Online Cursive Handwriting Mongolia Words Recognition with Recurrent Neural Networks

Wu Wei, Gao Guanglai

Computer Science Department, Inner Mongolia University, Huhhot, China
cswuwei@imu.edu.cn
doi: 10.4156/ijipm.vol2.issue3.3

Abstract

This paper primarily discussed Online Handwriting Recognition methods for Mongolia words which being often used among the Mongolia people in the North China. Because of the characteristic of the whole body of the Mongolia words, namely connectivity between the characters, thereby the segmentation of Mongolia words is very difficult. We introduced a recurrent neural network to online handwriting Mongolia words recognition. The system consists of an advanced recurrent neural network with an output layer designed for sequence labelings, partially combined with a probabilistic language model. Experimental results show that unconstrained Mongolia words achieve recognition rates about 80%, compared with about 70% using a previous developed HMM-based recognition system.

Keywords: Recurrent Neural Networks, Mongolia Words Recognition, Feature Extraction

1. Introduction

During recent years, the task of online handwriting recognition has gained an immense importance in every day applications. Mongolia words are often used among the Mongolian in the North China and other regions. So it is very important to research Mongolia words handwriting recognition technique. Mongolia word is a kind of spelling characters, which has a very special written structure different from Chinese and English characters. It is written from left to right, from top to bottom, all letters are connected together to form a vertical backbone, which makes the segmentation of Mongolia letter very difficult, and every letter may have different shapes in different positions. All these characteristics bring many difficulties to recognition.

The problem is especially acute for unconstrained handwriting, where the writing style may be cursive, printed or a mix of the two, and the degree of interdependency is therefore difficult to determine in advance. Delayed strokes require special treatment because they split up the characters and therefore interfere with localization. HMMs [6] and hybrid systems incorporating time-delay neural networks and HMMs [7] are commonly trained with such features. The standard solution is to preprocess the data into a set of localized features. And require to be presegmented into characters. Using hand crafted features often yields superior results, and in some cases can render classification essentially trivial. However, there are many reasons to consider initial data. Firstly, designing an effective preprocessor requires considerable time and expertise. Secondly, hand crafted features tend to be more task specific. For example, features designed for English handwriting could not be applied to languages with substantially different alphabets, such as Arabic or Chinese. In contrast, a system trained directly on pen movements could be applied to any alphabet. Thirdly, using initial data allows feature extraction to be built into the classifier, and the whole system to be trained together.

In this paper, we apply a recurrent neural network (RNN) to online handwriting Mongolia words recognition. The RNN architecture is bidirectional Long Short-Term Memory [3], chosen for its ability to process data with long time dependencies. The RNN uses the recently introduced connectionist temporal classification output layer [2], which was specifically designed for labelings unsegmented sequence data. An algorithm is introduced for applying grammatical constraints to the network outputs, thereby providing word level transcriptions.

In our system, we make use of the local features and spatial features for feature selection. Local features come from online information, and these features provide information on the dynamics of writing at a very high resolution. While these features are good for representing very local structures in...
the handwritten strokes, they do not necessarily represent the global structural properties well. So we add the spatial features from offline information. Then all these features are used for RNN classifier. Section 2 is our system framework.

2. System overview

Our system makes use of traditional process of pattern recognition. Figure 1 presents an overview of the recognition system. The flow of data during training is show by the dash line arrows, while the data flow during recognition is shown by the solid line arrows.

Now we will start with a description of our preprocessing, and feature extraction. In section 3, we shall discuss the technology of recurrent neural network classifier, and last is our experimental results and conclusions.

2.1. Preprocessing

Preprocessing attempts to eliminate some variability related to the writing process that is not very significant from the point of view of recognition, such as variability due to the writing environment, writing style, acquisition, digitizing of images, etc.

For example, \( p_i = p_{i+1} = \ldots = p_{i+d} \) for some \( i \) and \( d \). When computing differential features, these co-occurrences can cause disturbing singularities. Thus, we remove the extra occurrences \( p_i, p_{i+1} \ldots p_{i+d} \).

After removal of repeated samples, the input signals must be filtered in order to reduce noises. In our approach, we applied a Gaussian filter [1] independently to each of the x and y coordinates of the point sequence:

\[
X_{t \text{filtered}}^i = \sum_{i=-3}^{3} W_i X_{t+i}^\text{orig} 
\]

\[ W_i = \frac{e^{-\frac{i^2}{2\sigma^2}}}{\sum_{j=-3}^{3} e^{-\frac{j^2}{2\sigma^2}}} \]

Where

Slant correction is very important for writer-independent Mongolia words recognition. Because Mongolia word has a vertical baseline, we make use of Hough transform method.
Then the sequence is size normalized to 16×16 lattice for candidate Mongolia character recognition. Size normalization is applied for writing variation, and is convenient to feature extraction. So, the process of preprocessing is:

1. Removal of repeated samples;
2. Smoothing the input sequences (Filtering);
3. Slant detection/correction;
4. Size normalization (16×16 lattice) for character recognition.

After preprocessing, the input sequence was put into feature extraction module, and then for classifying.

2.2. Feature extraction

We make full use of online and offline information for accurate recognition of different characters. The set of extracted features can be divided into two classes. The first class consists of features extracted for each point \( p_i \) considering the neighbors of \( p_i \) with respect to time. The second class takes the off-line matrix representation of the handwriting into account.

We have studied several local features, inspired by recent publications, and observed the best character recognition rates—in combination with our classification—using the sequence \( \{X_i, Y_i, \theta_i\} \) as the first class features. \( X_i \) and \( Y_i \) is coordinate points, \( \theta_i \) is tangent slope angle at point \( i \). Where \( \theta_i = Di \).

\[
D_i = \arg((-1)^i x_{i+1} - x_{i-1}) + J \cdot (y_{i+1} - y_{i-1})
\]  

With \( J=-1 \) and “arg” the phase of the complex number above.

The features of the second class are all computed using a two-dimensional matrix representing the off-line version of the data. The following features are used:

- **Ascenders/descenders**: the number of points above/below the corpus in the \( x \)-vicinity of \( p_i \).
- **Context map**: the two-dimensional vicinity of \( p_i \) is transformed to a 3 × 3 map. The resulting nine values are taken as features.

3. Method

3.1. Bidirectional Recurrent Neural Networks

Recurrent neural networks (RNNs) provide a very elegant way of dealing with (time) sequential data that embodies correlations between data points that are close in the sequence. Figure 3 shows a basic RNN architecture with a delay line and unfolded in time for two time steps. In this structure, the input vectors are fed one at a time into the RNN. Instead of using a fixed number of input vectors as done in the MLP and TDNN structures, this architecture can make use of all the available input information up to the current time frame to classification.
While delaying the output by some frames has been used successfully to improve results in a practical system [8], the optimal delay is task dependent and has to be found by the “trial and error” error method on a validation test set.

For standard RNN architectures, the range of context that can in practice be accessed is limited. The problem is that the influence of a given input on the hidden layer, and therefore on the network output, either decays or blows up exponentially as it cycles around the recurrent connections [4].

Long Short-Term Memory (LSTM; [5]) is an RNN architecture designed to address the vanishing gradient problem. For many tasks it is useful to have access to future as well past context. Bidirectional RNNs [3] [6] achieve this by presenting the input data forwards and backwards to two separate hidden layers, both of which are connected to the same output layer. Bidirectional recurrent neural network (BRNN) can be trained using all available input information in the past and future of a specific time frame for overcoming the limitations of a regular RNN. The idea is to split the state neurons of a regular RNN in a part that is responsible for the positive time direction (forward states) and a part for the negative time direction (backward states). Inputs of backward states are not connected to outputs from forward states, and vice versa. Its structure can be seen in Figure 4. It is not possible to display the BRNN structure in a figure similar to Figure 3 with the delay line since the delay would have to be positive and negative in time. And we knew that without the backward states, this structure simplifies to a regular unidirectional forward RNN, as shown in Figure 3. With both time directions taken care of in the same network, input information in the past and the future of the currently evaluated time frame can directly be used to minimize the objective function without the need for delays to include future information, as for the regular unidirectional RNN discussed above.

Because recurrent neural networks (RNNs) require pre-segmented training data, and post-processing
to transform their outputs into label sequences, we represent the output to label sequence directly, which allow a recurrent neural network to be trained using unsegmented sequences directly. So we used Connectionist temporal classification (CTC) for RNN, as shown following.

3.2. Connectionist temporal classification

This section describes the output representation that allows a recurrent neural network to be used for CTC. The network can then be used a classifier by selecting the most probable labelings for a given input sequence.

Connectionist temporal classification (CTC) [2] is an objective function designed for sequence labelings with RNNs. Unlike traditional objective functions it does not require pre-segmented training data, or post-processing to transform the network outputs into labelings. Instead, it trains the network to map directly from input sequences to the conditional probabilities of the possible labelings.

A CTC output layer contains one more unit than there are elements in the alphabet L of labels for the task. The output activations are normalized with the softmax activation function [7]. At each time step, the first |L| outputs are used to estimate the probabilities of observing the corresponding labels. The extra output estimates the probability of observing a ‘blank’, or no label. The combined output sequence estimates the joint probability of all possible alignments of the input sequence with all possible labelings. The probability of a particular labelings can then be estimated by summing over the probabilities of all the alignments that correspond to it.

So far we have briefly described an output representation that allows RNNs to be used for CTC. Then we need derive an objective function for training CTC networks with gradient descent. The objective function is derived from the principle of maximum likelihood. That is, minimizing it maximizes the log likelihoods of the target labelings. This is the same as standard neural network objective functions [9]. We require an efficient way of calculating the conditional probabilities of individual labelings. This can be solved with a dynamic programming algorithm, similar to the forward-backward algorithm for HMMs [10].

The objective function for CTC is the negative log probability of the network correctly labelings the entire training set. Let S be a training set, consisting of pairs of input and target sequences (x, z), where target sequence z is at most as long as input sequence x. Then the objective function is:

\[
O^{CTC} = - \sum_{(x,z) \in S} \ln( P(z | x) )
\] (4)

The network can be trained with gradient descent by differentiating \(O^{CTC}\) with respect to the outputs, then using back propagation (BP) through time to differentiate with respect to the network weights. Once the network is trained, we would ideally label some unknown input sequence x by choosing the most probable labelings \(l^*\):

\[
l^* = \arg \max_j p(l | x)
\] (5)

3.3. Language model

It is common practice to use a language model to weight the probabilities of particular sequences of words. We can express these constraints by altering the probabilities in (5) to be conditioned on some probabilistic grammar \(G\), as well as the input sequence \(x\):

\[
l^* = \arg \max_j p(l | x, G)
\] (6)

Absolute requirements, for example that \(l\) contains only dictionary words, can be incorporated by setting the probability of all sequences that fail to meet them to 0.

Note that, according to Bayes rules,
Online Cursive Handwriting Mongolia Words Recognition with Recurrent Neural Networks  
Wu Wei, Gao Guanglai  
International Journal of Information Processing and Management Volume 2, Number 3, July 2011

\[
p(l | x, G) = \frac{p(l | x)p(l | G)p(x)}{p(x | G)p(l)} \tag{7}
\]

, where we have used the fact that \( x \) is conditionally independent of \( G \) given \( l \). If we assume \( x \) is independent of \( G \) and all label sequences are equally probable prior to any knowledge about the input or the grammar, we can drop the \( p(l) \) term in the denominator to get

\[
l^* = \arg \max_l p(l | x)P(l | G) . \tag{8}
\]

Let \( G \) consist of a dictionary \( D \) containing \( W \) words, and a set of \( W^2 \) bigrams \( p(w | \hat{w}) \) that define the probability of making a transition from word \( \hat{w} \) to word \( w \). The probability of any labelings that does not form a sequence of dictionary words is 0.

4. Experiments

For our experiments we used our own handwriting Mongolia words database, which contains 40000 word instances from a 15000 Mongolia word dictionary.

The training set consists of 30000 Mongolia words including all different words in the database. We randomly selected 4000 Mongolia words for testing sample from remained samples.

The CTC network used the BLSTM architecture. The forward and backward hidden layers each contained 100 single cell memory blocks. The input layer was fully connected to the hidden layers, which were fully connected to themselves and the output layer. The output layer contained 97 units (96 Mongolia word characters plus the blank label). For the preprocessed data, there were 13 inputs. tanh was used for the cell activation functions and logistic sigmoid in the range \([0, 1]\) was used for the gates. For both input representations, the data was normalized so that each input had mean 0 and standard deviation 1 on the training set. Training was carried out with back propagation through time and online gradient descent (weight updates after every training example), using a learning rate of \(10^{-4}\) and a momentum of 0.9. The weights were initialized with a flat random distribution in the range \([-0.1, 0.1]\). During training, Gaussian noise was added to the inputs with a standard deviation of 0.6 to improve generalization. Network activations were reset to 0 at the start of each example. Training was stopped when performance ceased to improve on the validation set.

Our experiments are divided into two groups, one is writing Mongolia words in constrained styles (no delayed stroke and complying with stroke order). We compared CTC with HMM system, and record the word recognition rate with a language model and a dictionary. Table 1 and 2 are our experimental results.

### Table 1. Unconstrained word recognition rate

<table>
<thead>
<tr>
<th>method</th>
<th>language model</th>
<th>word recognition rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Top-1</td>
</tr>
<tr>
<td>CTC</td>
<td>no use</td>
<td>74%</td>
</tr>
<tr>
<td>CTC</td>
<td>use</td>
<td>80%</td>
</tr>
<tr>
<td>HMM</td>
<td>use</td>
<td>72%</td>
</tr>
</tbody>
</table>

### Table 2. Unconstrained word recognition rate (no delayed stroke and complying with stroke order)

<table>
<thead>
<tr>
<th>method</th>
<th>language model</th>
<th>word recognition rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Top-1</td>
</tr>
<tr>
<td>CTC</td>
<td>no use</td>
<td>82%</td>
</tr>
<tr>
<td>CTC</td>
<td>use</td>
<td>90%</td>
</tr>
<tr>
<td>HMM</td>
<td>use</td>
<td>81%</td>
</tr>
</tbody>
</table>
Table 1 shows the unconstrained results of the CTC approach compared to the HMM-based system. The recognition accuracy of 80% is a significant improvement. If we comply with Mongolia words writing stroke order, the recognition rate can achieve 90% (Table 2.).

From the experimental results, as far as we know, RNN classifier for handwriting Mongolia words recognition is feasible and outperforms any other single model.

5. Conclusion

In this paper, we primarily introduced technology of recurrent neural network based on online and offline information. Our method fits naturally into the existing framework of neural network classifiers, and is derived from the same probabilistic principles. It obviates the need for pre-segmented data, and allows the network to be trained directly for sequence labelings. Moreover, without requiring any task-specific knowledge, it has outperformed an HMM on a real-world temporal classification problem.

In the future, we will continue improve our system for higher recognition rate of writer–independent and unconstrained cursive. The technique of whole Mongolia sentence recognition is also very important and worthwhile to research.

6. References