The Final Project

Yanwei Fu

May 7, 2022

Abstract

(1) This is the final project of our course. The deadline is 5:00 pm, June 15, 2022. Please upload the report via elearning.

(2) The paper must be in NIPS format (downloadable from [1]). We do not need the double-blind review.

(3) The goal of your write-up is to document the experiments you've done and your main findings. So be sure to explain the results. Generate a single pdf file of your projects and turned in along with your code. Package your code and a copy of the write-up pdf document into a zip or tar.gz file and named as Final-Project-student-id1-student-id2.[zip|tar.gz]

(4) You are open to using anything to help you finish this task.

(5) About the deadline and penalty. In general, you should submit the paper according to the deadline of each mini-project. The late submission is also acceptable; however, you will be penalized 10% of scores for each week's delay. **The submission can only be delayed for up to three days.**

(6) For all the projects, we DO care about the performance on each dataset with the correct evaluation settings.

Note that:

(a) If the size of training instances is too large, you may want to apply some sampling techniques to extract a small portion of training data.

(b) If you think the dimension of features is too high, you may also want to use some techniques to do feature dimension reduction.

(c) The referring papers are listed as an introduction to the context of problems. It's not necessarily to exactly implement these papers, which is not an easy task.

1 Introduction

1.1 Collaboration Policy

You are allowed to work in a group with at most one collaborator. You will be graded on the creativity of your solutions, and the clarity with which you can explain them. If your solution does not live up to your expectations, then you should explain why and provide some ideas on how to improve it. You are free to use any third-party ideas or codes that you wish as long as it is publicly available. You must provide references to any work that is not your own in the write-up.

[1] `https://nips.cc/Conferences/2016/PaperInformation/StyleFiles`

1.2 Writing Policy

The final project (20%) is finished by one team. Each team should have up to 2 students. You will solve a real-world Big-Data problem. The final report should be written in English. The main components of the report will cover

1. Introduction to the background and potential applications (2%);
2. Review of the state-of-the-arts (3%);
3. Algorithms and critical codes in a nutshell (10%);
4. Experimental analysis and discussion of proposed methodology (5%).

Please refer to our latex example[2].

1.3 Submitting Policy

The paper must be in NIPS format. Package your code and a copy of the write-up pdf document into a zip or tar.gz file called Final-Project-student-id1-student-id2.[zip|tar.gz]. Include functions and scripts that you had used. Upload them to eLearning. In the submitted documents, you should include a readme.txt file to well explain the authors and co-workers of this project. The TAs can know the names of your works.

1.4 Evaluation of Final Projects

We will review your work on the following NIPS criteria:

Overview: You should briefly summarize the main content of this paper, as well as the Pros and Cons (advantages and disadvantages) in general. This part aims at showing that you had read and at least understand this paper.

Quality: Is the paper technically sound? Are claims well-supported by theoretical analysis or experimental results? Is this a complete piece of work, or merely a position paper? Are the authors careful (and honest) about evaluating both the strengths and weaknesses of the work?

Clarity: Is the paper clearly written? Is it well-organized? (If not, feel free to make suggestions to improve the manuscript.) Does it adequately inform the reader? (A superbly written paper provides enough information for the expert reader to reproduce its results.)

Originality: Are the problems or approaches new? Is this a novel combination of familiar techniques? Is it clear how this work differs from previous contributions? Is related work adequately referenced?

Significance: Are the results important? Are other people (practitioners or researchers) likely to use these ideas or build on them? Does the paper address a difficult problem in a better way than previous research? Does it advance the state of the art in a demonstrable way? Does it provide unique data, unique conclusions on existing data, or a unique theoretical or pragmatic approach?

[2]`http://yanweifu.github.io/courses/SLML/chap5/IEEE_TAC_2016.zip`

1.4.1 Minimum Requirements

For all the projects listed below, in general you should devise your own deep learning models which target each specific problem of each project. You should compare with the machine learning algorithms taught in this course/projects. Thus, the minimum requirements, as you can imagine, just apply and compare with these methods; and explain the advantages and disadvantages of using these methods for the project problem. Note that, your algorithms can be derived from one of these existing deep learning algorithms; and feel free to use any machine learning packages you like.

2 Unsupervised Few-Shot Oracle Character Recognition

The classical few-shot learning aims at learning new visual concepts efficiently from extremely few labeled examples by transferring the knowledge from the large-scale labeled source data. Here, the task aims at utilizing large-scale *unlabeled* source data, including unlabeled oracle or other ancient Chinese characters, to facilitate novel oracle characters recognition. In this task, you are provided the few-shot Oracle-FS and Unlabeled Source Data [1] both in pixel and sequence format (downloadble from [3]). You can use all source data and training data from Oracle-FS to train your model and evaluate it on test data of Oracle-FS. Please note that even though some source data are provided with labels, you should not utilize them.

2.1 Background Introduction

Oracle characters are the earliest known hieroglyphs in China, which were carved on animal bones or turtle plastrons in purpose of pyromantic divination of weather, state power, warfare and, trading to mitigate uncertainty in the Shang dynasty [3]. Due to the scarcity of oracle bones and the long-tail problem in the usage of characters as shown in Fig. 1, oracle character recognition suffers from the problem of data limitation and imbalance. Many oracle characters have extremely limited samples, thus oracle character recognition is a natural few-shot learning problem, which is topical in computer vision and machine learning communities, recently. Besides, there is a high degree of intra-class variance in the shapes of oracle characters, resulting from the fact that oracle bones were carved by different ancient people in various regions over tens of hundreds of years. As a result, oracle character recognition or classification is a challenging task.

In the past decades, although identification and decipherment for oracle characters have made huge strides, there is still a long way to fully understand the whole writing system. So far, more than 150,000 bones and turtle shells had been excavated, including approximately 4,500 unique oracle characters. Only about 2,000 of them have been successfully deciphered[2], which means there exists large-scale unlabeled/undeciphered oracle characters.

Thus, different from standard few-shot learning setting, the task does not assume the existence of large-scale labeled source oracle characters. Formally, you have to study under a more practical setting, where your model has access to large-scale unlabeled source oracle characters, in order to efficiently recognize new oracle characters containing few labeled training examples.

2.2 Dataset Description and Train/Test Split

Oracle-50K is collected from three data sources using different strategies, with 2,668 unique characters and 59,081 images. Based on Oracle-50K, we create a few-shot oracle character recognition dataset under three different few-shot settings. Under the k-shot setting, there are 200 classes, with k training instances and 20 test ones per class.

[3]http://www.sdspeople.fudan.edu.cn/fuyanwei/course/projects/final_project/oracle/

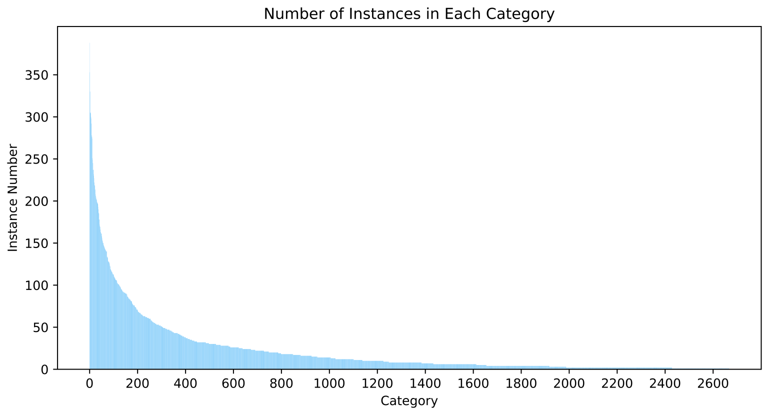

Figure 1: The distribution of oracle character instances in Oracle-50K. Categories are ordered by number of instances.

In this paper, we set k=1, 3, 5. We use the remained oracle character examples and other collected ancient Chinese character images as unlabeled source data, including 276,031 images.

The stroke orders of Chinese characters contain a lot of information, for which people can usually recognize a character correctly even if it is being written or incomplete. Characters can be intrinsically expressed in strokes. Although the stroke orders of oracle characters have been lost in history, there are two fundamental facts: 1) oracle writing or Shang writing is ancestral to modern Chinese script; 2) the modern Chinese writing system is in a left-to-right then top-to-bottom writing order. We assume oracle character writing is in the same order. Using existing online approximation algorithm in [4], we could convert offline character images in pixel format to online data in vector format. Both pixel and vector format data are provided in this project. Note that due to 3 failure cases during online approximation, the number of source samples in sequence format are 276,028.

2.3 Evaluation Protocol

You could use all source data and few-show training samples to train your model and evaluate it on the test data of Oracle-FS. Since both pixel and sequence format data are provided, you could use either one or both of them to train and evaluate your model. Please report the Top-1 accuracy for all 200 classes under all three few-shot setting.

2.4 Other Projects

You can also try other projects. However, please let TA and me know first to get the approval. Note that if the project is too easy, it would affect your scores of final projects.

2.5 Appendix

```
# A simple case for .npz file loading and sample visualization

import numpy as np
import random
```

```python
import matplotlib.pyplot as plt

npz_path = 'xxx.npz'
data = np.load(npz_path, encoding='latin1', allow_pickle=True)
train_data = data['train']
test_data = data['test']

sample = np.random.choice(test_data, 1)[0]

def strokes_to_lines(strokes):
  """Convert stroke-3 format to polyline format."""
  x = 0
  y = 0
  lines = []
  line = []

  for i in range(len(strokes)):
      if strokes[i, 2] == 1:
          x += float(strokes[i, 0])
          y += float(strokes[i, 1])
          line.append([x, y])
          lines.append(line)
          line = []
      else:
          x += float(strokes[i, 0])
          y += float(strokes[i, 1])
          line.append([x, y])
  return lines

def show_one_sample(strokes, linewidth=100):
  lines = strokes_to_lines(strokes)
  for idx in range(0, len(lines)):
      x = [x[0] for x in lines[idx]]
      y = [y[1] for y in lines[idx]]
      plt.plot(x, y, 'k-', linewidth=linewidth)

  ax = plt.gca()
  plt.xticks([])
  plt.yticks([])
  ax.xaxis.set_ticks_position('top')
  ax.invert_yaxis()

 # show
 plot(sample)
```

References

[1] Wenhui Han, Xinlin Ren, Hangyu Lin, Yanwei Fu, and Xiangyang Xue. Self-supervised learning of orc-bert augmentor for recognizing few-shot oracle characters. In *ACCV*, 2020.

[2] S. Huang, H. Wang, Y. Liu, X. Shi, and L. Jin. Obc306: A large-scale oracle bone character recognition dataset. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 681–688, 2019.

[3] David N Keightley. Graphs, words, and meanings: Three reference works for shang oracle-bone studies, with an excursus on the religious role of the day or sun, 1997.

[4] Martin Mayr, Martin Stumpf, Anguelos Nikolaou, Mathias Seuret, Andreas Maier, and Vincent Christlein. Spatio-temporal handwriting imitation. *arXiv preprint arXiv:2003.10593*, 2020.