Offline Handwriting Recognition Using LSTM Recurrent Neural Networks

Poster · November 2016

CITATIONS
2

READS
1,508

3 authors, including:

Matthia Sabatelli
University of Liège
20 PUBLICATIONS **52** CITATIONS
SEE PROFILE

Yaroslav Shkarupa
National University of Kyiv-Mohyla Academy
2 PUBLICATIONS **13** CITATIONS
SEE PROFILE

Offline Handwriting Recognition Using LSTM Recurrent Neural Networks

Yaroslav Shkarupa [a] Roberts Mencis[a] Matthia Sabatelli [a]

[a] *University of Groningen*
Department of Artificial Intelligence and Cognitive Engineering
Nijenborgh 4, 9747 AG Groningen
The Netherlands

Abstract

Handwriting recognition is a notoriously difficult problem in Machine Learning. Despite decades of research and development, modern handwriting recognition systems still exhibit suboptimal performance in the real world applications. Recent studies show great potential of Recurrent Neural Networks (RNN) with Long Short-Term Memory (LSTM) for unsegmented handwritten word recognition. In this paper we evaluate two approaches based on RNN with LSTM for the recognition of historic handwritten Latin texts. The first approach makes use of a Connectionist Temporal Classification output layer, and the second approach is based on Sequence-to-Sequence Learning. To test these approaches we have built a handwriting recognition system which takes an unsegmented word image as an input and provides a decoded string as an output. Both approaches show promising results with 78.10% and 72.79% word-level accuracy on the test dataset for respective methods. The Connectionist Temporal Classification approach consistently outperforms the Sequence-to-Sequence Learning approach in terms of generalization and prediction of long words.

1 Introduction

While optical character recognition (OCR) can be considered as a mature field with very high accuracy rates for printed documents, recognition of handwritten text remains a much harder challenge. The best performing teams of the ICDAR2015 competition were able to achieve just 69.8% word level accuracy, and 84.5% character level accuracy rates [13]. High variability of handwritten characters and limited accessibility of language models, especially for historical texts, are two big hurdles for existing OCR methods. While for a long time the main methods were based on carefully extracted features and probabilistic models, such as the Hidden Markov Models, recently, better results were achieved by using deep learning techniques, like the Convolution Neural Networks (CNN) and Recurrent Neural Networks (RNN) with Long Short Term Memory (LSTM).

In this paper we evaluate two RNN architectures - Connectionist Temporal Classification (CTC) and Sequence-to-Sequence Learning (Seq2Seq) for recognizing word-level labels in handwritten medieval Latin texts. We focus on balancing the performance and architectural simplicity of handwriting recognizing system so that the training of the models could be completed on average PC within reasonable time (less than 24 hours).

As a direct comparison of internal configuration (e.g. number of hidden units) of CTC and Seq2Seq models could be difficult due to considerable differences of both architectures, we evaluated different configurations for each model individually with an aim to maximize accuracy within reasonable training time. At the end the best performing configuration of each model was evaluated on the separate test dataset.

The research was done as part of the course "Handwriting Recognition" given by prof. dr. L. Schomaker at the University of Groningen in Spring 2016. The full pipeline of the recognition system is described alongside the achieved results. We used OpenCV for image preprocessing and Tensorflow framework for all machine learning tasks.

2 Dataset

The training dataset consisted of 118 scanned pages of handwritten medieval Latin texts from two sources - KNMP Chronicon Boemorum [1] (72 pages) and Stanford CCCC [3] (46 pages). The dataset was made available through the Monk System [2], a search engine for handwritten document collections developed at the University of Groningen by a research group at the Artificial Intelligence institute ALICE, under supervision by prof. dr. L. Schomaker [14]. Figures 1 and 2 show examples of handwritten text from both sources.

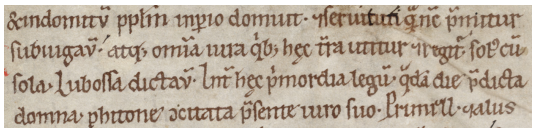

Figure 1: Example of KNMP handwritten text

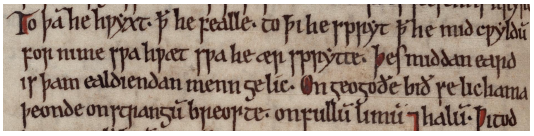

Figure 2: Example of Stanford handwritten text

Each scanned page was accompanied by XML file specifying positions of line, word and character rectangles as well as text labels for words and individual characters. In total 4988 labeled words and 14443 labeled characters were available for training, with 42.3 labeled words on average per page. Average length of a word was 2.9 characters with one or two character words making up around half of all labels. Figures 3 and 4 show some examples of segmented words with accompanying labels.

Figure 3: Examples of labeled KNMP words

Figure 4: Examples of labeled Stanford words

Additional 528 words from both sources were manually labeled to increase the number of multi-character words available for training. These labels were used only for training the Seq2Seq model as character-level labeling is not needed using this approach.

The final test dataset contained approximately the same number of pages as the dataset made available for training. The evaluation dataset was small subset of final test dataset and was made available few weeks before the final test. The test dataset itself was not available during training, but a lexicon of words with corresponding frequencies was provided.

3 Methods

3.1 Preprocessing

The preprocessing phase can be considered as the first stage of the recognition system. The main goal of this step is to modify the images in a way that will make it easier and faster for the recognizer to learn from them.

The word images were cut out of high-resolution RGB images. The average width and height of a word image was 90x60 pixels, with maximum width reaching 300-400 pixels for very long words. Thus 90x60x3=16200 features would be needed to describe an average word image. While images of similar size are nowadays commonly used in image recognition with convolutional networks our main goal was rapid prototyping of various RNN architectures and therefore the size of input data was important factor affecting the total time of training.

Taking into account above mentioned considerations, we applied following preprocessing steps to each word image:

1. Dataset augmentation
2. Conversion to grayscale
3. Resizing

Firstly, we generated 10 additional augmented images by applying random shear (with shear factor from -0.20 to 0.20) to each original image. Secondly, the augmented images were converted to greyscale, on the basis of the assumption that color information provides little discriminative power in the case of handwritten text. Finally, every image was resized to the fixed height (28 pixels) and to the 50% of the original width. This was done to convert the image into format suitable for feeding into RNN where image height would correspond to the size of feature vector and image width to the number of timesteps in the input sequence.

We considered using convolutional layers for feature extraction, but initial experiments showed that it provided only minor improvement of accuracy and considerably increased the training time. We also tried simple image enhancement techniques (Gaussian blur for noise reduction and binarization) as additional preprocessing steps, which provided small increase of accuracy for Seq2Seq approach, but did not substantially affect accuracy of CTC approach.

3.2 Recurrent Neural Networks (RNN)

The core part of the approach presented in this work is based on Recurrent Neural Networks (RNN), a particular type of Neural Network that is characterized by the presence of self-connected nodes. These nodes enable the network to memorize and keep track of previous inputs, allowing it to store and access information over long periods of time. This particular ability allows the RNN to learn the context of the labels which turns out to be a big advantage in handwriting recognition since the characters of the words can not be considered as really independent components. In order to use RNNs for handwriting recognition images with handwritten text are interpreted as time sequences along single or multiple axis.

Even though classical RNN provide the network with some memory they are still quite limited in their ability to preserve old content due to the vanishing gradient problem [9]. In fact, as explained by the authors, it is hard for a RNN to bridge gaps of more than about 10 time steps between the moment in which the relevant input is presented and the relative target events. To address this problem we have used Long Short-Term Memory (LSTM) [10]. LSTMs are specially constructed RNN nodes to preserve long lasting dependencies. They consist of self-connected memory cell that can be compared to the classical RNN node and three gates that control output and input of the node. Each gate is in fact a sigmoid function of the input to the LSTM node. The first gate is an input gate which controls whether new input is available for the node. The second gate is a forget gate which makes possible for the node to reset activation values of the memory cell. he last gate is an output gate controlling which parts of the cell output are available to the next nodes.

Further improvement to the RNN models based on LSTM is achieved by the use of opposite two-directional layers or so-called Bidirectional Long Short-Term Memory (BLSTM). The goal of the forward layer is to learn context of the input by processing the sequence from the beginning to the end, while the backwards layer performs the opposite operation by processing the sequence from the end to the beginning. It was demonstrated [8] that this architecture performs better than a simple uni-directional LSTM.

3.3 Connectionist Temporal Classification approach

The feed-forward approach is similar to the original recurrent neural networks by the fact both architectures require a direct alignment between the input features and target variables. However in the real-world handwriting recognition problems it is much easier to segment text into words rather than individual characters. Achieving direct alignment between the image of input character and character target label would require a prior segmentation step. Being a very hard problem by itself its complexity keeps increasing in time since there is a high tendency of encountering possible errors already in the segmentation step. As a result this would also limit the context of the data learned by the RNN.

To target this problem the Connectionist Temporal Classification (CTC) approach was introduced, originally for speech recognition [6] and afterwards also for handwriting recognition [7]. CTC makes it

possible to avoid the previously mentioned direct alignment between the input variables and the target labels by interpreting the output of the network as a probability distribution over all possible label sequences on the given input sequence.

The last layer of the network is a softmax output with $N+1$, where N is the total number of labels (in case of handwriting recognition - characters). These outputs define the probability of observing each label or not observing a label at a given moment in time. By using single character probabilities it is possible to obtain a set of probabilities that correspond to a possible output sequences given an input one. This logic is formalized by equation

$$p(\pi|x) = \prod_{t=a}^{T} y^t_{\pi_t}, \forall \pi$$

where x is the input sequence, π a possible output sequence and $y^t_{\pi_t}$ the probability of a given label from π at time t.

A Special operator β is defined to match sequences from previous step π to the final output sequences l by removing blank labels and merging repeated labels. For example a possible sequence as β (dd_a_b) = (dab). Since more than one sequence π can correspond to a single final sequence then probability of a particular final sequence l is computed as follows:

$$p(l|x) = \sum_{\beta(\pi)=l} p(\pi|x).$$

For each given input x and sequence l, the maximum probability $p(l|x)$ is considered as the final output.

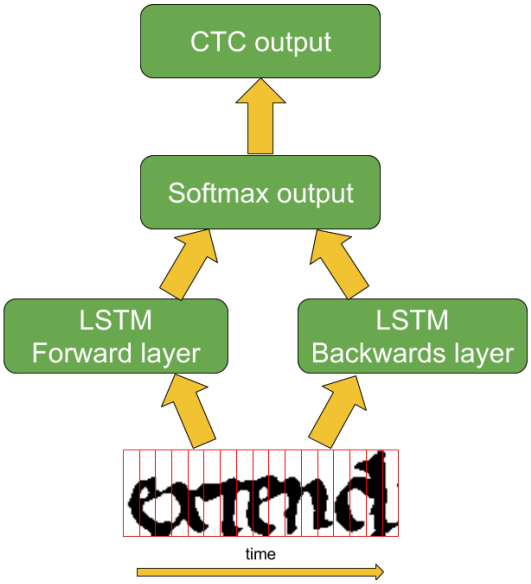

Figure 5: RNN architecture with Connectionist Temporal Classification output layer

For handwriting recognition problem a word image is interpreted as a time sequence where time is modeled along the width of the image with 1px slice as a single unit of time. Pixel values after preprocessing were used as input features for the network. The Figure 5 shows the general recognizer structure.

3.4 Sequence-to-Sequence Learning approach

Sequence-to-Sequence Learning (Seq2Seq) is based on the approach developed by Google researchers for the automatic sentence translation from English to French [15]. The main idea is to use two connected RNNs, the first RNN for reading an input sentence and mapping it to a fixed-dimensional vector

representation and the second RNN for decoding an output sequence from that representation. The approach is very suitable for machine translation, as input and output sequences can be of various lengths and ordering of words in each language can be different.

While in handwriting recognition it can generally be assumed that handwritten characters can be mapped to labeled characters in the same order, Seq2Seq approach still provides the advantage that no specific mapping between positions of characters in input image and target character labels is required.

The model consists of two connected RNNs, namely: the Image-RNN for the image encoding and the Label-RNN for the generation of the text label as shown in Figure 6. Each RNN is constituted of one or more LSTM or BLSTM layers. Weighted cross-entropy loss between a target sequence and a predicted sequence of characters is used as a cost function.

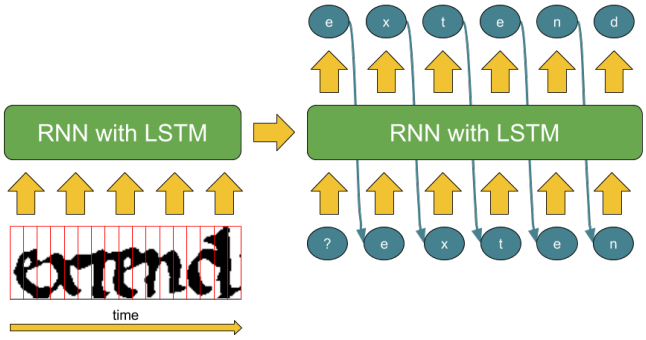

Figure 6: RNN architecture with Sequence-to-Sequence Learning

An input image is presented to the Image-RNN as a separate 1px slices with features corresponding to pixel values after preprocessing. The final output state of the Image-RNN is used as an initial state for the Label-RNN. As a number of hidden units in both RNNs can differ, a fully connected layer provides the mapping from the output state of the Image-RNN to the initial state of the Label-RNN.

At the first time step of the Label-RNN the ”start word” character is presented. The Label-RNN generates an output character, which is used as an input for the next time step and thus recursively the whole word is generated until a white space character is encountered. Both input and output characters are encoded as one-hot vectors.

4 Experiments

Initially we performed number of experiments to determine optimal configurations for each approach. The models were trained on both document sources (KNMP and Stanford) separately. The dataset was randomly divided into training data (90%) and hold-out data (10%) for validation and hyperparameter tuning. Mini-batch approach was used for training with each batch consisting of 128 word images. In each epoch all images were presented to the neural network in random order.

The CTC model was trained using a decaying learning rate. The learning rate was initially set to 0.01 and gradually decreased to 0.0001. For the Seq2Seq model we discovered that setting the learning rate to 0.1 ensured the best convergence in spite of occasional jumps of the loss function value. Other learning rates (0.01, 0.001) led to much slower convergence and considerably increased the total time of training. The moderate 0.75 dropout keeping probability was found to achieve an optimal balance between training speed and prevention of over-fitting as suggested by Pham et al [12].

BLSTM was used in all CTC experiments, but in case of Seq2Seq it turned out that BLSTM offered no increase in the performance, therefore most of Seq2Seq experiments were performed with uni-directional LSTM to achieve faster training. Limited improvement provided by BLSTM can be explained by prevalence of short words in the dataset as noted by Lee et al [11].

It was found that simply increasing the number of LSTM nodes in forward and backwards layers did not considerably influence the accuracy of the model, but rather lead to overfitting on the training set. We found that 128 nodes on each layer was an optimal configuration for the CTC model and 96 nodes in both Image-RNN and Label-RNN were optimal choice for the Seq2Seq model with good generalization performance and reasonable training time.

As it could be assumed that first characters of the word are more important for predicting the whole word, the cost function weights for the Seq2Seq model were chosen so that higher significance (weight values - 10.0, 7.0, 4.9, ..., 1.0) were assigned to the first letters of the word. The use of weighted cross-entropy loss function reduced convergence time of the model.

Initially we used preprocessed grayscale images to achieve optimal configuration of the models. Subsequent experiments showed that using binarized images with Gaussian blur offered consistent 5-7% increase in word-level accuracy for the Seq2Seq model, but did not affect accuracy of the CTC model.

We also examined the use of convolutional layers for feature extraction. 1-2 convolutional layers with up to 16 filters and sliding window with patch size from 3x3 to 7x7 were used to extract 1px slices of features from word images. Unfortunately, contrary to expectations, this approach did not provide increase of accuracy and thus was not used for the final recognizer.

After finding optimal configurations for both models we performed a number of experiments to compare them. Two test datasets which were not included in training process were used to compare accuracies of both models. We used results from evaluation dataset to improve our models before the final test. Using Levenshtein distance to find the closest word in provided lexicon (instead of using decoded word straight from the model) also improved accuracy on the final test dataset.

5 Results

5.1 Metrics

Two metrics were used to evaluate the performance of the handwriting recognition system:

- Word accuracy rate (WAR) - percentage of words with all characters correctly recognized
- Character accuracy rate (CAR) - average Levenshtein distance normalized by the length of the longest word and subtracted from 1 (thus 0 means no similarity and 1 means complete similarity).

5.2 Accuracy

The following table shows the accuracy of both approaches on the training dataset as well as on the evaluation and the final test dataset:

	Connectionist Temporal Classification		Sequence-to-Sequence Learning	
	Character Accuracy Rate (CAR)	Word Accuracy Rate (WAR)	Character Accuracy Rate (CAR)	Word Accuracy Rate (WAR)
KNMP training dataset	96.0%	88.2%	86.9%	73.4%
Stanford training dataset	96.2%	94.5%	88.7%	82.0%
Evaluation dataset	-	78.7%	-	68.3%
Final test dataset	-	**78.10%**	-	**72.79%**

Table 1: Accuracy of the CTC and Seq2Seq model

5.3 Impact of word length on accuracy

Figure 7 shows how accuracy of both models depends on the length of the word using the training dataset without any lexicon. The performance of the CTC model decreases faster at the beginning, but with longer words it performs slightly better than the Seq2Seq model.

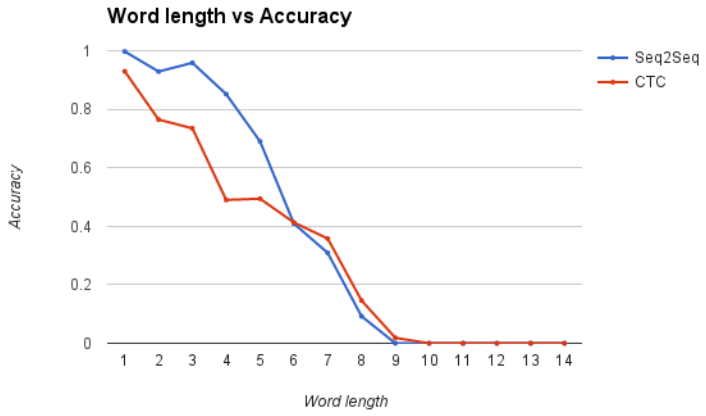

Figure 7: Impact of word length on accuracy (on the training dataset)

6 Discussion & Conclusion

In this paper we have evaluated two RNN architectures for handwritten text recognition based on Connectionist Temporal Classification and Sequence-to-Sequence Learning approach. The obtained results (see Table 1) are comparable with 81.5% average recognition rate over all manuscripts in Monk System [14] and in similar range as 69.8% word accuracy rate achieved in ICDAR2015 competition [13]. However it should be noted that our models were fine-tuned to specific data sources and might show varying performance on different datasets.

Although both methods showed promising results, the CTC model consistently outperformed the Seq2Seq model on both training and test datasets. During the experiments we observed that the Seq2Seq model had tendency to overfit on the training data and was not generalizing well on the validation data, while the CTC model was more robust against overfitting and as a result had higher accuracy on previously unseen data.

Figure 7 shows that Seq2Seq was better at predicting short words, but CTC model had higher accuracy predicting longer words. That could be explained by the fact that the CTC model predicts characters ”on the fly” while the Seq2Seq model first represents the whole image of handwritten word as a fixed-size vector and then decodes it to generate predicted label. As a result the Seq2Seq model has a tendency to learn whole words instead of separate character sequences. Thus CTC approach seems more natural choice for problems where an input sequence has the same ordering as an output sequence.

Various experiments during training phase showed that image preprocessing and dataset augmentation by additional image transformations can improve accuracy rates and prevent overfitting. Some additional preprocessing steps could be suggested, for example, vertical alignment of letters and removal of unnecessary white-space at the top and the bottom of word images. The use of more advanced features instead of raw pixel intensities (for example, HOG [5]) could be considered, although experiments with automatic feature learning using convolutional layers showed that other factors are probably more important for RNN performance than feature engineering.

Even though our models showed good results on provided datasets it is hard to infer the performance of examined methods on different handwritten text sources. In the future work the use of more advanced methods like Multi-Dimensional Long-Short Term Memory (MDLSTM) and attention mechanisms could be considered [4][11]. However usage of these methods also means increased complexity and training time of the models and therefore was not used in this research.

7 Acknowledgments

We are grateful to prof. dr. Lambert Schomaker and Sheng He for valuable advise during this research and for providing an access to the KNMP and Stanford dataset. For any questions regarding the dataset please contact prof. dr. L. Schomaker (email: *schomaker@ai.rug.nl*).

References

[1] KNMP Chronicon Boemorum dataset in Monk System. http://application02.target.rug.nl/cgi-bin/monkweb?cmd=showbook&pagid=*&db=1201.

[2] Monk System. http://www.rug.nl/society-business/target/projects-and-rd/target_projects/monk?lang=en.

[3] Stanford CCCC dataset in Monk System. http://application02.target.rug.nl/cgi-bin/monkweb?cmd=showbook&pagid=*&db=61200.

[4] Théodore Bluche and Ronaldo Messina. Scan, attend and read: End-to-end handwritten paragraph recognition with mdlstm attention. *arXiv preprint arXiv:1604.03286*, 2016.

[5] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893. IEEE, 2005.

[6] 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. ACM, 2006.

[7] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):855–868, 2009.

[8] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005.

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

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for OCR in the wild. *arXiv preprint arXiv:1603.03101*, 2016.

[12] Vu Pham, Théodore Bluche, Christopher Kermorvant, and Jérôme Louradour. Dropout improves recurrent neural networks for handwriting recognition. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 285–290. IEEE, 2014.

[13] Joan Andreu Sanchez, Alejandro H Toselli, Veronica Romero, and Enrique Vidal. ICDAR 2015 competition HTRtS: Handwritten text recognition on the transcriptorium dataset. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 1166–1170. IEEE, 2015.

[14] Lambert Schomaker. Design considerations for a large-scale image-based text search engine in historical manuscript collections. *it-Information Technology*, 58(2):80–88, 2016.

[15] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.