CAPTCHA Code Detection

29th April 2024

Ahmed Dider Rahat

BHT Berliner Hochschule für Technik

Studiere Zukunft

Agenda

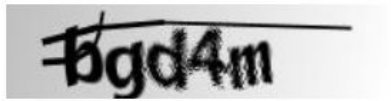
1. Goal of the Project
2. About the Dataset
3. Implementation Process
4. Result Analysis
5. Conclusion
6. References/Links

The Goal of the Project

- The goal of this project is to design and implement a Convolutional Recurrent Neural Network (CRNN) to predict the text content within CAPTCHA code.
- The 1st portion will processing the visual information through convolutional layers to capture the spatial hierarchies in the images.
- Then the 2nd portion will utilizing recurrent layers to decode the sequential information into a string of text.
- The project aims to handle a variety of CAPTCHA styles and complexities with high accuracy.

About the Dataset

- Open source dataset: link
- Properties of the Dataset
  - The dataset contains 1070 captcha images.
  - All the images have the original labels as a file name.
  - Each of the images contains a 5-digit captcha code.
- Example Images:

		
2pfpn	3ny45	bgd4m

Implementation Process

Three phases of the implementation:-

1. Pre-processing
2. Model Training and Prediction
3. Post-processing

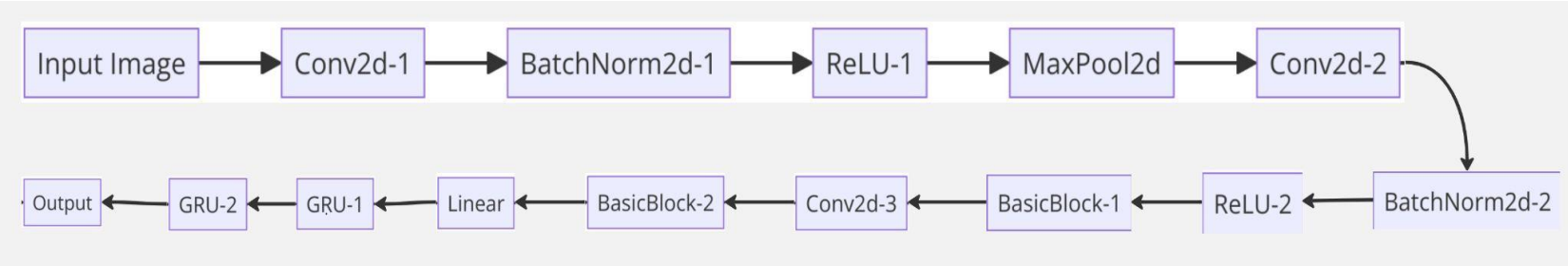
Implementation Process: Pre-Processing

There are several process included in the pre-processing section:

1. Image Loading and Label Extraction
2. Data Set Splitting
3. Creating Data-loader objects

Implementation Process: Model Training

- **Model Architecture:**



- **Component of the Architecture:**
  - **Pretrained ResNet-18 Backbone**
  - **Custom CNN Layers**
  - **Custom RNN Layer**
  - **Linear Transformation**

Model Training (continue…)

Initialization and Training

- **Weight Initialization:**
  - Xavier initialization for linear and convolutional layers
  - Normal distribution for batch normalization layers
- **Training Setup:**
  - Batch Size: 16
  - Number of Epochs: 50
  - Learning Rate and Weight Decay: 0.001
  - Loss Function: CTCLoss
  - Optimizer: Adam
  - Learning Scheduler: ReduceLROnPlateau

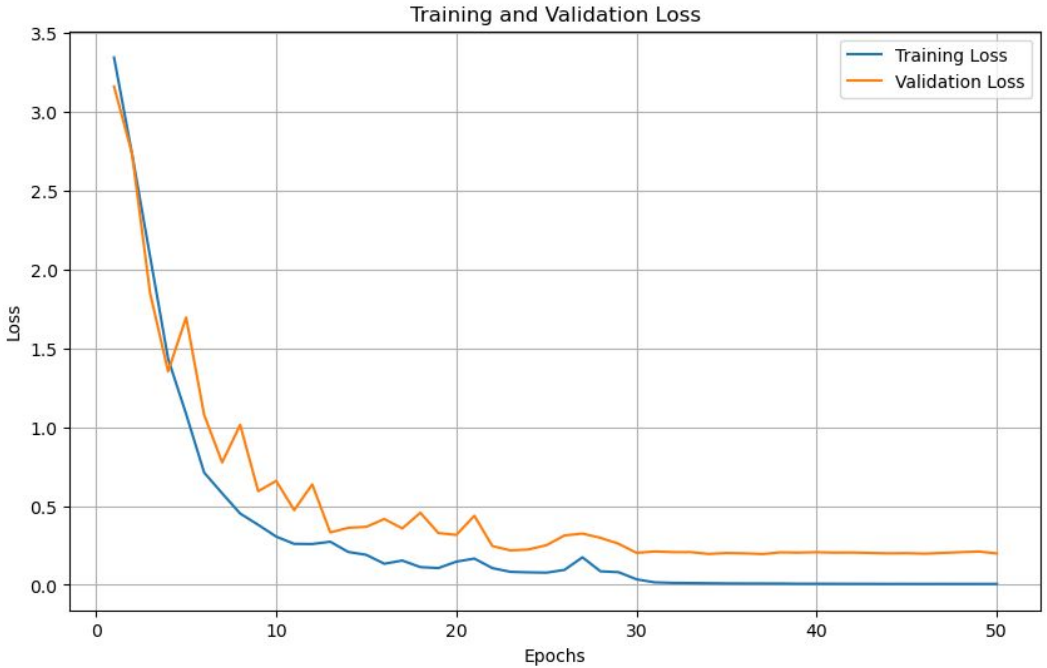
Implementation Process: Post-Processing

Post-processing pipeline:

- Prediction Results Gathering
- Duplicate Character Removal
- Post-processing Correction

Result Analysis

Loss Curve for both training and validation set:



Result Analysis (Continue)

actual	prediction	prediction_corrected
w4cdc	w4----cd-c	w4cdc
n4xx5	n4----xx-5	n4x5
e76n4	e7----6n-4	e76n4
ddcdd	dd--d-cd-d	ddcdd
bbymy	bb----ymmy	bymy

Correct Prediction

actual	prediction	prediction_corrected
n4xx5	n4----xx-5	n4x5
bbymy	bb----ymmy	bymy
yyg5g	yy----g55g	yg5g
emwpn	em-----wpp	emwp
mx8bb	m-x----8bb	mx8b

Wrong Prediction

Model Accuracy

```
----------------------------------------------------------

Model Accuracy on training set:  0.999

Model Accuracy on validation set:  0.806

Model Accuracy on test set:  0.813

----------------------------------------------------------
```

Conclusion

- The model has achieved impressive accuracy of 99.9% on the training set with an accuracy.
- But the performance on the validation and test sets is lower, with accuracies of 80.6% and 81.3%, respectively.
- This suggesting a gap in the model's ability to generalize to unseen data compared to its performance on the training data.

References/Links

1. https://www.kaggle.com/datasets/fournierp/captcha-version-2-images
2. Project Link: https://github.com/AhmedDiderRahat/captcha_detection/tree/main

Thank You