Oracle Bone Script Character Recognition

Data Incubator Capstone Project Proposal

Xiaochen Liu
University of California, Davis

Contact Information:
Department of Mathematics
University of California
Davis, CA

Phone: +1 (530) 304 1047
Email: xchliu@ucdavis.edu

Introduction

Oracle bone script was the ancient form of Chinese characters inscribed on animal bones or turtle shells. It is highly pictographic, meaning that it uses pictorial symbols to represent words.

While the earliest oracle bone script can be traced back to 1200 BC, some of the writing style evolved from it is still widely used for seal engraving and calligraphy, for example, the **seal style**.

As a subclass of oracle bone script, seal style maintains the pictographism and most of the characters look totally different as modern simplified Chinese characters. Although it is depicted in a pictographic way, most of them are not realistically enough for starters to recognize what they stand for.

Figure 1: The Chines character "water" in oracle bone script (seal style) and regular script.

This unreadability causes problems to calligraphy or seal engraving learners, e.g., the author himself. As a consequence, our project aims to train models and help recognize oracle bone script.

Main Objectives

Stage I. Train models to recognize single characters.

Stage II. Train models to detect character positions on a seal and recognize the characters separately.

Data

Although the project is primarily motivated by recognition of seal scripts, we collect the larger class of data, i.e., the oracle bone scripts. The two styles coincide on most of the characters, while the oracle bone script has more abundant amount of data samples.

No existing downloadable database is known. The majority of our data are scrapped from Chinese Etymology.

We plan to collect the bone scripts of the most commonly used 1000 characters. Under each character, we are able to collect about 100 different training samples, i.e., 100 RGB pictures, with size about 5 mb. We resize the images and transform the RGB format to greyscale. By shifting, rotating, shearing the images, we may augment the training data to 1000+ images for each character.

Methodology

While we have trained our classifier (on 10 characters) using logistic regression, K-nearest-neighbors, we decide to complete this project using convolutional neural network. The architecture we used to train the small model is the LeNet 5, which is widely used to train the MNIST database of handwritten digits and other pattern recognition problem. Deeper NN, e.g., AlexNet, will be considered in training the real model.

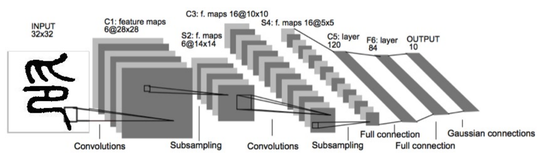


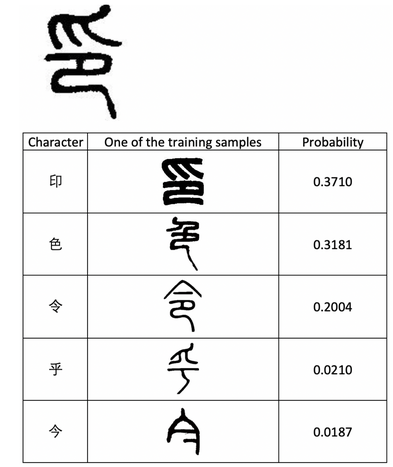
Figure 2: Architecture of LeNet 5 on 10 classes

Stage I: Current Result

We obtain the following consequences of classification on n characters, for $n = 10$, 30 and 50 and their top 1, 3 and 5 accuracy. The following table gives the prediction probabilities on a single character, which gives the desired results.

# Char.	Top 1	Top 3	Top 5
10	.71	.89	.96
30	.46	.55	.64
50	.65	.70	.88

Table 1: Classification accuracy.



Character	One of the training samples	Probability
印		0.3710
色		0.3181
令		0.2004
乎		0.0210
今		0.0187

Figure 3: Prediction probability on a single character.

Stage II: Expected Result

In the last, we present a rendering of the expected result. Given an seal, we detect the positions of the characters using CNN. Then feed each character into the model we trained in Stage I and recognize them separately.

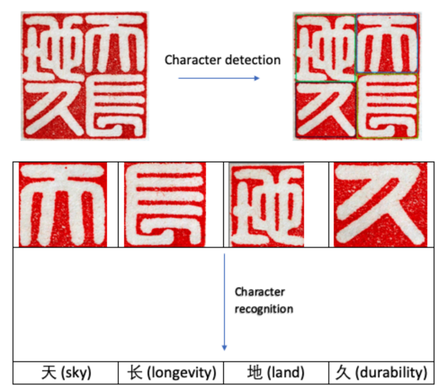


Figure 4: Expected output on Stage II.