Abstract: Recognizing lines of unconstrained handwritten text is a challenging task. The difficulty of segmenting cursive or overlapping characters, combined with the need to exploit surrounding context, has led to low recognition rates for even the best current recognizers. Most recent progress in the field has been made either through improved preprocessing or through advances in language modeling. The most systems rely on the same hidden Markov models that have been used for decades in speech and handwriting recognition. So an alternative approach is recurrent neural network, specifically designed for sequence labeling tasks where the data is hard to segment and contains long-range bidirectional interdependencies.

Recurrent neural networks (RNN) have been successfully applied for recognition of cursive handwritten documents, both in English and Arabic scripts. Ability of RNNs to model context in sequence data like speech and text makes them a suitable candidate to develop OCR systems for printed Devnagari scripts. A regular recurrent neural network (RNN) is extended to a bidirectional recurrent neural network (BRNN). The BRNN can be trained without the limitation of using input information just up to a preset future frame. This is accomplished by training it simultaneously in positive and negative time direction. Bidirectional Long Short Term Memory (BLSTM) architecture with Connectionist Temporal Classification (CTC) output layer was employed to recognize printed Devnagari text.

Keywords: Handwriting recognition, online handwriting, offline handwriting, connectionist temporal classification, bidirectional long short-term memory, recurrent neural networks, hidden Markov model.
1. Introduction:

Handwriting recognition is traditionally divided into online and offline recognition. In online recognition, a time series of coordinates, representing the movement of the pen tip, is captured, while in the offline case, only an image of the text is available. Because of the greater ease of extracting relevant features, online recognition generally yields better results. Another important distinction is between recognizing isolated characters or words and recognizing whole lines of text. Unsurprisingly, the latter is substantially harder, and the excellent results that have been obtained for digit and character recognition have never been matched for complete lines. Last, handwriting recognition can be split into cases where the writing style is constrained in some way for example, only hand-printed characters are allowed and the more challenging scenario where it is unconstrained.

In this paper, comparative study of different handwriting recognition methods are discussed in section II. Application of handwriting recognition areas and comparison table is discussed in section III. The paper and the work is concluded and future work is in section IV.

2. Comparative Study Of Different Handwriting Recognition Methods:

Here to compare two of the most popular approaches for handwriting recognition: namely Hidden Markov Models (HMMs) and BLSTMs (Bidirectional Long-Short Memory). Both approaches are efficient for text-line recognition since they are segmentation-free. A sliding window is shift from left to right in order to extract feature sequences, thus avoiding to segment the text-line into words and characters. So use an efficient set of features has proposed for Latin and Arabic scripts. However window parameters must be adequately set according to handwriting image resolution. In HMMs, frame dependency with neighboring ones is taken into account by derivative features (1st order regressions). In contrast, frame dependency with long-distant ones on its left and right, is taken into account in the so-called bi-directional architecture of BLSTMs, by the forward and backward recurrent connections. Since frame dependency is implemented in very different ways for HMMs and BLSTMs, it is of interest to compare the feature extraction parameter setting obtained for each approach. Larger values of window step parameter lead to shorter frame sequences, thus shorter training and decoding times. Thus, correctly adjusting such parameter is also a practical issue for text-line recognition.
HMMs are based on the stochastic modeling of observed frames while BLSTM are based on recurrent neural networks. Both take as input frame sequences extracted from a sliding window.

2.1. HMM Model:

In HMM modeling, characters are represented as a succession of states with left-right transitions and a self-transition. In each state, observations follow a Gaussian mixture probability density function of \( N_G \) Gaussian components.

Such approach leads to an increase number of parameters, compared to a context-independent HMM. The state-tying is based on decision-tree clustering. These decision trees are based on questions concerning the right and left context of a central character. A decision tree is built for each central letter and node splitting is decided according to two parameters: cluster minimal occupancy and likelihood maximization. This approach presents the advantage of allowing trigraph models, unseen during training, to be introduced at decoding time. Cluster minimal occupancy and likelihood threshold have both been optimized on a distinct validation database. This approach led to improved recognition performance as highlighted and this system participated in Arabic and Latin handwritten word recognition competitions.

To model full text lines with HMMs, concatenate word models with space models in-between. Word models themselves are made of the succession of its compound character models. System parameters are learnt through iterations of the Baum-Welch algorithm. \( N_G \) is gradually incremented between re-estimations. Among parameters, the model includes some predominant ones such as \( N_G \) and the number of states per character model. Performance also depends on feature extraction parameters. Decoding is done by a Viterbi token passing algorithm through lattices. These lattices can include language model probabilities. Pruning is performed at token and word levels.

2.2. BLSTM Model:

Use a BLSTM (Bidirectional Long-Short Term Memory) with one hidden layer containing 100 blocks (each containing one cell). In BLSTM, classical neural network units are replaced with memory blocks, called LSTM (Long Short-Term Memory). Those blocks can keep information through long time intervals (more than 1000 time samples): LSTM includes a memory cell and multiplicative logical gates which are specifically designed to memorize or forget relevant information through time. Those gates can pass or block signals. BLSTMs are
bidirectional recurrent networks: They consist of the coupling of 2 recurrent neural networks. They take a mono-dimensional signal as input (sequence of frames from 1 to T) and introduce two contexts in the image (past and future). Thus, contextual information is taken into account from both handwriting directions (left-to-right and right-to-left).

So choose this RNN architecture rather than a Multidirectional RNN (MDRNN) since its input is similar to HMMs: frames extracted by a sliding window. Sharing the same inputs makes it easier to compare recognizers. Here consider a network with only one hidden layer. The hidden layer is made of two independent neural layers: the forward layer which takes the original frame sequence as input (from 1 to T) and the backward layer which takes the reversed sequence (from T to 1) as input (Fig 1). Hence, the value of an output unit at time step t is the linear combination of the outputs of the forward and backward hidden layers at this time step t. The BLSTM recognizer is trained with a gradient-based method, “Back-Propagation Through Time” (BPTT) (Werbos 1988, Williams 1995). After each training epoch, recognition error rates are evaluated on the validation set. If error rates don’t improve for twenty epochs, network training is stopped. This strategy avoids data over-fitting. Forward and backward layer are processed independently through training and decoding phases. A backward-forward algorithm, referred to as CTC (Connectionist Temporal Classification) token passing, follows that takes the posteriors as input and provides a sequence of words given a dictionary and a language model.

![Figure 1: BRNN unfolded through 3 times steps](image)

Frame sequences are extracted from previous preprocessed text-line images by a sliding window of size \( w \) and step \( \delta \), the shift between two consecutive windows (Fig. 2). Total number of parameters depends on \( w \) window size. Each frame consists of a set of features.
which have been successfully applied to HMM-based Latin and Arabic handwritten word recognition and recently to text lines recognition. These features are the following:

i. F1: foreground pixel density
ii. F2: number of foreground/background transitions between adjacent cells
iii. F3: gravity center position difference with following window
iv. F4: relative position of gravity center
v. F5: pixel density above upper baseline (ascenders zone)
vi. F6: pixel density under lower baseline (descender zone)
vii. F7: number of foreground/background transitions between cells above lower baseline (ascenders + core zone)
viii. F8: relative position of gravity center to baselines
ix. F9-20: local convexity features
x. F21-...: pixel densities for each frame column (depends on w)

In order to extract features, each window is divided into a fixed number of cells. This allows the feature extraction to cope with different images’ heights. Using the same features allows us to fairly compare the two recognizers.

![Feature extraction with sliding window](image)

*Figure 2: Feature extraction with sliding window*

### 3. Application Areas And Comparison Table:

#### 3.1. Application Areas:

i. Telecommunication

ii. control of chemical plants
iii. control of engines and generators
iv. fault monitoring, biomedical diagnostics and monitoring
v. speech recognition
vi. robotics, toys and edutainment
vii. video data analysis
viii. Man-machine interfaces.

<table>
<thead>
<tr>
<th>Hidden Markov Model (HMM)</th>
<th>Recurrent Neural Network (RNN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard HMMs are generative.</td>
<td>RNN is discriminative.</td>
</tr>
<tr>
<td>The mixtures of diagonal Gaussians used in standard HMMs provide less flexible models of the input features than the RNNs.</td>
<td>RNNs provide more flexible models of the input features than the mixtures of diagonal Gaussians used in standard HMMs.</td>
</tr>
<tr>
<td>The internal states of a standard HMM are discrete and univariate.</td>
<td>RNNs have a continuous multivariate internal state (the hidden layer) whose information capacity grows linearly with its size.</td>
</tr>
<tr>
<td>HMMs are constrained to segment the input into a sequence of states or units.</td>
<td>RNN trained with CTC does not need to segment the input sequence.</td>
</tr>
<tr>
<td>HMMs assume that the probability of each observation depends only on the current state.</td>
<td>RNNs, on the other hand, modeling continuous trajectories is natural, since their own hidden state is itself a continuous trajectory.</td>
</tr>
</tbody>
</table>

Table 1: Comparison table

4. Conclusion And Future Work:

Several years ago, people who used computers took for granted the notion that they would have to adapt to their style of input to something computer friendly whether in typing or filling out forms with letters neatly boxed. But now, computers whose sole input method is handwriting are doing well, and computers are taking on tasks once thought beyond their abilities. Handwriting recognition is, without doubt, changing the way people relate to computers. For recognizing unconstrained handwritten text an RNN is better than the HMM. With respect to recognition performance, BLSTM distinctly surpass HMM. However when it comes to computing times, HMM shows fastest training and BLSTM fastest decoding phases. The key features of the network are the LSTM architecture, which provides access to long-
range bidirectional contextual information, and the CTC output layer, which allows the
network to be trained on unsegmented sequence data. In future plan to develop a strategy for
including a statistical n-gram language model in the RNN. It is reasonable to integrate a
statistical language model, since performing handwritten text recognition on text lines and
not only on single words. It also plan to overcome the problem of Out-Of-Vocabulary words
(OOV). This could be done by using the network likelihood and the edit distance to the
nearest vocabulary word.

Acknowledgment:

I heartly thanks Dr. R. Jayadevan, HOD of Information Technology (Ph.D.), MMCOE Pune
for her keen interest and useful advice in my paper.
References:


