Handwritten Digit Recognition with Neural Networks

- Course Project
CS 419 M: Introduction to Machine Learning

https://github.com/aakash2314/Handwritten-digit-recognition-using-CNN

Abstract

1)Recognizing handwritten digit and processing it This project aims on recognizing handwritten digit and converts them to computer-readable format. We can use CNN directly on a word to classify the word

2)Unsupervised learning, Deep learning, Convolutional neural network

3)Dataset - The Modified National Institute of Standards and Technology dataset (MNIST), It's a collection of 60,000 little square grayscale photos of handwritten single numbers

Description of Task

The first step is to create a model that serves as a baseline.

With five basic parts, we can create this test harness.

- Loading of the dataset
- The preparation of the dataset
- The model definition
- The model evaluation
- The results display

Dataset Employed

MNIST Handwritten Digit Classification Dataset

The Modified National Institute of Standards and Technology dataset is an acronym for Modified National Institute of Standards and Technology dataset.

It's a collection of 60,000 little square grayscale photos of handwritten single numbers ranging from 0 to 9.

The goal is to sort a handwritten digit image into one of ten groups that represent integer values ranging from 0 to 9, inclusively.

Top-performing models are deep learning convolutional neural networks, which achieve a classification accuracy of above 99 percent with an error rate of between 0.4 percent and 0.2 percent

Tasks Performed

Preprocessing dataset

1. We import the photos and rearrange the data arrays so that they only have one colour channel
2. Normalizing the pixel values of grayscale photos
3. Switch the data type from unsigned integers to floats, then divide the pixel values by the maximum value

Tasks Performed

Model Building and Evaluation

1. We can start with a single convolutional layer with a small filter size (3,3) and a small number of filters (32) for the convolutional front-end, followed by a max pooling layer
2. The filter maps can then be flattened to give the classifier features
3. We can add a dense layer between the feature extractor and the output layer to interpret the features
4. Five-fold cross-validation will be used to assess the model
5. The test set for each fold will be used to evaluate the model throughout each epoch of the training run so that learning curves can be created afterwards

Tasks Performed

Present Results

1. The diagnostics of the model's learning behaviour during training and the estimation of the model's performance are the two main features to present.
2. The diagnostics begin with the creation of a line plot depicting model performance on the train and test set throughout each of the k-fold cross-validation folds

Tasks Performed

Depthening of model

1. Modifying the capacity of the feature extraction element of the model
2. changing the capacity or function of the classifier part of the model
3. We can deepen the feature extractor section of the model by following a VGG-like pattern of adding more convolutional and pooling layers

Tasks Performed

Finalizing Model

A final model configuration must be determined and implemented at some time

We'll start by finalising our model by fitting it to the whole training dataset and saving it to a file for future usage

Final Model saving- We can now load and test the final model on the holdout test dataset.

Evaluate Model- We can now load and test the final model on the holdout test dataset.

Make Prediction - We can utilise our previously saved model to forecast new photos.

Results/Analysis

The model accurately predicts sample grayscale images taken from internet.

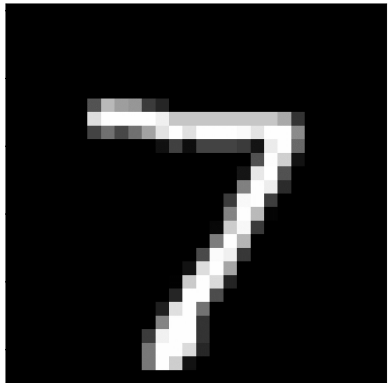
Initially without depthening of model the accuracy was 98.60%

Using a double convolutional layer with 64 filters each, followed by another max pooling layer in this increased the model accuracy.

The model accuracy is increased to around 99.30%

We have tested the model on sample images of 3 and 7

Testing the Model on Sample Images



```
model = load_model('final_model.h5')
predict_value = model.predict(img)
digit = argmax(predict_value)
print(digit)

run_example()
```

```
7
```

```
model = load_model('final_model.h5')
predict_value = model.predict(img)
digit = argmax(predict_value)
print(digit)

run_example()
```

```
3
```

Thank You

Prepared by -

1. Prateek Jha (200040106) - Model Building and Evaluation, Finalizing Model
2. Pratham Kingar(200040075)- Depthening of model, Model Building and Evaluation
3. Gautam Asodiya(200040054)- Model Building and Evaluation, Preprocessing dataset
4. Aakash Jha(200040002) - Depthening of model, Finalizing Model
5. Dylan Rodrigues(200040049)- Model Building and Evaluation Present Results