RECOGNITION OF ONLINE HANDWRITING WITH VARIABILITY ON SMART DEVICES

Guohua Ren, Viswanath Ganapathy

LG Silicon Valley Labs
5150 Great America Pkwy, Santa Clara, 95054
guohua.ren@lge.com, viswanath.ganapathy@lge.com

ABSTRACT

Online handwriting recognition enables a convenient user interface for smart devices. However, readability of the handwriting varies based on the device user and the writing surface. State of art handwriting recognition systems are not yet robust with respect to different variability in writing text which results from user movement, writing speed, available space to write on, sloppiness *etc*. In this work we employ hybrid deep neural network architectures based on Bidirectional Long Short Term Memory (BLSTM) and Connectionist Temporal Classification (CTC) networks to recognize user independent dynamic handwriting with variability. Evaluated on IAMOnDB database, we show that our proposed model achieves state of the art accuracy with minimal data preprocessing and recognizes text with variability. We further compress the model using knowledge distillation to deploy on resource constrained smart devices. Our novel strategy of training student model for CTC networks enjoys significant model size reduction with moderate performance degradation.

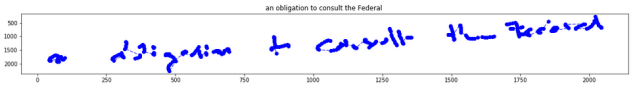
Index Terms— Handwriting recognition, LSTM, CTC, overlapped handwriting, Knowledge Distillation

1. INTRODUCTION

Online handwriting recognition systems have gained significant importance due to the increasing adoption of smart devices such as white boards and mobile phones. However, handwritten text on most of these smart devices is not always legible and has strong variability. This variability in the written text is introduced by factors such as the speed of writing, user movement, available of writing space, sloppiness and results in overlapping characters, variable spacing *etc*. The performance of several commercially available handwriting systems is limited due to such inherent variability in the written text [1], [2].

Learning and inferring using deep neural network (DNN) models on smart devices is constrained by available memory and computational resources. For example, the SpotGarbage app [3] uses convolutional neural networks to detect garbage in images but consumes 83 percent of CPU and takes more than five seconds to respond. Therefore, approaches to compress DNN models for inference on smart devices have been gaining significant attention [4], [5].

In this paper we describe a hybrid DNN based handwriting recognition system that can reliably infer handwritten text on a smart device. The hybrid model employs a bi-directional LSTM [6] and connectionist temporal classifier (CTC)[7]. The design space consisting of input features and hyper-parameters and approaches to reduce model size are investigated to enable handwriting recognition with inherent variability on smart devices.



(a) GT: 'an obligation to consult the Federal'

(b) GT: 'is a painful blow to the West'

(c) GT: 'though fighting a grumbling rearguard'

Fig. 1: Samples from IAM-OnDB Database

The main contributions of this work are: 1) development of a writer-independent handwriting recognition system which achieves state-of-the-art recognition accuracy (as measured in terms of character error rate) while requiring minimal preprocessing of input data 2) detection of illegible handwritten text with overlapped characters 3) development of a novel knowledge distillation training method for CTC networks to train low complexity student models for inference on resource constrained smart devices.

Our paper is structured as follows. In Sec. 2 the architecture of the hybrid DNN for handwritten text recognition is explained. Our novel knowledge distillation strategy for training CTC networks is detailed in Sec. 3. The database employed, model training setup as well as performance analysis are detailed in Sec. 4. Sec. 5 concludes this work and suggests possible directions for future extentions.

2. SYSTEM ARCHITECTURE

Online handwritten data on smart devices are usually captured and stored in the form of strokes. Each stroke consists of a sequence of pen-point positions. Based on the pen-point positions, various features [8] like stroke coordinates, time stamps, writing angle and curvature can be extracted and used as input to the handwriting recognition system. The desired output of such a system is the detected text representing the handwritten pen-point sequence. A few examples from IAM-OnDB database is shown in Fig. 1.

Outlined below is our hybrid handwriting recognition framework. We will start by reviewing the two major building blocks, namely LSTM and CTC networks.

2.1. Reccurrent Neural Network

Recurrent neural networks are a special kind of neural network which have drawn a lot of attention due to their capability to store and make use of previously stored information compared to other forms of NNs.

In the literature, RNNs have found a lot of applications in sequence modeling, e.g. speech recognition [9], natural language processing [10], etc. Though it is well known that RNNs suffer from the vanishing gradient problem [11], LSTMs were proposed in [6] to solve the problem. By including a 'memory cell', information can be maintained in the cell for long period of time. A set of 3 gates are introduced to control information entry into the memory cell, cell output, and duration of memory. Subsequently, peephole-LSTM was proposed in [12] to learn precise timing of spike signals by letting LSTM cell control the gates. GRU [13] was proposed as an alternative to LSTM. Fewer gates are involved in GRU and, as a result, it is computationally more efficient than LSTM. Recently, [14] put forward a Minimal RNN with greater interpretability and trainability. We will explore how these different RNN structures perform on the task of online handwriting recognition.

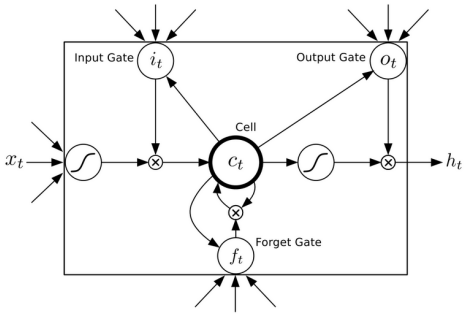


Fig. 2: LSTM memory block with one cell, containing input gate, forget gate and output gate. [1]

Considering the fact that in online handwriting data, one character is connected with another, it is natural to extend LSTM to a bi-directional form following [15], so that when recognizing the current character, its next and previous characters play equally important roles. The objective function of the LSTM training algorithm involves a label corresponding to each point of the training sequence. This requires segmented data associated with each character. Therefore, to detect the most likely character sequence from unsegmented data a modified output layer has to be employed.

2.2. Connectionist Temporal Classification

Our recognition task aims at transcribing a sequence of (w,h) coordinates into characters or words which further forms sentences. It is commonly known that LSTMs are suitable for such sequence learning tasks, though a typical LSTM training objective function requires pre-segmentation of the input sequence since a true label is needed for each point of training sequence.

However for unconstrained cursive handwriting, segmentation is not a trivial task. HMMs have been frequently employed where segmentation and recognition are done at the same time. But HMM models have their own drawbacks, [7].

Considering the success that discriminative models have achieved in image classification, in this work, we adopt the CTC framework which is completely discriminative and eliminates the need for pre-segmented data.

Assuming that the handwriting sequence contains at most K unique labels, an additional blank label indicating that no label is emitted to the current input is added and indexed as 0. Given any input handwriting sequence $X = (x_1, \ldots, x_T)$, the target sequence is denoted as $z = (z_1, \ldots, z_U)$. The length U of the target sequence z is generally less than or equal to length K of the input sequence X. By optimizing the LSTM model parameters, CTC aims at maximizing the log-likelihood of the target sequence z given input X.

The LSTM output activations are normalized by a softmax layer which has $K+1$ nodes, accounting for K true labels and the added 1 blank label. At each time step t, the output vector $\mathbf{y}_t$ consists of elements y_t^k's where each of them y_t^k is the probability of observing label k. Recall that labels z are not aligned to the input, evaluating the likelihood of z given the LSTM output is difficult. In order to make connections between the LSTM outputs and the label sequences, [7] the CTC path is introduced and serves as a kind of medium. A CTC path is a sequence of predicted labels p, in our case, at the character level. Unlike the target label z, occurrences of repeating characters and blank labels are allowed. Therefore, the total probability of a specific path can be evaluated as the product of the probabilities of individual labels:

$$Pr(\mathbf{p}|\mathbf{X}) = \sum_{t=1}^{T} y_t^{p_t} \tag{1}$$

Note that the mapping between target sequence z and the CTC path is "one to many", since CTC paths may correspond to the same target sequence. For example, paths $a, a, \_, b, \_, \_, c$ and $a, b, \_, c, c$ will be mapped to the same sequence a, b, c after first merging the repeated labels and then removing the blank labels. If we denote all the CTC paths for z as a set B, then the probabilities of all these paths will sum up to the likelihood of z.

$$Pr(z|x) = \sum_{p \in B} Pr(p|X) \tag{2}$$

In this way, the model automatically learns the alignment between the input sequence and the predicted labels without any input pre-segmentation or output post-processing.

2.3. Hybrid Network

As illustrated in Fig 3, our model fed handwriting pen-point sequences into a couple of bidirectional LSTM layers after which the CTC layer was employed for transcription. Note that the input handwriting sequence is subject to a preprocessing and feature extraction stage. We only applied size scaling for preprocessing. In terms of feature extraction, following [8] in the literature, we tried computing different kind of features. Our finding was that using features (w, h)-co-ordinates and time stamp sufficed to achieve high-level recognition accuracy and other features like writing direction and curvature did not add positive value to the recognition performance.

We explored different model structure and hyper-parameter settings. Our best performing model consisted of two BLSTM layers, with 96 and 192 hidden nodes respectively. As for the CTC decoding algorithm, beam search with beam size greater than 1 lead to very limited performance boost while heavily sacrificing training speed. As such, we employed a fixed beam size of 1 (greedy search) in our experiments.

[1] http://blog.otoro.net/2015/05/14/long-short-term-memory/

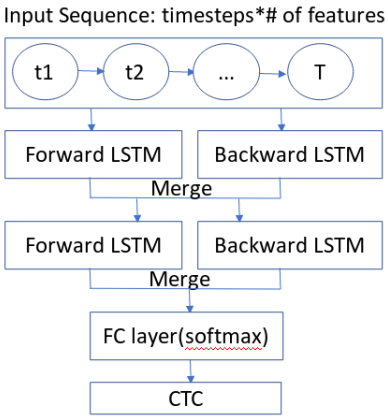


Fig. 3: Model Structure

3. MODEL COMPRESSION

Despite the huge success of DNN models on applications such as image classification, object detection, machine translation and speech recognition, their capacity is known to rely upon a huge number of parameters which make model deployment impractical on smart devices with restricted computational resources and memory storage.

To alleviate the above problem, a plethora of works have been put forward to compress the deep models which include low rank approximation [4], sparsity [16], quantization [5] and knowledge distillation [17], [18]. In this work, we explored the potential of knowledge distillation.

Knowledge distillation is a method for compressing deep models. The core idea is to have a student model mimic the behavior of a teacher model. The teacher model is trained with the original true labels while the student model is typically trained with the teacher model's predicted labels. For CNN-like networks, it has been proven that softening the teacher model's output can reveal the so-called dark knowledge which facilitates the training of a good student model [17]. Unlike CNN networks, CTC networks directly output a decoded sequence of hard labels and, as such, there is no way of softening the CTC output. Therefore, knowledge distillation for CTC networks remains an open problem [19]. One possible solution that was attempted in [18] was to force the student model's output to be close to both teacher model's CTC output and the true labels.

In the field of machine learning, it is commonly known that ensemble methods can be used for improving prediction performance, [20]. The idea of ensemble methodology is to build a predictive model by integrating multiple models. It has been proven that properly integrating multiple models' outputs outperforms selecting the single best model [21].

Inspired by the idea of ensemble methodology, here we propose to train a student model by learning from ensemble *i.e.* multiple teacher models simultaneously. Output predictions from several teacher models are acquired first and then randomly assigned as true labels while training the student model. In this way, we circumvent the hard label softening issue and introduce uncertainty into the true label in the mean time.

Training Strategy	**CER(%)**
[23]	4.3
[22]	9.26
LSTM with 2 input features w, h	5.66
GRU with 2 input features w, h	5.87
peephole-LSTM with 2 input features w, h	5.69
Minimal-RNN with 2 input features w, h	6.50
peephole-LSTM with 3 input features w, h, t	5.82
LSTM with 3 input features w, h, t	5.82

Table 1: Model Comparison

4. EXPERIMENTS

4.1. Data Description

All experiments have been conducted on the IAM-OnDB [1], a large online handwriting database. 221 writers contributed their handwriting samples to generate a total of 13049 isolated and labeled text lines, containing 11,050 distinct words, acquired from a E-Beam System.

Two benchmark tasks have been defined for the IAM-OnDB, among which we selected the IAM-OnDB-t2 benchmark task for all the experiments to compare with the state-of-art approaches. In this task the database is divided into four predefined disjoint sets: one training set containing 5,364 lines; two validation sets containing 1,438 and 1,518 lines respectively which can be used for hyper-parameter tuning; and one test set containing 3,859 lines. We consider a writer-independent recognition task here since no same writer appears in more than one set.

4.2. Setup

RMSProp optimizer was adopted to train the hybrid network, using a fixed learning rate of 0.001 and mini-batches of 32 samples.

We measured the Character Error Rate (CER%) which is calculated as the quotient of the number of changes needed to transform the model predicted output into the ground truth text and the total number of characters in the ground truth label:

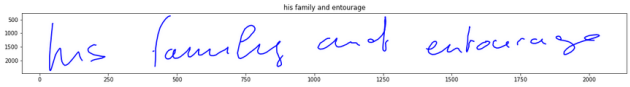
$$CER = \frac{insertions + substitutions + deletions}{\#\ of\ total\ characters} \quad (3)$$

4.3. Results

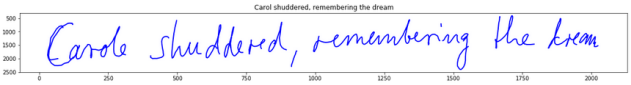
4.3.1. Model Performance

Since we use the same experimental setting as [22] and [23], we cite the results from the original works. Both [23] and our scheme outperforms [22] with a large margin, Table. 1. Even though [23] achieved the best testing recognition accuracy, it is important to note that [23] uses a different but much larger training dataset and requires extensive preprocessing, whereas our approach does not require any kind of preprocessing except for data scaling. Here we also explore how different RNN structures affect the recognition result and we find that with the same hyper-parameter setting, LSTM, GRU and peephole-LSTM achieve comparable accuracy.

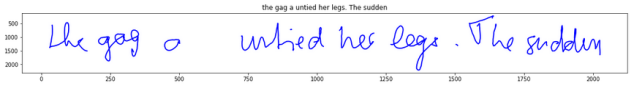
Fig. 4 depicts several testing samples from IAM-OnDB database along with the ground truth labels (GT) and our model predictions. It is worth noting that some of the faulty predictions should actually be right, such as'cCarole' in Fig. 4(b) since the character c is



(a) GT: 'his family and entourage' Predicted: 'his family and entourage'



(b) GT: 'Carol shuddered, remembering the dream' Predicted: 'cCarole shuddered, rememtbering the dream'



(c) GT: 'the gag a untied her legs. The sudden' Predicted: 'the gag a umtied her legs. The suddm'

Fig. 4: Sample predictions on our own generated overlapped handwriting. Error predictions are marked in red.

(a) GT: 'pineapple' Pred: 'pineapple'

(b) GT: 'Overlapped data' Pred: 'Overlappeddata'

Fig. 5: Sample predictions on our own generated overlapped handwriting.

overwritten. Furthermore, some of the faulty predictions could be forgiven, such as the last word 'sudden' in Fig. 4(c) which has been recognized as 'suddm'.

4.3.2. Handwriting Recognition with Variability

Recall that our goal is to achieve handwriting recognition with variability which often leads to overlapped handwriting. Our solution is that in addition to the coordinate information w, h,the time stamp t of each pen point should be included as an additional input feature to the model. Model training is implemented in the same way as in Sec. 4.3.1.

Due to a lack of datasets representing overlapped handwritten text in the IAM-OnDB database, for the purposes of validation, we generated a small dataset of our own handwriting using a similar E-beam system. We can tell from Table. 1 that adding time information as an additional feature leads to marginal degradation in performance. Importantly, our model using coordinate information as well as time stamp was able to reliably recognize overlapped handwriting Fig. 5. This is, to the best of our knowledge, the first demonstration of recognition of overlapped handwritten text involving realistic handwriting variability.

Student Model Training Strategy	**CER(%)**
Randomly pick teachers' predictions	7.51
Randomly pick teachers' predictions & true labels	7.78
Use true labels	8.15
[18] use LSTM model as teacher	9.26
Randomly pick LSTM model's predictions & true labels	7.54

Table 2: Knowledge Distillation for CTC networks

Training Strategy	**CER(%)**	Model Size (MB)
LSTM Model	5.66	8.4
Student model trained following proposed KD strategy	7.51	2.5
8-bit LSTM [5]	6.45	6.2

Table 3: Model Compression

4.3.3. Knowledge Distillation for Model compression

The results of the knowledge distillation experiments are reported in Table 2. Our LSTM, peephole-LSTM and Minimal-RNN models were used as three teachers, and we tried to train a uni-directional GRU student model. As expected, the student model trained with just true label achieved the worst performance, whereas using a random mixture of three teacher models' prediction as true label performed the best. Even though the trained sudent model did not outperform any of the teachers, it still achieved a reasonable level of accuracy.

We also compared our approach with the knowledge distillation method proposed in [18], for which our LSTM model was used as the teacher model. It is clear that using the same teacher model, our proposed knowledge distillation strategy resulted in a better student model, as shown in Table 2.

Furthermore,we explored other methods for model compression. Based on [5], we implemented the bit-constrained LSTM network and the results are shown in Table 3. 8-bit constraint for [5] was applied since 4-bit constraint leads to model divergence. Importantly, our student model trained following the proposed knowledge distillation strategy achieved a much larger model compression rate while attaining a similar level of recognition accuracy.

5. CONCLUDING REMARKS AND FUTURE WORK

With the goal of improving handwriting interface on smart devices, we studied the problem of handwriting recognition with variability in this work. Using a hybrid model of BLSTM and CTC, our model outperformed all previously reported approaches on IAM-OnDB database with minimal preprocessing. To cope with handwriting variability caused by limited writing space or user movement, we proposed a simple but effective solution of using the time stamp in addition to the coordinates of the pen positions. A novel knowledge distillation methodology was also developed to further enable model inference on smart devices with resource constraints.

Future directions involves enabling learning on smart devices so that user's handwriting style can be individually treated and optimized. In addition, special weight matrix structures could be explored to further accelerate on-device model learning and inference.

6. REFERENCES

[1] Marcus Liwicki and Horst Bunke, "Iam-ondb-an on-line english sentence database acquired from handwritten text on a whiteboard," in *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*. IEEE, 2005, pp. 956–961.

[2] Lutz Gericke, Matthias Wenzel, Raja Gumienny, Christian Willems, and Christoph Meinel, "Handwriting recognition for a digital whiteboard collaboration platform," in *Collaboration Technologies and Systems (CTS), 2012 International Conference on*. IEEE, 2012, pp. 226–233.

[3] Gaurav Mittal, Kaushal B Yagnik, Mohit Garg, and Narayanan C Krishnan, "Spotgarbage: smartphone app to detect garbage using deep learning," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 2016, pp. 940–945.

[4] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman, "Speeding up convolutional neural networks with low rank expansions," *arXiv preprint arXiv:1405.3866*, 2014.

[5] Qinyao He, He Wen, Shuchang Zhou, Yuxin Wu, Cong Yao, Xinyu Zhou, and Yuheng Zou, "Effective quantization methods for recurrent neural networks," *arXiv preprint arXiv:1611.10176*, 2016.

[6] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] 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*. ACM, 2006, pp. 369–376.

[8] Isabelle Guyon, Paul Albrecht, Yann Le Cun, J Denker, and Wayne Hubbard, "Design of a neural network character recognizer for a touch terminal," *Pattern recognition*, vol. 24, no. 2, pp. 105–119, 1991.

[9] Alex Graves and Navdeep Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International Conference on Machine Learning*, 2014, pp. 1764–1772.

[10] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze, "Comparative study of cnn and rnn for natural language processing," *arXiv preprint arXiv:1702.01923*, 2017.

[11] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al., "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001.

[12] Felix A Gers, Nicol N Schraudolph, and Jürgen Schmidhuber, "Learning precise timing with lstm recurrent networks," *Journal of machine learning research*, vol. 3, no. Aug, pp. 115–143, 2002.

[13] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[14] Minmin Chen, "Minimalrnn: Toward more interpretable and trainable recurrent neural networks," *arXiv preprint arXiv:1711.06788*, 2017.

[15] Mike Schuster and Kuldip K Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[16] Song Han, Jeff Pool, John Tran, and William Dally, "Learning both weights and connections for efficient neural network," in *Advances in neural information processing systems*, 2015, pp. 1135–1143.

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[18] Yoon Kim and Alexander M Rush, "Sequence-level knowledge distillation," *arXiv preprint arXiv:1606.07947*, 2016.

[19] Haşim Sak, Félix de Chaumont Quitry, Tara Sainath, Kanishka Rao, et al., "Acoustic modelling with cd-ctc-smbr lstm rnns," in *Automatic Speech Recognition and Understanding (ASRU), 2015 IEEE Workshop on*. IEEE, 2015, pp. 604–609.

[20] Lior Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 1–39, 2010.

[21] Saso Dzeroski and Bernard Zenko, "Is combining classifiers better than selecting the best one?," in *MACHINE LEARNING*. 2004, pp. 255–273, Morgan Kaufmann.

[22] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[23] Daniel Keysers, Thomas Deselaers, Henry A Rowley, Li-Lun Wang, and Victor Carbune, "Multi-language online handwriting recognition.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1180–1194, 2017.