OCR for Chinese Characters with CNN and EfficientNet

Team Number: 26 **Authur:** Yubo Cai Junyuan Wang

Link of the GitHub Repository

LINK: [https://github.com/yubocai-poly/Chinese-character-OCR-with-CNN-and-EfficientNet]

1. Table of Working Content

Working Content	
Yubo Cai	Junyuan Wang
- Constructing the datasets - Constructing the simple and complex CNN model - Constructing the testing function - Visualization of the result - README.md, Prepare the slides and report - Debug and Code Organization - Model Result Anlysis	- Constructing the datasets - Constructing the EfficientNetB0 model - Constructing the training function - Implementing training callbacks - Training the model - Analyzing the models

2. Highelists

1. **Training for large image databases**. This was the most difficult point in our project, how to encode the **gnt** file into a **tfrecord** for training, and decoding the **tfrecord** file.
2. Implementation of the simple and complex **CNN** model and **EfficientNet** [2] with **tenserflow** and **Keras**.
3. **Analysis of model results**, like why complex CNN models do not converge and why EfficientNet has the best results.
4. **Implementation of checkpoint saving function and logging callback functions to track the training of the model.** This is important for our project since training the EfficientNet and CNN requires high computation and time.
5. **Image processing and transformation**. Since the size of the images in our dataset varies, we cut and grayscale the images to meet the training requirements of the CNN, and since EfficientNetB0 requires different formats for the input images, there is the problem of image transformation involved.

3. Overview of the project

In this project, we have implemented OCR of Chinese characters in handwritten calligraphy based on **CNN** and **EfficientNet**. We constructed three models, a **structurally simple CNN model** and a **complex one**, as well as the **EfficientNet B0** model. We receive a result of 96% accuracy in **training** dataset and 87% accuracy in **testing** dataset which is a great result since handwritten Chinese has a complex structure, numerous styles, and a large number of characters.

4. The challenges of this project

Compared with the traditional MNIST English OCR, the Chinese Characters OCR is much more difficult for the following reasons:

1. There are more than 10k Chinese Characters which is much more than the 26 letters in MNIST. Therefore, it's quite difficult to label the data and build a complex model to accurately recognize all the characters.

2. Chinese characters are more complex in shape. Compared to the MNIST dataset, Chinese characters have a higher number of strokes, and more complex shapes and structures.
3. There are numerous variations and styles of Chinese Handwriting.

5. Dataset Preparation

We used the dataset from [CASIA's open source Chinese handwriting dataset]. This dataset contains 7185 Chinese characters and 171 English letters, numbers, punctuation, etc. The format of the dataset is **gnt** with 32 characters and labels which is not convenient for us for training, we wrote a Python program to convert **gnt** files into **tfrecord** format for training.

We first decode the **gnt** file as the index of **width, height, image, label** into **tfrecord**. The complexity of this part is that the characters in the label are Chinese characters and we have to decode the Chinese characters into **UTF-8** to convert the label into a Python readable form corresponding to our characters.txt file. Also, images, widths, etc. are for different data types, so we need to decode them separately.

6. Model Construction and Result

In this project, we mainly used the **tf.Keras API** to build the model, here is the result of our 3 models:

1. **Simple CNN**: Accuracy on training dataset: around 95%. Accuracy on the testing dataset: around 46%.
2. **Complex CNN**: Not even converge.
3. **EfficientNet B0**: Accuracy on training dataset: around 99%. Accuracy on the testing dataset: around 88%.

Detailed analysis of the result please see the comments on REAMDE.md.

7. Further Working Direction

In this project we are only recognizing individual Chinese characters in OCR, we can extend this model to recognize whole paragraphs of text. We have two ideas to expand our project in this direction:

1. We first perform the segmentation of the paragraphs into each character then use our model for OCR. Sentences in Chinese are composed of individual characters rather than words, and the relative similarity in character sizes, coupled with the infrequent occurrence of connected strokes between characters, makes segmentation easier compared to English. Therefore, segmentation may perform well in Chinese paragraphs.
2. If the direct approach does not yield satisfactory results, an alternative method to address this issue is through the use of Connectionist Temporal Classification (CTC) models [1], employing techniques such as Long Short-Term Memory (LSTM) [3] or Convolutional Recurrent Neural Network (CRNN).

References

[1] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[2] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

[3] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.