Definition

project overview :

Handwriting recognition , text is analyzed after being written. The information that can be analyzed is the binary output of a character against a background.
The approach to solving this would be to extract language dependent features using TF.

OCR ,This software used a more developed use of the matrix method (pattern matching). Essentially,this would compare bitmaps of the template character with the bitmaps of the read character and would compare the mto determine which character it most closely matched with .The downside was this software was sensitive to variations in sizing and the distinctions between each individuals way of writing.

The IAM Handwriting database is the biggest database of English handwriting images.

Problem Statement:

I will use the deep learning based approach to identifying the features of image of characters . I will use mainly CNN .
The code will take an image as an input and detect the word as an output .

1) Download the dataset and preprocessing it .
2) Train the model to read the handwriting

Metrics :

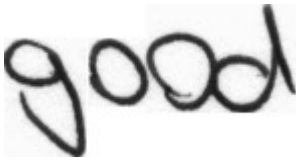
The accuracy score :

Accuracy = number of words ok / numbers of total words

Analysis

Data Exploration and Exploratory Visualization:

The IAM Handwriting Database contains forms of handwritten English text which can be used to train and test handwritten text recognizers and to perform writer identification and verification experiments.



The database was first published in at the ICDAR 1999. Using this database an HMM based recognition system for handwritten sentences was developed and published in at the ICPR 2000.

The database contains forms of unconstrained handwritten text, which were scanned at a resolution of 300dpi and saved as PNG images with 256 gray levels.

It has 1539 pages of scanned text written by 600+ writers. contributing to 5500+ sentences and 11500+ words.
the text is recognized on character-level, therefore words or texts not contained in the training data can be recognized too (as long as the individual characters get correctly classified).

This dataset contains images of each handwritten sentence with the dash-separated filename format. The first field represents the test code, second the writer id, third passage id, and fourth the sentence id.

Algorithms and Techniques

The classifier is a CNN , which is the popular algorithm for most

image processing tasks, including classification. It needs a large amount of training data compared to other approaches; fortunately, the COCO and COCO-Text datasets are big enough. The algorithm outputs an assigned probability for each class; this can be used to reduce the number of false positives

we will create a simple CNN for IAM , including Convolutional layers, Pooling layers and Dropout layers.

1. Zero Padding layer 1*1
2. lambda layer to resize the image

3. Convolutional layer with 32 feature maps of size 5×5.
4. Relu activation layer .
5. Pooling layer taking the max over 2*2 patches.

6. Convolutional layer with 64 feature maps of size 3×3.
7. Relu activation layer .
8. Pooling layer taking the max over 2*2 patches.

9. Convolutional layer with 128 feature maps of size 3×3.
10. Relu activation layer .
11. Pooling layer taking the max over 2*2 patches.

12. Flatten layer.
13. Dropout layer with a probability of 50%.

14. Fully connected layer with 512 neurons , relu activation and Dropout layer with 50%. .
15. Fully connected layer with 256 neurons , rectifier activation and Dropout layer with 50%. .
16. Output layer as softmax within 50 classes.

* Training parameters

1- number of epochs = 6

2- Batch size = 16 (how many images to look at once during a single training step)

I used the GPU of the udacity platform in the deep learning project.

Because my cpu can’t do this train , unfortunately my GPU hours are finished .

Benchmark

On the IAM data set

benchmark(1)

Figure 6. Word Level and Character Level Classification Results

Architecture	Training Accuracy	Validation Accuracy	Test Accuracy
VGG-19	28%	22%	20%
RESNET-18	31%	23%	22%
RESNET-34	35%	27%	25%
Char-Level	38%	33%	31%

benchmark(2)

a machine learning model for recognizing handwritten characters on form document.

by evaluated some SVM and found that the linear SVM using L1 loss function and L2 regularization giving the best performance both of the accuracy rate and the computation time. Based on the experiment results using data from NIST SD 19 2 nd edition both for training and testing.

the original CNN achieves an accuracy rate of 98.30% on numeral characters, 92.33% on uppercase characters, 83.54% on lowercase characters, and 88.32% on the merger of numeral and uppercase characters.The learning model was used to construct a handwriting recognition system to recognize a more challenging data on form document automatically.

Methodology

Data Preprocessing

. The images are divided into a training set and a validation set

. Resize so height = 113 while keeping aspect ratio

. Generate crops of size 113x113 from this resized image and keep random 10% of crops

. The list of images is randomized

. reshape X_train for feeding it

. convert to float and normalize

*There are also some pre processing steps before training:

. resize image to 64x64

.The images are converted to gray scale

. The pixel values get divided by the standard deviation of the pixel values

Implementation

The classifier training stage

During the classifier was trained on the pre processed training data. This was done in a Jupyter notebook (using my deep learning section work station)

1. Load both the training and validation images , pre processing them as described

a. generate_data(samples, target_files, batch_size=batch_size, factor = 0.1) : train generator shared in the class and add image augmentations

b. resize_image(image):

2. Define the network architecture and training parameters

3. Train the network, logging the validation/training loss and the validation accuracy

4. Define the loss function, accuracy

Refinement

This was improved upon by using the following techniques:

74% of the words from the IAM dataset are correctly recognized by the NN when using vanilla beam search decoding.

If you need a better accuracy :

* Data augmentation: increase dataset-size by applying further (random) transformations to the input images. At the moment, only random distortions are performed.

* Remove cursive writing style in the input images

* Increase input size

* Add more CNN layers

* Replace LSTM by 2D-LSTM.

* Replace optimizer: Adam improves the accuracy, however, the number of training epochs increases

* Decoder: use token passing or word beam search decoding to constrain the output to dictionary words.

* Text correction: if the recognized word is not contained in a dictionary, search for the most similar one.

Results

Model Evaluation and Validation

The final architecture and hyperparameters

** The shape of the filters of the convolutional layers is 5*5 at the first layer and 3*3 on others

** The first convolutional layer learns 32 filters, the second learns 64 filters and 128 filter .

** The convolutional layers have a stride of 2, so the resolution of the output matrices is half the resolution of the input matrices.

** Like the convolutional layers, the pooling layers halve the resolution too.

** The first fully connected layer has 512 outputs, the second 256.

** The training runs for 3268 iterations.

Justification

Compared by benchmark (1) , it is better than it.

compared with benchmark(2) , it is near from it.

```
6: val_loss did not improve
3268/3268 [==============================] - 3986s 1s/step - loss: 0.4394 - acc: 0.8701 - va
l_loss: 0.4835 - val_acc: 0.8577
```

Conclusion

Reflection

The process used for this project can be summarized using the following steps:

1. An initial problem and relevant, public datasets were found

2. The data was downloaded and preprocessed

3. A benchmark was created for the classifier

4. The classifier was trained using the IAM data

5. Feeding the extracted text to the system

the difficult step is the traing step , as I had to wait my laptop about one day for one epoch , I try to use the work station on the deep learning stage..

Improvement

* Adam improves the accuracy

* Better support for languages other than English

* Remove cursive writing style in the input images

* Increase input size

* Add more CNN layers

* Replace LSTM by 2D-LSTM.

* Replace optimizer: Adam improves the accuracy, however, the number of training epochs increases

* Decoder: use token passing or word beam search decoding to constrain the output to dictionary words.

* Text correction: if the recognized word is not contained in a dictionary, search for the most similar one.

References

- Handwriting Recognition on Form Document Using Convolutional Neural Network and Support Vector Machines (CNN-SVM)
- ww.researchgate.net

- Build a Handwritten Text Recognition System using TensorFlow (https://towardsdatascience.com/2326a3487cd5)
- Scheidl - Handwritten Text Recognition in Historical Documents (https://repositum.tuwien.ac.at/obvutwhs/download/pdf/2874742)
- Shi - An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text_Recognition (https://arxiv.org/pdf/1507.05717.pdf)
- Scheidl - Word Beam Search: A Connectionist Temporal Classification Decoding Algorithm (https://repositum.tuwien.ac.at/obvutwoa/download/pdf/2774578)
- Marti - The IAM-database: an English sentence database for offline handwriting recognition (http://www.fki.inf.unibe.ch/databases/iam-handwriting-database)