Stroke Augmenter and Self-supervised Learning: the Better Way to Recognize Few-Shot Oracle Characters

Haitian Jiang
School of Data Science
Fudan University
19307110022@fudan.edu.cn

Leiru Long
School of Data Science
Fudan University
19307130350@fudan.edu.cn

Abstract

Oracle character recognition is critical for archaeology, ancient text comprehension, historical chronology and other applications. We first train SVM and pure CNN models to see how traditional machine learning methods perform on this task. In order to enhance the few-shot performance, we proposed a simple yet powerful data augmentation method, Stroke Augmenter: tiny translations are exerted on each stroke in the sequence data extracted from online approximation to add versatility to characters. Even a vanilla CNN model trained on this augmented data can already beat the state-of-the-art model. Due to the poor quality of the sequence data, direct classification using sequence models like Sketch-BERT does not produce results as expected. So further improvement is still focused on image data. We add self-supervised learning to the CNN models to make them more expressive. With model ensemble, we finally achieve the accuracy of 85.47% on the 5-shot task.

1 Introduction

The Oracle character is one of the earliest hieroglyphics, dating back to China's Shang Dynasty. Oracle characters widely documented human activities throughout that time period, serving as both a foundation for studying Chinese etymologies and a window into Chinese history. If we can use machine learning to identify oracle characters, then we can classify more unlabeled oracle characters and help archaeologists with their researches.

Unlike general character recognition, oracle characters have the distinct feature that the amount of samples varies greatly between categories and is woefully inadequate in comparison to the number of categories. Considering the imbalance and insufficiency of oracle characters, it's instinctive to take oracle characters recognition as a few-shot learning task which is similar to the real-word archaeology scenario.

Since models trained with small samples can easily fall into overfitting to training samples, there are three mainstream methods to overcome this problem [1]:

1. Augment training data set by prior knowledge;
2. Constrain hypothesis space by prior knowledge;
3. Alter search strategy in hypothesis space by prior knowledge.

However, we have no prior knowledge about oracle characters to confine hypothesis space. Instead, we know that every Chinese character consists of strokes and the relative position of each stroke does not affect the recognition of characters. This prior knowledge can be used to design our data augmentation method. What's more, the search strategy can be altered from training a vanilla CNN to combining self-supervised learning into it.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

Inspired by Han et al. [2] who simply assumed oracle character writing is in the same order as the modern Chinese writing and generated pseudo online stroke data, we can derive a more simple and powerful data augmentation method from strokes. We also self-supervised learning and model ensemble [5] to further improve the accuracy. The followings are our main contributions:

- We implement various learning models using image and sequence data to do few-shot oracle characters recognition and compare them, including SVM, multiple CNN structures, and a fine-tuned Sketch-BERT [4] pre-trained on unlabeled sequence data.
- We propose Stroke Augmenter as a simple and powerful data augmentation method in oracle character few-shot learning scenario. When combined with a DenseNet161 classifier to do the recognition, we can achieve new state-of-the-art results.
- Based on augmented samples, we use self-supervised learning and model ensemble to further improve the performance.

2 Related Work

2.1 Oracle Dataset

Oracle-50K is a dataset collected by Han et al. [2]. It contains in total 2668 classes of 59081 oracle character instances, collected from three openly available data sources.

Based on Oracle-50K, **Oracle-FS** is composed as a few-shot benchmark containing 1, 3, and 5 shots tasks using oracle character images from Oracle-50K. The test set contains 20 instances of the same oracle character, and the number of classes (different characters) sum up to 200.

Except for oracle character images, sequence data of each character also come along with the Oracle-FS benchmark dataset. The sequence data is organized in a 3-element vector format, with a 2-dimensional continuous value for the position and a binary value for the state. As a result, a character is represented as a series of dots, each of which has three attributes: $O_i = (\Delta x, \Delta y, p)$, where Δx, Δy are the horizontal and vertical distances between two neighboring points and p represents whether this point is the end of a stroke.

2.2 Online Sequence Approximation

As offline oracle character images do not contain temporal annotations, if we want the online sequence approximation, we need a heuristic approach to analytically synthesis online data. The sequence data in Oracle-FS uses the method brought by Mayr et al. [3] to compose the online sequence in a left-to-right and top-to-bottom fashion.

However, the model from Mayr et al. [3] works for English characters, so it does not faithfully produce the outcomes on Chinese characters, let alone oracle ones. And even in the English setting, it generates many unexpected result, as shown in Figure 1, demonstrating the algorithm's failure to interpret lines put on top of each other and to connect crossing lines, resulting in the continuous line being divided into several segments.

Therefore, when it comes to using this model on oracle characters, only low quality results like Figure 2 can be expected. This problem leads to our discarding sequence data as input for classification and focusing only on image data.

2.3 Sketch-BERT

Sketch-BERT [4] is adapted from BERT [6] to do recovery on sketch sequence data (Gestalt task). Like the BERT model used in natural language processing which use masked language model to learn the token representation, Sketch-BERT use masks to cover the sequence data of sketches to enable the model to learn how to recover the masked sketch. The difference lies in that Sketch-BERT removes the next sentence prediction task in BERT, like what RoBERTa do. The embeddings of Sketch-BERT input is constructed by point, positional and stroke embeddings.

Figure 1: Failure cases in the sequence approximation of English characters

Figure 2: Generated sequence data cannot faithfully reflect the spatial-temporal information of oracle characters

$$
\begin{aligned}
\mathrm{E}_{pt} &= W_{pt}\left(\Delta x, \Delta y, p\right)^{T} \\
\mathrm{E}_{ps} &= W_{ps}\mathbf{1}_{ps} \in R^{d_F} \\
\mathrm{E}_{str} &= W_{str}\mathbf{1}_{str} \in R^{d_E} \\
\mathrm{E} &= \mathrm{E}_{pt} + \mathrm{E}_{ps} + \mathrm{E}_{str}
\end{aligned}
$$

The Sketch-BERT model also use a special [CLS] token like BERT to do downstream classification tasks.

2.4 Orc-BERT Data Augmentation

In the area of few-shot oracle character recognition, the current state-of-the-arts model is to train CNN on the few-shot training data augmented by Orc-Bert Augmenter [2].

The Orc-Bert Augmenter modifies Sketch-BERT as its backbone, learning to reconstruct the sequence data from a Gestalt task. The augmenter is trained on large-scale unlabeled source Chinese characters to recover to masked part of the sequence data with a varying mask probability.

Then it is used to recover the manually masked training sequence data of the characters in Oracle-FS to produce a more versatile combination of oracle character images as the way of data augmentation.

Point-wise augmentation(PA), namely adding Gaussian noise to each point in the sequence data, is also performed on the sequence data recovered by Orc-Bert Augmenter to add more vary to the training data.

3 Preliminary Models

First, we did some experiments using traditional machine learning or deep learning models to see how far we can go without the few-shot strategies.

3.1 SVM

Since SVM has the adavtage that it is very efficient in high dimensional space even when the dimensions of the data are larger than the number of training samples, we can easily stretch oracle character images to vectors and use C-SVM as our classifier. The objective function is:

$$
\begin{aligned}
&\min_{w,b,\xi} \frac{\|w\|^2}{2} + C\sum_{i=1}^{n}\xi_i \\
&\text{s.t. } y_i\left(w^T x_i + b\right) \geq 1 - \xi_i,\ \xi \geq 0
\end{aligned}
$$

We use radial basis function: $\kappa(x_i, x_j) = \exp(-\frac{||x_i - x_j||^2}{2\sigma^2})$ as our kernel function.

3.2 CNN on non-augmented data

According to the experiment result of Han et al. [2], DenseNet161 outperforms other selected CNN models on the non-augmented training data. So we also reproduce this to see the limitation of no data augmentation. We expect to see it beats SVM by a large margin.

3.3 Preliminary Experiments Results

As what we expected, the CNN performs better than SVM, and the results of DenseNet161 are mainly in line with those mentioned in [2], while we highly doubt its 5-shot result is a typo of 63.9.

This part of preliminary experiments shows the lower bound of models, so a good few-shot learning model on this task should perform no worse than the DenseNet161 shown here.

models	Accuracy(%)		
	1 shot	3 shot	5 shot
SVM	19.20	36.00	45.33
DenseNet161	22.60	49.38	63.80

Table 1: Results of SVM and CNN on non-augmented training data

4 Few-shot Models

As said before, we can hardly change the model hypothesis – CNN, BERT, etc. – to adapt models to fit the few-shot scenario, so we can only focus on data augmentation and self-supervised learning for this oracle character recognition task.

4.1 Fine-tuned Sketch-BERT

The most natural self-supervised learning model among all what mentioned before is the BERT-based model. Since large unlabeled sequence data are provided, we can pre-train a Sketch-BERT model on the unlabeled font character sequence data, and then use the few-shot sequence data of oracle characters to fine-tune the model by exploiting the [CLS] to directly do the classification task, in that it can grab the global information in the whole sequence, but not as what Orc-Bert does: merely use the powerful BERT model to do augmentation.

This fashion resembles how classification tasks are done in modern natural language processing: like using the pre-trained BERT model to fine-tune a sentiment classifier by exploiting the [CLS] token.

We use the Sketch-BERT model pre-trained both on unlabeled font character sequence data and quickdraw sketch dataset, and add an two-layer MLP after the [CLS] output to do classification in fine-tuning on the Oracle-FS dataset, as the *Class Label* module shown in Figure 3.

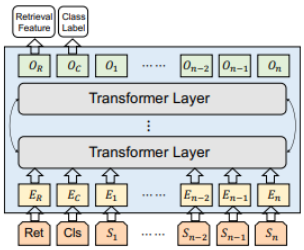

Figure 3: Structure of Sketch-BERT

However, the fine-tuned Sketch-BERT cannot even beat the SVM model in doing such classification directly, as shown in Table 2. This outcome mainly owes to the poor quality of the sequence data

generated from image data. In Figure 2, we can see that even for the same character, the extracted stroke varies greatly with none of them representing the true stroke order(especially line 4). Therefore, only if a faithful, accurate and stable stroke order generator is obtained to convert the image data to sequence data can the BERT-based models really be tapped to its full potential.

4.2 Stroke Augmenter: a simple yet powerful data augmentation method

Since directly using the poor quality sequence data has been proved invalid, now we can only focus on the image data.

Orc-Bert Augmenter uses a tremendously intricate way to add changes to the original oracle character. Nonetheless it does not change the character structure but slightly adjust strokes, as Figure 4 shows.

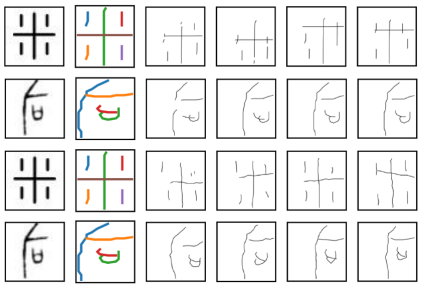

Figure 4: Examples of Orc-Bert augmented data. Line 1-2: Orc-Bert. Line 3-4: Orc-Bert+PA

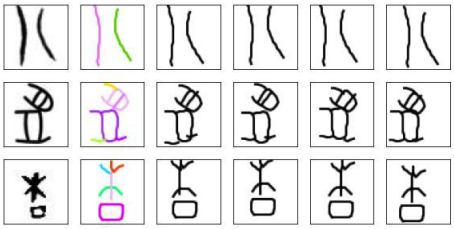

Figure 5: Examples of stroke augmented data.

We can also use another way to modulate strokes to fulfil data augmentation with nearly the same effect. Inspired by Chinese calligraphy that the position of strokes will affect the appearance but not structure of a character, we directly move strokes to generate augmented training samples. Because the sequence data only contains relative shift, just simply moving the first point in a stroke can change the whole stroke's position. From Figure 5, we can see that our augmented data are very similar to the Orc-Bert augmented ones in form. So we expect this method should be no less powerful than Orc-Bert but extremely simple and explainable. According to Occam's Razor, this Stroke Augmenter should be way better than the previous SOTA: Orc-Bert Augmenter.

Algorithm 1 Framework of Stroke Augmenter

Input: The sequence data $O_i = (\Delta x, \Delta y, p)$ of an oracle character I
Output: A pixel image I'

Deep copy O_i as O'_i
for every first point O'_j in strokes **do**
    Generate random numbers $z_1,\ z_2$ from Gaussian distribution $\mathrm{N}(\mu, \sigma^2)$
    $\Delta x_j = \Delta x_j + z_1,\ \Delta y_j = \Delta y_j + z_2$
end for
Convert O'_i back to pixel image I' and save it
return I'

4.3 Self-supervised Learning and Model Ensemble

Given that we have already proposed a more powerful way of constructing data augmentation, further improvement can only be brought from self-supervised learning. Bendou et al.[5] shows that by assembling individual components of data augmentation, self-supervised learning and model ensemble, CNN models can achieve nearly state-of-the-art effects on a myriad of few-shot learning tasks.

We randomly rotate the image by a multiple of 90 degrees. In the training time, a separate linear layer after the extracted feature is used to classify how much the image is rotated. This self-supervised loss is added to the image classification loss. By combining this part into the few-shot learning, the model indeed get a better representation of the character images.

Model ensemble can further enhance the model performance in the few-shot learning scenario by combining representations learnt from various models with great variance in hidden space construction, thus incorporate the knowledge from all these models. In [5], the feature vectors from multiple feature extractors are concatenated, centered and normalized. Then they are compared with the feature vectors of training data with the same operation to select the nearest neighbor as its class.

5 Experiments and Results

5.1 Implementation Details

For SVM model, the penalty coefficients corresponding to 1-shot, 3-shot, and 5-shot tasks are 1, 6, and 8 respectively.

For Stroke Augmenter, random noise $\sim N(0, 1.5^2)$ is added to x and y coordinate of the beginning of each stroke to move it as a whole. We trained both ResNet18 and DenseNet161 on the augmented dataset. We use a batch size of 128 and adopt Adam as the optimizer with a initial learning rate of 0.001.

For self-supervised learning, we use ResNet18 as the backbone network structure due to its commensurate performance and relative small amount of parameters for further ensemble. In the ensemble scene, three ResNet18 trained with different random seeds are combined to obtain the final results.

The following Table 2 shows the results of all the experiment we have done and the comparing performance of Orc-Bert Augmenter.[1]

		1 shot	3 shot	5 shot
Preliminary Results	Sketch-Bert	15.76	26.65	33.68
	SVM	19.20	36.00	45.33
	DenseNet161	**22.60**	**49.38**	**63.80**
Orc-Bert Augmenter	No PA	26.4	56.4	66.6
	With PA	**28.2**	**58.3**	**69.0**
Stroke Augmenter	ResNet18	28.48	58.25	**70.75**
	DenseNet161	**32.58**	**66.62**	70.30
our augmentation + self-supervised learning	one model	40.55	67.03	76.40
	ensemble	**44.68**	**71.05**	**85.47**

Table 2: All experiment results.

5.2 Result Analysis

As said in section 4, the fine-tuned Sketch-Bert cannot even beat SVM due to the poor quality of sequence data shown in Figure 2.

It can be seen in Table 2 that our simple and powerful Stoke Augmenter beats the state-of-the-art Orc-Bert Augmenter with DenseNet161 as backbone both with and without point-wise augmentation. It is worth noting that our 3-shot result(66.62% of accuracy) even overshadows the 5-shot result of Orc-Bert Augmentor(66.6% accuracy).

By using self-supervised learning and model ensemble on the augmented data, we indeed see the improvement on accuracy as expected. The best 1-shot result is even comparable to the 5-shot result from the SVM model.

6 Conclusions

In this report, we tested how traditional machine learning methods and convolutional neural networks performs on the few-shot learning oracle recognition task. Then we proposed a simple but powerful

[1]This result is quoted from Han et, al. [2], using the DenseNet161 ones.

data augmentation method named Stroke Augmenter for oracle character recognition, and achieved new state-of-the-art. It can be viewed as a strong baseline for this task. By Occam's Razor, this is much better than the previous SOTA: Orc-Bert Augmenter.

We also used self-supervised learning method to further improve the performance. It works terrible on the sequence data using BERT-based model for the lack of high quality sequence data. But self-supervised learning does enhance performance on image data together with model ensemble and brings the accuracy to a new level.

Future studies on this task should be focused on how to improve the quality of extracted sequence data from the image data. Not only can it better the performance of models purely using sequence data like fine-tuned Sketch-Bert, but it can also help our data augmentation method to produce new data with higher quality.

7 References

[1] Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. 2020. Generalizing from a Few Examples: A Survey on Few-shot Learning. ACM Comput. Surv. 53, 3, Article 63 (May 2021), 34 pages. https://doi.org/10.1145/3386252

[2] Han, W., Ren, X., Lin, H., Fu, Y., Xue, X. (2021). Self-supervised Learning of Orc-Bert Augmentor for Recognizing Few-Shot Oracle Characters. In: Ishikawa, H., Liu, CL., Pajdla, T., Shi, J. (eds) Computer Vision – ACCV 2020. ACCV 2020. Lecture Notes in Computer Science(), vol 12627. Springer, Cham. https://doi.org/10.1007/978-3-030-69544-6_39

[3] Mayr, M., Stumpf, M., Nicolaou, A., Seuret, M., Maier, A., Christlein, V. (2020). Spatio-Temporal Handwriting Imitation. In: Bartoli, A., Fusiello, A. (eds) Computer Vision – ECCV 2020 Workshops. ECCV 2020. Lecture Notes in Computer Science(), vol 12539. Springer, Cham. https://doi.org/10.1007/978-3-030-68238-5_38

[4] H. Lin, Y. Fu, X. Xue and Y. -G. Jiang, "Sketch-BERT: Learning Sketch Bidirectional Encoder Representation From Transformers by Self-Supervised Learning of Sketch Gestalt," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 6757-6766, doi: 10.1109/CVPR42600.2020.00679.

[5] Yassir Bendou, Yuqing Hu, Raphael Lafargue, Giulia Lioi, Bastien Pasdeloup, Stéphane Pateux, Vincent Gripon (2022) EASY: Ensemble Augmented-Shot Y-shaped Learning: State-Of-The-Art Few-Shot Classification with Simple Ingredients, arXiv:2201.09699

[6] Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.