CS 4824 Project Milestone Report

Chenzhuoer Li
Department of Computer Science
Virginia Tech
Blacksburg, VA 24060
lchenz98@vt.edu

Zike Yu
Department of Computer Science
Virginia Tech
Blacksburg, VA 24060
zekeyu@vt.edu

Zinan Guo (Withdrawn from course)
Department of Computer Science
Virginia Tech
Blacksburg, VA 24060
zinan@vt.edu

Abstract

Our project aims to develop a deep learning method based on convolutional neural networks, using the TensorFlow framework to automatically identify CAPTCHA programs through machine learning. The project used one million CAPTCHA image examples for training. In the training process, it is found that the recognition accuracy rate is significantly improved, while the loss rate is significantly reduced. We finally selected the result of the 67th epoch as the final result, and its accurate prediction accuracy reaches 90.4%, and the accuracy rate of ignoring the upper-lower case is 95.2%.

1 Preface

In this semester's machine learning course, we will conduct a project. In this project, we will apply the knowledge we have learned in class and outside class to practice. This group chose to apply machine learning technology in the field of image recognition. Develop a program that can automatically recognize CAPTCHA.

2 Preparation

In order to carry out this project, our project team made the following preparations.

2.1 Development Environment

Language: Python 3.6

IDE: JetBrains PyCharm Community Edition

Framework: TensorFlow

Package Include: numpy 1.19.2, os, captcha, opencv-python 4.4.0.44, tensorflow 1.14.0, progressbar2, matplotlib 3.3.3

2.2 Background Knowledge

In order to complete this project, in addition to using the knowledge learned in the classroom, our project team also self-studied the following:

CAPTCHA Package: Learn the CAPTCHA package and its API, use the functions it provides, and generate the CAPTCHA images required for training.

TensorFlow Framework: Understand the role and usage of the entire TensorFlow framework, and learn to use its API to implement algorithm functions.

OpenCV Framework: Understand the role and usage of the entire OpenCV framework, and learn to use its API for image recognition, conversion, and processing.

Deep Learning: Understand the basic principles and algorithms of deep learning.

Convolutional Neural Networks: Understand the basic principles and algorithms of convolutional neural networks.

Long Short-Term Memory: Understand the basic principles and algorithms of LSTM, use for CAPTCHA images split.

3 Method

Our research group aims to use the TensorFlow[1] framework for overall software development. The program algorithm will be based on the convolutional neural network algorithm. Convolutional Neural Networks is a deep learning model or multilayer perceptron like an artificial neural network, which is often used to analyze visual images. The convolutional neural network architecture is very similar to the conventional artificial neural network architecture, especially in the last layer of the network, which is fully connected. In addition, notice that the convolutional neural network can accept multiple feature maps as input instead of vectors [2]. Since the last milestone report, we have additionally used the Long Short-Term Memory algorithm to cut the characters in the CAPTCHA image [3]. Solve the problem that the result of recognizing characters in CAPTCHA images is out of order.

Our research group will write a program to generate CAPTCHA pictures to obtain the training set, validation set, and test set for machine learning. The overall accuracy for the final goal is expected to achieve 80%.

In order to complete this project, our project team chose to use the convolutional neural network algorithm and designed the following process (Figure 1).

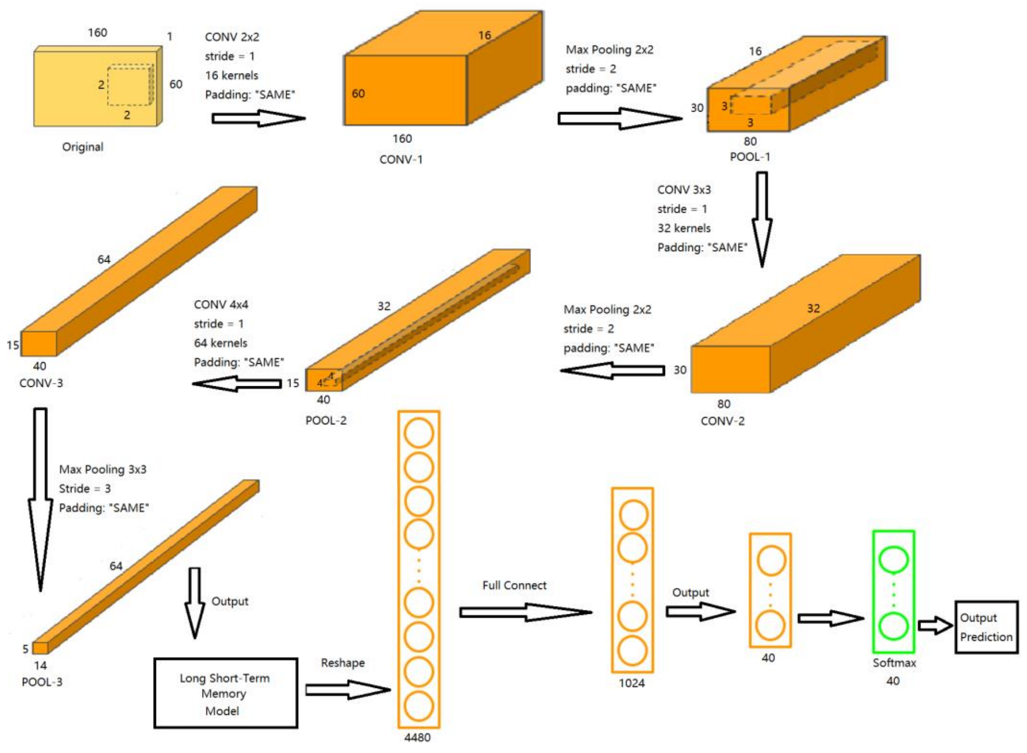

Figure 1: Project Algorithm

According to our algorithm, the original image size is 160x60. The data will go through three times of convolution and three times of max pooling, the long short-term memory algorithm, and output the final prediction results after reshape, full connection, etc.

4 Progress

In order to achieve the project goals, our project team took the following steps.

4.1 Configuration Environment

We first configured the Python environment used to develop this project according to the packages used in this project mentioned above.

4.2 Pre-Program Development

In order to obtain a sufficient number of CAPTCHAs for machine learning, we did not choose to directly collect CAPTCHA pictures from online but wrote a program to generate CAPTCHA pictures.

4.3 CAPTCHA Image Generation

We use the written program to generate CAPTCHA pictures. In the progress of the project, we were generating data, doing machine learning, and optimizing algorithm methods in the same time. The total amount of data generated depends on the learning results. In the initial proposal, we plan to generate more than ten millions pictures, but we finally only generated one million pictures for training and achieved our goal.

4.4 Program Development

We use the above algorithm as the core machine learning program, and add additional functions for reading in data, processing data, generating prediction results, saving machine learning results, and reading previous round's machine learning results.

5 Result

After training with a training set of one million CAPTCHA image examples, we got the following results.

5.1 Overall Progress

After 71 rounds of training, we believe that the results have converged and stopped subsequent machine learning. For the first 71 rounds of learning, we visualized the accuracy rate and loss value on the training set and validation set, and obtained the following results (Figure 2).

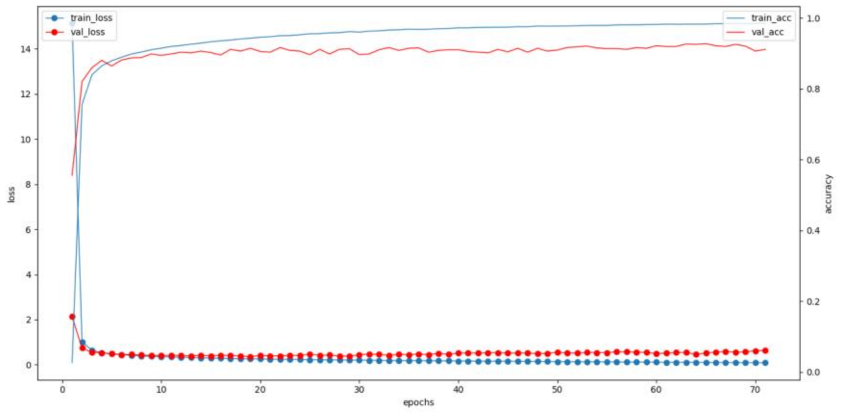

Figure 2: Loss Value and Accuracy Visualization

From the figure, we can see that based on the training results of one million CAPTCHA image samples, the recognition accuracy rate has been significantly increased after the first round, and the loss rate has dropped significantly. And in the subsequent training, the accuracy rate continued to rise, and finally stabilized.

5.2 Outcome Selection

Based on the results in the above figure, we finally selected the checkpoint value (.ckpt files) from Epoch 67th. We generated 100,000 CAPTCHA images for the test set. In the test set, the training result achieved recognition accuracy of 90.4%. If the upper and lower case is ignored, the recognition accuracy rate reaches 95.2%.

5.3 Incorrect Sample Analysis

Through the analysis program, we loaded the predicted results to obtain the recognition accuracy rate and printed out all the results of the incorrect predictions. Regarding the results of these incorrect predictions, we have analyzed the main types and reasons. We believe that there are two main types of errors, one is the Confusion of Similar Characters, and the other is the Overlap of Character Patterns.

5.3.1 Confusion of Similar Characters

The main reason for the incorrect results caused by such reasons is that in English characters, many characters have similar appearances.

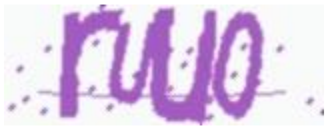

Figure 3: Confusion of Similar Characters Example 1

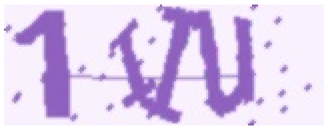

Figure 4: Confusion of Similar Characters Example 2

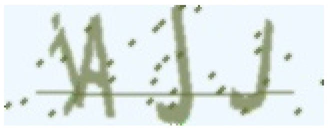

Figure 5: Confusion of Similar Characters Example 3

In figure3, we can see that the label of the picture itself is “rUu0” and the result predicted by our model is “rUuO”. This is because the capital letter "O" in the English alphabet is very similar to the number "0" in appearance.

In figure4, we can see that the label of the picture itself is “1IIu” and the result predicted by our model is “1Ilu”. This is because the uppercase letter "O" and the lowercase letter "l" in the English alphabet are very similar in appearance.

In figure5, we can see that the label of the picture itself is “jAJJ” and the result predicted by our model is “lAJJ”. This is because the lowercase letter "j" in the English alphabet and the number "1" are very similar in appearance.

The above three examples are all due to the similarity of characters, which leads to confusion in recognition. But such confusion is not always happening, and can still be correctly recognized in most cases. It's just that compared to other letters and numbers, this kind of letters and numbers with similar appearance has a higher incorrect recognition rate.

5.3.2 Overlap of Character Patterns

For this type of incorrect recognition result, the main reason is that there is a large amount of overlap of character patterns in the CAPTCHA picture.

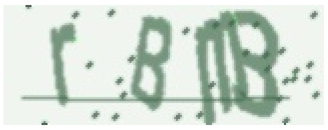
Figure 6: Confusion of Similar Characters Example 1

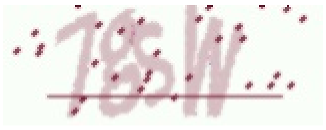
Figure 7: Confusion of Similar Characters Example 2

In figure6, we can see that the label of the picture itself is “rBm3” and the result predicted by our model is “rBmB”. This is because the number "3" coincides with the rightmost of the letter "m", which results in the number "3" being recognized as a capital letter "B" during recognition.

In figure7, we can see that the label of the picture itself is “7gSw” and the result predicted by our model is “7gw”. This is due to the partial overlap of the letter "g" and the capital letter "S", resulting in a loss in the process of character splitting, and the capital letter "S" is not recognized.

The above two examples are due to the overlapping of adjacent character patterns in the picture, which leads to poor splitting effect during recognition, and then produces wrong results. Similarly, in most cases of such overlapping CAPTCHAs can still be correctly identified. It's just that compared with other CAPTCHAs that do not overlap, the incorrect recognition result rate of this CAPTCHA with graphics overlap is higher.

5.3.3 Other Incorrect Results

For this type of incorrect recognition results can be caused by multiple reasons.

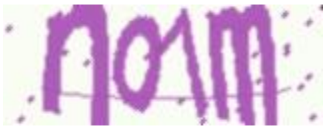
Figure 8: Confusion of Similar Characters Example 3

In figure8, we can see that the label of the picture itself is “no7m” and the result predicted by our model is “rno7m”. This may be due to the fact that the left half of the lowercase letter "n" was repeatedly recorded when it was split, and then it was considered as an additional lowercase letter "r".

For this kind of incorrect results, we do not have a particularly good explanation, and can only be regarded as an inevitable random error in machine learning.

6 Conclusion

In general, we used the deep learning method based on convolutional neural network to complete our CAPTCHA Automatic Recognizer project. The correct rate exceeded the planned 80%, reaching 90.4%. And the accuracy rate for ignoring upper & lower case reaching 95.2%. Overall, I’m pretty satisfy with this project’s result. This project also allowed us to learn a lot, and at the same time let us put what we learned in class into practice.

References

[1] https://www.tensorflow.org/overview

[2] Aegeus Zerium, *"Demystifying Convolutional Neural Networks"*, Sep 2, 2018, URL: https://medium.com/@eternalzer0dayx/demystifying-convolutional-neural-networks-ca17bdc75559

[3] S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural Computation, 9(8):1735–1780, 1997.