University of Technology and Economics of Budapest
Faculty of Electrical Engineering and Informatics

Indra Narayan Dutta

HUNGARIAN HELPER

Convolutional Text Detection of Hungarian Characters

(Documentation)

BMEVITMMA19

Deep Learning

BUDAPEST,2023

Contents

1 Introduction

The Hungarian Helper is the first step in my attempt to create an mobile app for translating hungarian texts aimed for international students that come to Budapest. For this semester I created the basic foundation first – the character recognition system that can read hungarian characters from images. In this project, a deep convolutional network is trained that is fed with a large amount of image character dataset and outputs the correct class labels

2 Dataset Creation

This was one of the most challenging aspects due to unavailability of large scale hungarian language data. In order to solve this problem I exploited the fact that hungarian has most of the same character dataset as english except for 6 special characters. I found an online dataset of hungarian hand-written character dataset containing 2000 images per character. I combined the handwritten character with natural scene images from the ICDAR robust reading challenges. The following are the datasets used:

1. bartosgaye_ver1 - handwritten character dataset consisting of characters a to z and 6 Hungarian special characters.
(Datasource: https://github.com/bartosgaye/thedataset)
2. bartosgaye_ver6 - handwritten character dataset of just 6 Hungarian special characters. (Datasource: https://github.com/bartosgaye/thedataset)
3. challenge2 - Scene text English character dataset extracted from ICDAR Robust Reading Challenge (Datasource: https://rrc.cvc.uab.es/?ch=2&com=downloads)
4. icdar2003_chars - Scene text English characters dataset extracted from ICDAR 2003 Challenge (Datsource: http://www.iapr-tc11.org/mediawiki/index.php/ICDAR_2003_Robust_Reading_Competitions)
5. icdar2003_chars_test - Test dataset of the ICDAR 2003 challenge also added

A total of 131,637 Train Images and 32,915 Test Images were obtained

3 Training

Several models were experimented with to determine the best alternative.

Model 1: 5 Layers of Convolution + Batch Norm:

Max Validation Accuracy: **75.37%**

https://wandb.ai/msc_bme/HungarianHelper/reports/Model-1-5-Layer-Conv-BatchNorm--- Vmlldzo2MjIwOTE5

Model 2: 6 Layers of Convolution + Batch Normalization:

Max Validation Accuracy: **76.14%**

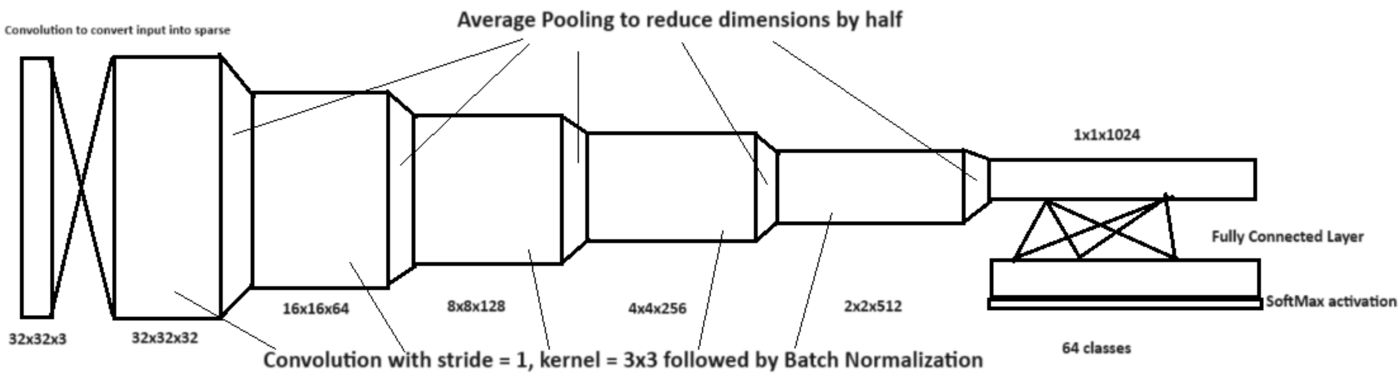
https://wandb.ai/msc_bme/HungarianHelper/reports/Model-2-6-Layer-Conv-BatchNorm-- Vmlldzo2MjIxNTcy

Model 3 (New Model): A new model I wrote that uses 6 Blocks of Convolution + Batch Normalization + Average Pooling

Max Validation Accuracy: **78.79%**

https://wandb.ai/msc_bme/HungarianHelper/reports/Model-3-6-Layer-Conv-BatchNorm-AvgPool--Vmlldzo2MjI0NTcx

3.1 New Model Architecture



3.2 Hyperparameter optimization

Hyperparameter optimization was done manually. Other than the model architecture itself, the following were the hyperparameters found to be ideal

1. image_dims = (3,32,32) -> Sets the dimensions of the input image. The bartosgye character dataset is a 28x28 character dataset so higher dimensions might distort the image.
2. lr = 0.002 -> Learning rate for adam optimizer
3. epochs = 500 -> Total epochs to run training for
4. batch_size = 128 -> Batch size to feed to training

4 Evaluation

Cross Entropy Loss was used for training the model. I used Torch Multiclass accuracy to determine the accuracy of the predicted labels. A Model Checkpoint was created to keep track of the best performing model. Early stopping was also used with a patience of 30 epochs.