Implementing hand-written digit recognition in Keras with MNIST data-set API.

Name: Shashank Raj
Author
AI Tech & Systems
www.ai-techsystems.com
E-Mail: shashiraj0308@gmail.com

Abstract -- Implementing hand-written digits recognition in Keras with MNIST Dataset API and finding the effect of changing loss function and optimizer algorithm during model compilation.

Keywords -- Deep Learning, Image Recognition, Image Processing, Machine Learning, MINST Data-set, Neural Networks.

I. Introduction

Deep Learning, to a large extent, is really about solving massive optimization problems. A Neural Network is merely a very complicated function, consisting of millions of parameters, that represents a mathematical solution to a problem. Consider the task of image classification. AlexNet is a mathematical function that takes an array representing RGB values of an image, and produces the output as a bunch of class scores.

Todays world is totally filled with sporadic Images, people using Facebook, Instagram, Google and many more Companies has been uploading thousands of Terra-Bytes for storing these images. Hence research in this field has attracted many researchers to research in Image Recognition, Processing and Classification .

In this Project we took a wiser look at processing images using Keras with MINST data-set and finding the effect of changing loss function and optimizer algorithm during Model compilation.

The Optimizers we use here are AdaDelta, Adagrad, Adamax, Nadam, RMSprop, Adam SGD.We will make a robust use of it for predicting the hand-written digits in MNIST data-set and analyze these Optimizers for a better accuracy.

A loss function (or objective function, or optimization score function) is one of the two parameters required to compile a model, we will analyze some loss functions and make predictions for a better accuracy.

By training neural networks, we essentially mean we are minimising a loss function. The value of this loss function gives us a measure how far from perfect is the performance of our network on a given dataset.

II. KERAS

Keras is an open-source neural network library written in Python. It is capable of running on top of Tensor Flow, Microsoft Cognitive Toolkit, Theano, or PlaidML.Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System).

Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel.

In addition to standard neural networks, Keras has support for convolutional and recurrent neural networks. It supports other common utility

layers like dropout, batch normalization, and pooling.

Keras allows users to productize deep models on smartphones (iOS and Android), on the web, or on the Java Virtual Machine.It also allows use of distributed training of deep-learning models on clusters of Graphics Processing Units (GPU) and Tensor processing units (TPU).

III. OPTIMIZERS

An optimizer is one of the two arguments required for compiling a Keras model, We can either instantiate an optimizer before passing it to compilation.

A. *AdaDelta Optimizer*

Adadelta is a more robust extension of Adagrad that adapts learning rates based on a moving window of gradient updates, instead of accumulating all past gradients. This way, Adadelta continues learning even when many updates have been done.

B. *Adagrad*

Adagrad is an optimizer with parameter-specific learning rates, which are adapted relative to how frequently a parameter gets updated during training. The more updates a parameter receives, the smaller the learning rate.

C. *Adam*

The Adam optimization algorithm is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications in computer vision and natural language processing.

D. *Adamax*

AdaMax is an adaptive stochastic gradient descent method, and a variant of Adam based on the infinity norm. In contrast to the SGD, AdaMax offers the important advantage of being much less sensitive to the choice of the hyper-parameters.

E. *RMSprop*

The RMSprop optimizer restricts the oscillations in the vertical direction. Therefore, we can increase our learning rate and our algorithm could take larger steps in the horizontal direction converging faster.

F. *Nadam*

Much like Adam is essentially RMSprop with momentum, Nadam is Adam RMSprop with Nesterov momentum.

G. *SGD*

Stochastic gradient descent optimizer, Includes support for momentum, learning rate decay, and Nesterov momentum.

IV. LOSS FUNCTIONS

A loss function (or objective function, or optimization score function) is one of the two parameters required to compile a model.

A loss function is used to optimize the parameter values in a neural network model. Loss functions map a set of parameter values for the network onto a scalar value that indicates how well those parameter accomplish the task the network is intended to do.

A. *Catagorical Cross Entropy*

Cross entropy loss, or log loss, measures the performance of the classification model whose output is a probability between 0 and 1. Cross entropy increases as the predicted probability of a sample diverges from the actual value. Therefore, predicting a probability of 0.05 when the actual label has a value of 1 increases the cross entropy loss

Mathematically, for a binary classification setting, cross entropy is defined as the following equation:

$$-(y_i \log(p_i) + (1 - y_i) \log(1 - p_i))$$

B. *Kull Back*

The **KullbackLeibler divergence** (also called **relative entropy**) is a measure of how one probability distribution is different from a second, reference probability distribution.

In the simple case, a KullbackLeibler divergence of 0 indicates that the two distributions in question are identical. In simplified terms, it is a measure of surprise, with diverse applications such as applied statistics, neuroscience and machine learning.

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

$$D_{KL}(P||Q) = \int P(x) \log \frac{P(x)}{Q(x)} dx$$

C. Sparse Catogorical Cross Entropy

Cross entropy is a loss function, used to measure the dissimilarity between the distribution of observed class labels and the predicted probabilities of class membership. Categorical refers to the possibility of having more than two classes (instead of binary, which refers to two classes).

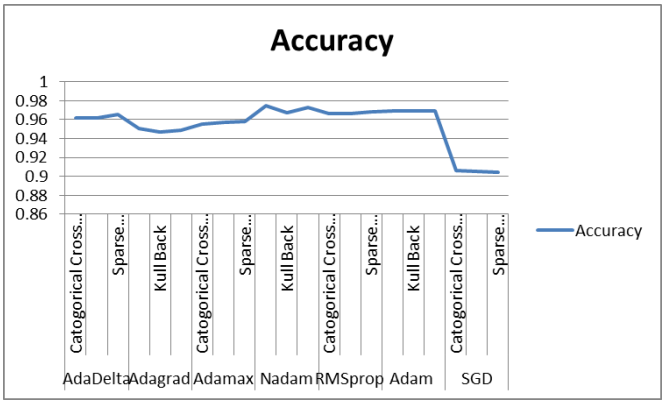
Sparse refers to using a single integer from zero to the number of classes minus one (e.g. { 0; 1; or 2 } for a class label for a three-class problem), instead of a dense one-hot encoding of the class label

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$$

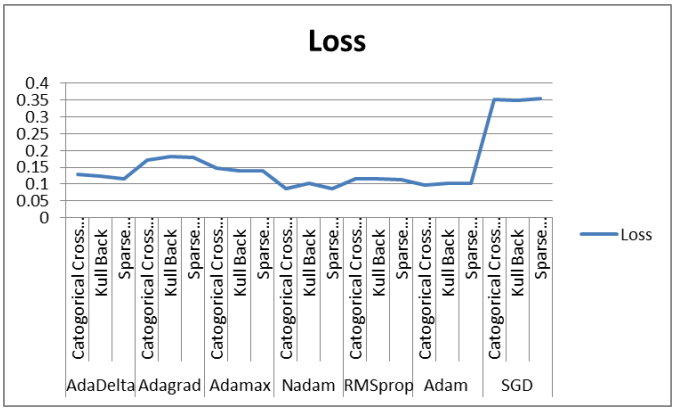
V. TABLES

OPTIMIZER	LOSS FN	ACCURACY	LOSS
AdaDelta	CCE	0.962	0.129
	Kull Back	0.962	0.125
	SCCE	0.965	0.115
Adagrad	CCE	0.951	0.172
	Kull Back	0.947	0.183
	SCCE	0.949	0.179
Adamax	CCE	0.955	0.148
	Kull Back	0.957	0.141
	SCCE	0.958	0.141
Nadam	CCE	0.975	0.086
	Kull Back	0.967	0.103
	SCCE	0.973	0.088
RMSprop	CCE	0.966	0.115
	Kull Back	0.966	0.116
	SCCE	0.968	0.114
Adam	CCE	0.969	0.098
	Kull Back	0.969	0.103
	SCCE	0.969	0.103
SGD	CCE	0.906	0.352
	Kull Back	0.905	0.349
	SCCE	0.904	0.355

VI. ACCURACY



VII. LOSS



VIII. OBSERVATIONS

Nadam Optimizer and Catogorical Cross Entropy has the highest accuracy and lowest loss function value.

IX. INFERENCE

- The best Optimizer for Hand written digit image identification in Keras is **Nadam.**
- The best Loss Function for Hand written digit image identification in Keras is **Catogorical Cross Entropy.**

X. REFERENCES

1. Ayoosh Kathuria, Intro to optimization in deep learning: Gradient Descent, PaperSpace Blog, 1st June 2018.
2. Keras-Website.
3. Wikipedia-Website.

**Abbreviations and Acronyms*

MINST- Modified National Institute of Standards and Technology.

API- Application Program Interface.

NN- Neural Networks

SGD- Stochastic gradient descent

CCE- Catogorical Cross Entropy

SCCE- Sparse Catogorical Cross Entropy