy percentage and error (y-axis) vs number of basis functions (x-axis) respectively (c) graph of accuracy percentage and error (y-axis) vs η^τ (x-axis) respectively

4. **Optimizer function:** Optimization algorithms help to minimize (or maximize) an Error function which is simply a mathematical function dependent on the Model's internal learnable parameters which are used in computing the target values from the set of predictors used in the model [10].

5. **Activation function:** In artificial neural networks, the activation function of a node defines the output of that node given an input or set of inputs [11].

6 Analysis of Hyper-Parameters

For analysis of the effect of various hyper-parameter, the model is fitted on the testing data obtained by subtraction of feature vectors of Human Observed Feature Dataset to predict labels and compare it with original labels of testing data to determine the accuracy and RMS error. The effect of hyper-parameter on the accuracy and RMS error is shown with the help of graphs.

6.1 Analysis on Regression Models

1. **Regularization Term λ:** The program is executed for different values of λ ranging from 0.001 to 5 and the accuracy and error of the program on the testing data is plotted for each value of λ (Fig.1 (a)). It is observed that the for lower value of λ, the accuracy is low but it increases steadily for some values and then drops. This can be fairly justified by the fact

that when λ is 0, it results in over-fitting of the model due to which the error is high and accuracy is low. As λ increases the accuracy increases.

2. **Number of Basis Functions M:** The program is executed for different values of M ranging from 1 to 50 and the accuracy and error of the program on the testing data is plotted for each value of M (Fig.1 (b)). It is observed that with the increase in the value of M, the error decreases for the model. The more the number of clusters, the lesser would be the variance in the clusters. Therefore, Big sigma would be lower, which means the basis function value would be higher

3. **Learning Rate η^{τ}:** The program is executed for different values of η ranging from 0.001 to 0.2 and the accuracy and error of the program on the testing data is plotted for each value of η (Fig.1 (c)). The accuracy increased as the learning rate was increased but then became constant. Ideally, if the learning rate continues to increase, the error will start to increase [12].

6.2 Analysis on Neural Network Model

1. **Dropout rate of the Neural Network:** The program is executed for different values of dropout rates ranging from 0 to 0.5 and the accuracy of the program on the testing data is plotted for each dropout rate (Fig.2 (a)). It is observed that the higher the dropout rate the lower the accuracy of the program. The best result was obtained for the dropout value of 0.1.

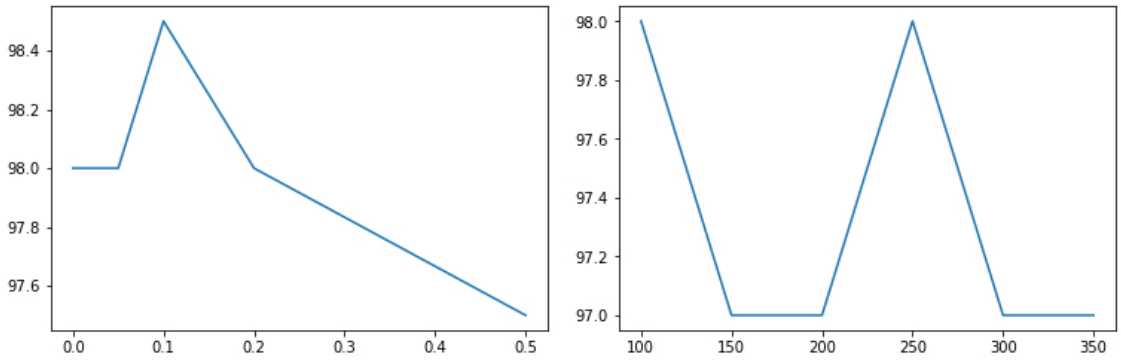

Figure 2: graph of accuracy percentage (y-axis) vs dropout rate (x-axis) and graph of accuracy percentage (y-axis) vs Number of Hidden Nodes (x-axis)

2. **Number of nodes in the hidden layer:** The program is executed for different values of Hidden layer nodes ranging from 100 to 350 and the accuracy of the program on the testing data is plotted (Fig.2 (b)). It is observed that there is no particular correlation between this hyper-parameter and the accuracy of the program. However, some numbers perform better compared to others such as the value of 256 gave higher accuracy during analysis on Human Observed data set.

3. **Input data batch size:** The program is executed for different values of Input data batch size ranging from 16 to 128 and the accuracy of the program on the testing data is plotted. The accuracy remained constant for all different values of batch sizes.

4. **Optimizer function:** The program is executed for different Optimizer functions such as AdaDelta, stochastic gradient, and RMSprop. The model gave an accuracy of 98% for AdaDelta, 97% for RMSprop.

5. **Activation function:** ReLU and Sigmoid activation function were used at hidden layer and Sigmoid at the output layer. The results obtained were similar with either combination. Sigmoid was used at Hidden layer as well as the Output layer of the Neural Network.

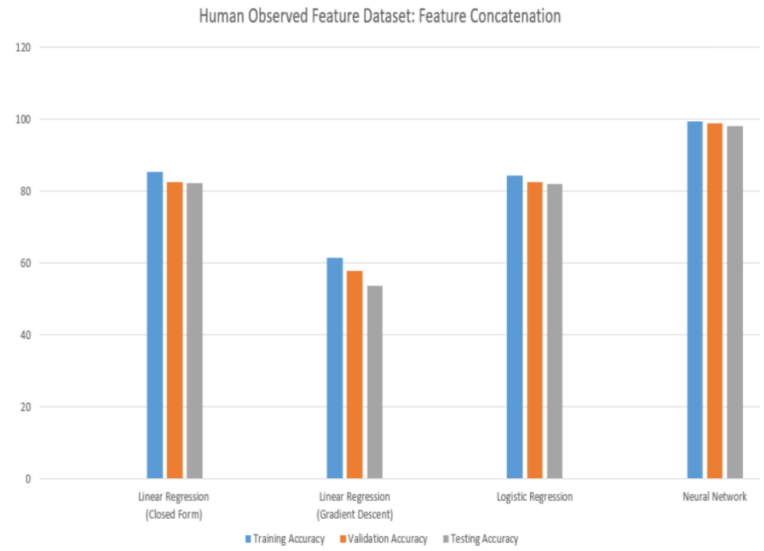

Figure 3: The accuracy of different Machine Learning Models on different data sets

Figure 4: The accuracy of different Machine Learning Models on different data sets

7 Results

In this experiment, the linear regression model using closed form and gradient descent, logistic regression model and Neural Network was implemented for Handwriting Comparison Problem. The accuracy of all models on training, validation, and testing data set of Human Observed Feature Dataset and GSC Feature Dataset for both feature concatenation and feature subtraction is shown with the help of figures and tables. It was observed,

1. **Linear Regression:** The accuracy obtained by performing Linear Regression on both Human Observed Feature Dataset and GSC Feature Dataset were not great for both feature concatenation and subtraction. The Accuracy obtained from Closed Form was good but stochastic gradient descent didn't perform well. The result obtained is shown in Figure 3, Figure 4, Figure 5 and Figure 6 for all datasets and feature vectors.

2. **Logistic Regression:** The accuracy obtained by performing Logistic Regression on both Human Observed Feature Dataset and GSC Feature Dataset were good for both feature concatenation and subtraction. The Accuracy obtained on feature subtraction was around 70% and feature concatenation was around 86% on testing human observed feature dataset. The Accuracy obtained on feature subtraction was around 90% and feature concatenation

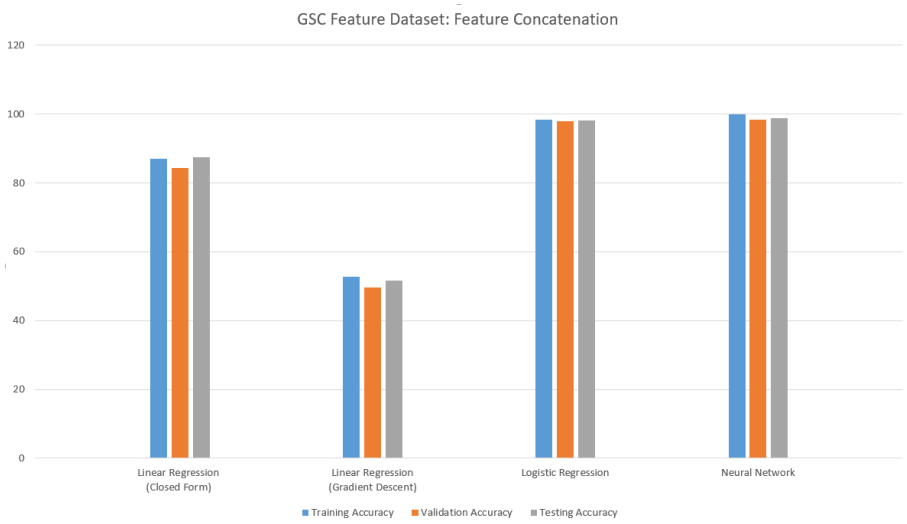

Figure 5: The accuracy of different Machine Learning Models on different data sets

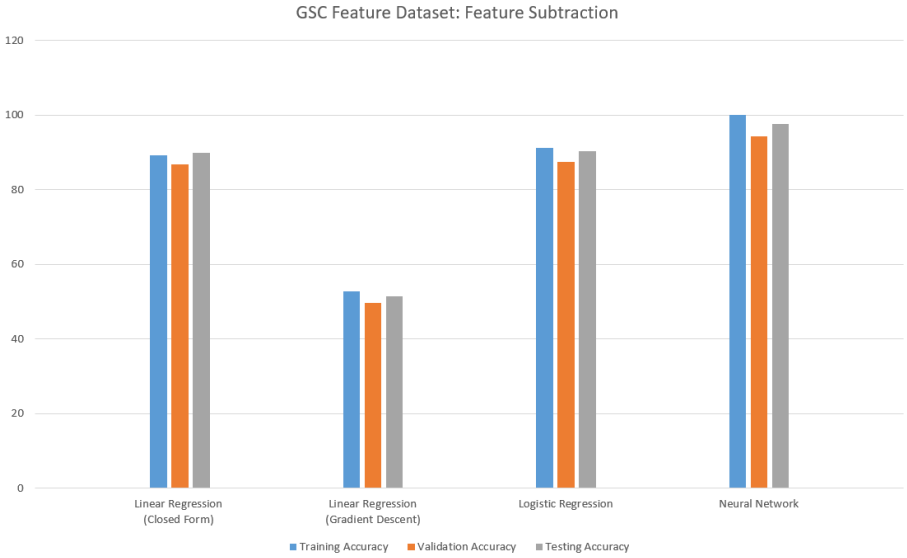

Figure 6: The accuracy of different Machine Learning Models on different data sets

was around 98% on testing GSC feature dataset. The result obtained is shown in Figure 3, Figure 4, Figure 5 and Figure 6 for all datasets and feature vectors.

3. **Neural Network:** The accuracy obtained by fitting Neural Network on both Human Observed Feature Dataset and GSC Feature Dataset were good for both feature concatenation and subtraction. The Accuracy obtained on feature subtraction was around 86% and feature concatenation was around 99% on testing human observed feature dataset. The Accuracy obtained on feature subtraction was around 98% and feature concatenation was around 99% on testing GSC feature dataset. The result obtained is shown in Figure 3, Figure 4, Figure 5 and Figure 6 for all datasets and feature vectors.

Neural Network performed the best for both GSC feature dataset and Human Observed feature dataset. The Neural Network model contained two hidden layers with approximately double the input layers while fitting GSC feature dataset and 256 while fitting Human Observed Feature Dataset. Logistic Regression performed better for GSC Feature Dataset. Results obtained by using feature concatenation were better than feature subtraction in all cases. Root Mean Square Error value obtained from Linear Regression for different datasets and feature vectors are shown in the table given below.

Cases	Training Error	Validation Error	Testing Error
Human Observed Feature: Feature Concatenation - Closed Form	0.34	0.37	0.36
Human Observed Feature: Feature Concatenation - Gradient Descent	0.5	0.51	0.48
Human Observed Feature: Feature Subtraction - Closed Form	0.37	0.39	0.33
Human Observed Feature: Feature Subtraction - Gradient Descent	0.49	0.5	0.47
GSC Feature: Feature Concatenation - Closed Form	0.32	0.35	0.31
GSC Feature: Feature Concatenation - Gradient Descent	0.67	0.69	0.68
GSC Feature: Feature Subtraction - Closed Form	0.30	0.32	0.29
GSC Feature: Feature Subtraction - Gradient Descent	0.57	0.59	0.57

8 Conclusion

In this paper, three Machine Learning method is proposed to solve Handwriting Comparison problem, Linear Regression, Logistic Regression, and Neural network. Several hyper-parameters, such as the learning rate, number of basis function, regularization term, etc. are analyzed for their effect on the accuracy and the root mean square error of the program and the performance of the program is observed. When the above conditions were met, the best accuracy of 60%, 87%. and 98% was achieved on testing data set of Test Human Observed Feature data set for Linear Regression, Logistic Regression and Neural Network respectively. Similarly, the best accuracy of 60%, 98%. and 99% was achieved on testing data set of Test GSC Feature data set for Linear Regression, Logistic Regression and Neural Network respectively. Also, It was observed that the Concatenation of feature vectors performed better than subtraction of feature vectors.

References

[1] Lecture 1: Linear Regression
http://home.iitk.ac.in/m̃ithlesh/cs300/5.pdf

[2] Stochastic Gradient Descent
https://machinelearningmastery.com/implement-linear-regression-stochastic-gradient-descent-scratch-python/

[3] Logistic Regression from scratch in Python
https://medium.com/@martinpella/logistic-regression-from-scratch-in-python-124c5636b8ac

[4] Wikipedia:Hyperparameter (machine learning)
https://en.wikipedia.org/wiki/Hyperparameter_(machine_learning)

[5] Regularization
http://cmci.colorado.edu/classes/INFO-4604/files/slides-6_regularization.pdf

[6] What is the learning rate in neural networks?
https://www.quora.com/What-is-the-learning-rate-in-neural-networks

[7] Dropout in (Deep) Machine learning
https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5

[8] Neural Network Basics
http://www.uta.fi/sis/tie/neuro/index/Neurocomputing2.pdf

[9] Keras Documentation
https://keras.io/models/model/

[10] Types of Optimization Algorithms
https://towardsdatascience.com/types-of-optimization-algorithms-used-in-neural-networks-and-ways-to-optimize-gradient-95ae5d39529f

[11] Wikipedia: Rectifier (neural networks)
https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

[12] Understanding Learning Rates and How It Improves Performance in Deep Learning
https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10