

# Semi-Supervised Learning for Cursive Handwriting Recognition using Keyword Spotting

Volkmar Frinken

Computer Vision Center, Autonomous University of Barcelona  
Edifici O, 08193 Bellaterra, Barcelona, Spain  
*vfrinken@cvc.uab.es*

Markus Baumgartner, Andreas Fischer, Horst Bunke  
Institute of Computer Science and Applied Mathematics, University of Bern  
Neubrückestrasse 10, CH-3012 Bern, Switzerland  
*markus.baumgartner@students.unibe.ch, {afischer,bunke}@iam.unibe.ch*

## Abstract

*State-of-the-art handwriting recognition systems are learning-based systems that require large sets of training data. The creation of training data, and consequently the creation of a well-performing recognition system, requires therefore a substantial amount of human work. This can be reduced with semi-supervised learning, which uses unlabeled text lines for training as well. Current approaches estimate the correct transcription of the unlabeled data via handwriting recognition which is not only extremely demanding as far as computational costs are concerned but also requires a good model of the target language. In this paper, we propose a different approach that makes use of keyword spotting, which is significantly faster and does not need any language model. In a set of experiments we demonstrate its superiority over existing approaches.*

## 1 Introduction

The automatic transcription of handwritten text, such as letters, manuscripts, or books has received substantial attention over the last decades [13]. Current state-of-the-art approaches are learning-based systems that automatically infer, from a given set of training data, the rules how to recognize new text. However, due to the variety of different writing styles even among writers with the same cultural and educational background, large amounts of training data are needed for each language and writing style to be able to model characters, words, and sentences accurately.

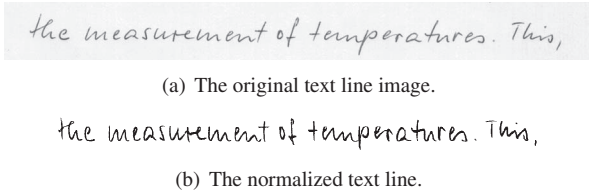
The training data usually consist of images of text

lines labeled with their machine-readable transcription, which has to be created manually in a tedious and costly process. On the other hand, unlabeled data are usually abundant and can be gathered at little or no cost. Learning approaches that make use of both types of data, labeled and unlabeled, are called semi-supervised learning.

Semi-supervised learning has received remarkable attention in the last few years. Most of the existing works, however, deal with the standard classification scenario where a single point in a feature space has to be mapped into the label space [3]. In the current paper, a more general problem is considered in the sense that a (possibly long) sequence of feature vectors has to be mapped to a (usually much shorter) sequence of labels, i.e., words or characters.

A promising approach to semi-supervised learning for writer independent handwriting recognition is self-training [3]. Under this paradigm one starts with an initial system trained on the available labeled data. This system is then used to select words from the set of unlabeled data that have been recognized with high confidence. The most confidently recognized samples are assumed to be correct and added to the training set. Using the augmented training set, a new system is created. This procedure of enlarging the training set can be continued for several iterations. Increasing the training set as much as possible while keeping noise out of the training set is one of the most crucial aspects of the system. Therefore in this paper, we investigate several decision rules that decide which elements are to be added to the training set.

For the specific task of writer-independent semi-



**Figure 1. A visualization of the text line preprocessing.**

supervised learning for handwriting recognition, only few publications exist. In [5], the authors use a complete text transcription system that recognizes every line of the unlabeled data using a large language model. Words recognized with a sufficient confidence are used for retraining.

In this paper we propose and investigate approaches that select the elements used for retraining via keyword spotting. It has been shown that keyword spotting reduces the computational costs dramatically compared to text line transcription. Secondly, keyword spotting does not need any language information. Hence, this approach is specifically suited in cases where little or no language information is available.

The rest of this paper is structured as follows. Data preprocessing and the handwriting recognition system are presented in Section 2. In Section 3, details about the keyword spotting approach are explained. Semi-supervised learning in general and self-training in particular are introduced in Section 4. An experimental evaluation is given in Section 5 and conclusions are drawn in Section 6.

## 2 Handwriting Recognition

### 2.1 Preprocessing

To focus on the recognition task, we consider perfectly extracted text lines only. Once extracted, the text lines are normalized in order to cope with different writing styles, as far as skew angle, writing slant, text height and text width is concerned. The result of the preprocessing steps can be seen in Fig. 1.

A normalized text line image is then represented by a sequence of feature vectors, using the sliding window approach. The window has a width of one pixel and moves in steps of one pixel from left to right across the text line. At each position, 9 feature are extracted. These are the 0<sup>th</sup>, 1<sup>st</sup> and 2<sup>nd</sup> moment of the black pixels' distribution within the window as well as the position of the top-most and that of the bottom-most black

pixel, the inclination of the top and bottom contour, the number of vertical black/white transitions, and the average gray scale value between the top-most and bottom-most black pixel. For further details on the text normalization and feature extraction step, we refer to [9].

### 2.2 BLSTM Neural Networks

The considered recognition system is based on a recently developed recurrent neural network, termed *bidirectional long-short term memory* (BLSTM) neural network [8]. This is an architecture with two separate hidden layers, consisting of LSTM cells. An LSTM cell is a compound of several nodes, arranged in such a way that it is possible to store information over arbitrary long time steps. In this bidirectional architecture, the sequence is fed into the network from both directions, left-to-right and right-to-left. Hence two different hidden layers are needed. The output layer consists of one node for each possible character and sums up the activation levels from both hidden layers at each position in the text line. Using the *softmax* normalization, the final output activations can be treated as a vector indicating the probability for each letter to occur at this position. The output of the network is therefore a matrix of probabilities for each letter and each position.

In the recognition step, a path through that matrix is sought after that maximizes the product of the single letter probabilities along its way. The sequence of characters along the path is the final output of the recognizer. To restrict the search space to paths representing actual words, the token passing algorithm can be used. For more details about BLSTM networks and the CTC token passing algorithm, we refer to [8].

## 3 Keyword Spotting

Keyword spotting refers to the process of retrieving all instances of a given word or phrase from a collection of documents. Usually, keyword spotting can be classified into query-by-example (QBE) and query-by-string (QBS) approaches. The former category can be considered as being a sub-image retrieval task where all positions of the database similar to a selected example are to be returned. The later one is the task of retrieving an arbitrary character sequence. Hence, this task is often approached using learning based systems that model characters individually [2, 4, 7].

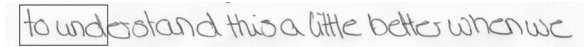
The BLSTM neural network introduced in the previous section can be used as a learning-based keyword-spotting system by modifying the the sequence of output activations returned by the neural network instead of using the token passing algorithm. In short, the letter



(a) Returned log Likelihood: -1.7125



(b) Returned log Likelihood: -8.4097



(c) Returned log Likelihood: -11.0900

**Figure 2. Search results for the word “found”.**

probability matrix is extended by an additional pseudo-character \* with a constant value of 1 and the keyword is extended by adding this \* symbol at the beginning and the end. Then, dynamic programming is used to find the best path that crosses through the keywords characters but bypasses the remaining content of the text line. The result of this procedure is the position and probability of the keyword at its most likely position. For more details, we refer to [7]. An example output of the system is shown in Fig. 2.

### 3.1 Combination of Several Systems

Clearly, two differently initialized networks are likely to produce a different output, even when trained on the same training data. Therefore the performance of such a keyword spotting system can greatly be improved by generating several networks and combining the outputs [6]. Consequently, in this paper, an ensemble of several neural networks is used, each of which spots a given keyword separately. The logarithmic probabilities are then averaged and divided by the number of characters to return a normalized matching score of the system.

## 4 Self-Training

Classic supervised learning algorithms use both data elements and their class labels to infer the relation between input features and the corresponding class label. In semi-supervised learning, unlabeled data, i.e., elements without a class label, are used as well. In handwritten text recognition, unlabeled data are images of written text, which are ubiquitously available.

One way to achieve semi-supervised learning is via self-training [11, 5]. An initial recognizer is trained using the labeled data only. Afterwards, during the self-training iterations, the recognizer is used to label

the entire set of unlabeled data and assigns a confidence measure to each classification. The most confidently classified elements are assumed to be labeled correctly and added to the training set, which is then used to train a new recognizer. Due to its descriptive nature, self-training is generally applicable to every form of learning-based classification and recognition system. This is in contrast to most other semi-supervised learning techniques that require certain prerequisites, such as training samples being represented as single elements of a vector space, rather than feature vector sequences.

### 4.1 Proposed Approach

For handwriting recognition, current approaches typically consist in transcribing all words or text lines in the unlabeled data set and select the most confidently recognized ones. The disadvantage of that approach is the need for either a word segmentation algorithm in case of single word recognition [1] or a language model for text line recognition [5]. Furthermore, text line transcription is computationally expensive.

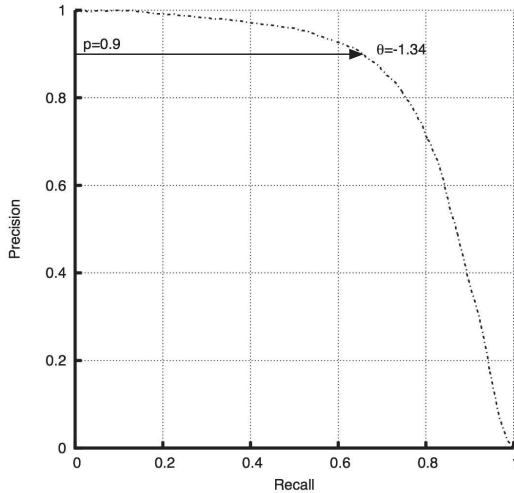
In comparison, estimating the matching score of a keyword is done by means of dynamic programming using the output activation matrix of the BLSTM NN, which itself is computed in linear time. Hence, the complexity for keyword spotting is only  $O(L \cdot n)$ , where  $L$  is the length of the text line and  $n$  indicates the number of characters. This is a great advantage in terms of computational speed when compared to transcription-based approaches using a lexicon of words. Here, the time complexity would be  $O(LN^2)$  for a lexicon of size  $N$ . For transcription-based systems that deal with texts written in natural language, typical values for  $N$  are several ten thousands, whereas the number  $n$  of characters in an alphabet is usually below one hundred [4].

As a result, keyword spotting is several orders of magnitude faster than text line transcription. To give a quantitative evaluation of the run time, consider that it takes several minutes to decode a text line using a language model of 20,000 words, while a keyword can be spotted with the same system in about one millisecond [7].

### 4.2 Precision Based Thresholds

Based on the previous considerations, we propose to use keyword spotting instead of text transcription for semi-supervised learning. That is, the most confidently spotted words are used for self-training. The procedure works in detail as follows.

All keywords in a given list are first spotted on the validation set to estimate the expected *precision* as a



**Figure 3. Determining the likelihood threshold using the desired precision.**

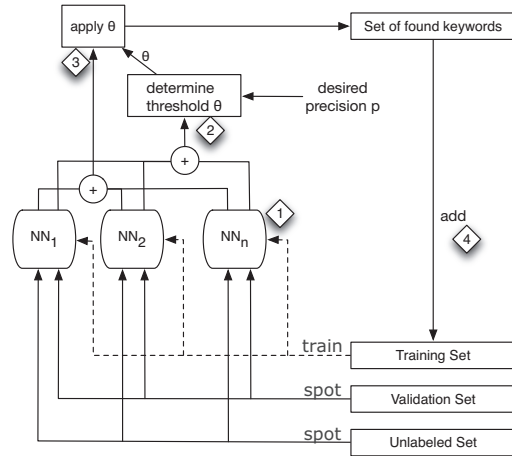
function of the *recall* when applying a global threshold  $\theta$  on the matching score. Recall reflects the returned percentage of all occurring instances of the keyword, while precision informs us of the percentage of correct results in the returned data. This enables us to define retraining strategies based on the expected noise that is added to the training data.

Using the information gathered on the validation set, the global matching threshold  $\theta$  is set to return results with a desired precision  $p$  (cf. Fig. 3). This factor is an external parameter of the system. Finally, all words in the keyword list are spotted on the entire set of unlabeled data and every word that has been spotted with a matching score higher than  $\theta$  is added to the training set.

In Fig. 4, an overview of the proposed system is given. An initial, labeled training set is used to train an ensemble of BSLTM neural networks (1). With these networks, keywords are spotted on the validation set using a combination technique. The performance on the validation set is used to compute the threshold  $\theta$ , according to a desired precision value (2). Next, the ensemble is used to spot the same keywords on the set of unlabeled data. All positions that are found with a likelihood greater than  $\theta$  (3) are considered to be correct. Finally, these words are added to the training set (4).

### 4.3 Reference System

As a reference system, we use the transcription based self-training system published in [5]. In that system, transcription-based self-training is performed using an



**Figure 4. Overview of the proposed system.**

ensemble of ten BSLTM neural networks. In each iteration, all ten networks decode each text line. Then, the returned hypotheses of the networks are aligned and the fraction of networks agreeing on a word serves as a recognition confidence estimate. Three retraining rules have been proposed for this approach. The *High Threshold* retraining rule chooses those words that are recognized by all ten networks. A second, more refined threshold is the *Medium Threshold*. It is set so that all elements added are expected to be more likely to be correct than wrong as can be estimated from the behavior on the validation set. The third threshold is the *Low Threshold* in which all words, regardless of their recognition confidence, are added to the training set.

The *High Threshold* and the *Low Threshold* retraining rule form the two extreme values in the data quality vs. data quantity trade-off, while the *Medium Threshold* can be seen as a good balance between the two extremes.

## 5 Experimental Evaluation

### 5.1 Setup

The impact using keyword spotting in the self-training iterations is evaluated on continuous text lines of the IAM database [10]<sup>1</sup>. The database is split up into a working set of 6,161 text lines, a validation set of 920 text lines and a writer independent test set of 929 text lines. The three sets are writer disjunct, i.e., a person who has contributed to any of the three sets did not con-

<sup>1</sup><http://www.iam.unibe.ch/fki/databases>

tribute to any of the other sets. The working set is randomly split up into a training set consisting of 1,000 labeled text lines and a set of 5,161 unlabeled text lines.

For the ensemble, ten neural networks are randomly initialized and trained on the labeled training set. Using the ensemble for spotting keywords on the validation set, a likelihood threshold is determined according to a desired precision and the keywords are spotted on the set of unlabeled data. We selected 3,421 keywords to be spotted. The keywords were chosen to be the most frequent English words without any stop words<sup>2</sup>.

In a first set of experiments, the effects of using several different precision thresholds are compared to each other. Five precision thresholds are investigated,  $\theta_{prec}^1 = 0.5$ ,  $\theta_{prec}^2 = 0.8$ ,  $\theta_{prec}^3 = 0.9$ ,  $\theta_{prec}^4 = 0.95$ , and  $\theta_{prec}^5 = 0.99$ .

In a second set of experiments, an extension is investigated in which several instances of the same word can be added to the training set several times, depending upon how confidently they are spotted. This way, both the quantity and quality of the elements added to the training set can be increased. Let  $S_{\theta_{prec}}$  denote the set of words selected with the precision threshold  $\theta_{prec}$ . To add more confidently recognized words more often to the training set than less confidently recognized words, the following six setups are used.

- A =  $S_{0.8} + S_{0.9}$
- B =  $S_{0.8} + S_{0.95}$
- C =  $S_{0.8} + S_{0.99}$
- D =  $S_{0.8} + S_{0.95} + S_{0.99}$
- E =  $S_{0.8} + S_{0.99} + S_{0.99}$
- F =  $S_{0.8} + S_{0.9} + S_{0.95} + S_{0.99}$

As reference values, the results of the transcription-based approach proposed in [5] are given as well. In this system, the transcription confidence-based thresholds *High Threshold*, *Medium Threshold*, and *Low Threshold* are used.

The recognition accuracy is evaluated on the test set at first before applying self-training and then after each iteration. The number of retraining iterations for all experiments is fixed to 4. Also, the algorithm used to recognize the text lines of the test set used more restrictive pruning parameters than the algorithm in the original publication of the reference system [5]. Both of these limitations had to be done to keep computational expenses within reasonable bounds. Note that without this pruning, generating the new training set in one self-training iteration takes about 6 to 7 days on a single

<sup>2</sup>We used the stop words given in the SMART project [12]. <http://jmlr.csail.mit.edu/papers/volume5/lewis04a/all-smart-stop-list/english.stop>

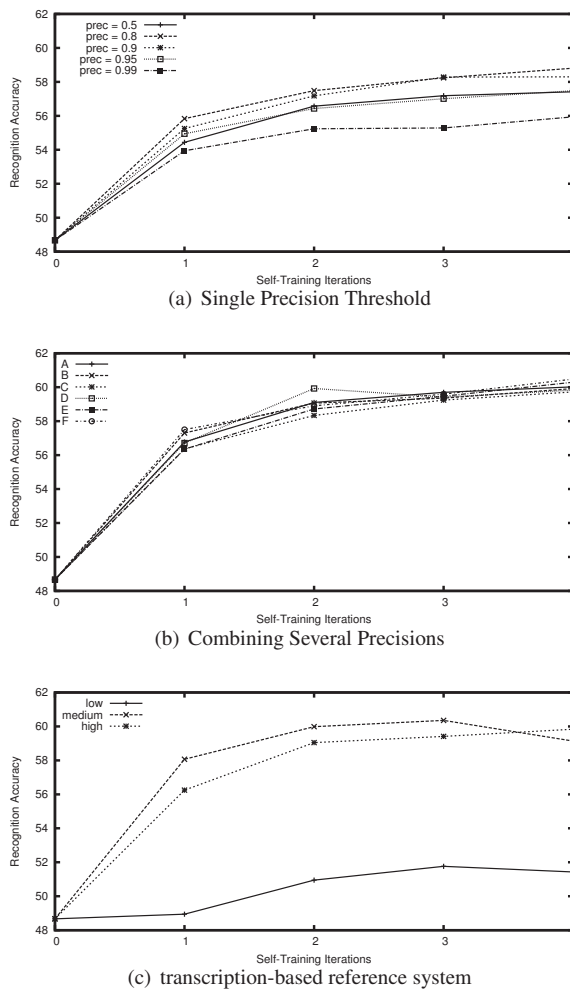


Figure 5. The performance of the system.

computer, while the proposed approach needs around 4 to 5 hours.

## 5.2 Results

Fig. 5 show the performances of the systems using single precision value, the performance of the extended system combining several precision values and the performance of the transcription-based reference system.

Several observation can be made. First of all, all investigated retraining rules achieve a statistically significant performance increase compared to the one trained on the initial training set. The performance of the retraining rules that make use of a single precision value in Fig. 5(a) clearly demonstrate the importance of a good balance between data quality and data quantity. The most conservative retraining rule that only used images that are found with a precision of  $prec = 0.99$  has the lowest performance. By relaxing the precision

threshold, the performance increases until it reaches a maximum at  $prec = 0.8$ . The least restricting retraining rule  $prec = 0.5$  again performs significantly worse.

From Fig. 5(a) and (b) it becomes apparent that retraining rules based on a single precision value perform overall not as good as the ones that combine the retraining sets of several precision values. Interestingly, the specific combination of the thresholds does not seem to have a substantial impact. The networks retrained using the worst combination,  $D = S_{0.8} + S_{0.95} + S_{0.99}$  reach an average recognition accuracy of 59.82%, while retraining using the best combination,  $F = S_{0.8} + S_{0.9} + S_{0.95} + S_{0.99}$  leads to an accuracy of 60.48%.

The comparison with the reference system shows that the keyword spotting based retraining rules perform worse than the best transcription based retraining rules during the first 3 iterations. However, the performance increase using keyword spotting based self-training continues longer and the final results after the fourth iteration are better than the reference system.

As a final comparison and to evaluate how good the BLSTM NN system can get on this database, we trained ten neural networks on the entire working set of 6,161 labeled text lines and reached an accuracy of 71%.

## 6 Conclusion

In this paper, we propose to use keyword spotting as a means to select words for self-training. Doing so increases the speed of a self-training iteration substantially. Furthermore, by eliminating the need for a language model, this approach becomes suitable for a broader range of applications. Especially worth mentioning are applications where semi-supervised learning techniques are useful, due to a lack of existing annotated text data, such as historical documents or recognition systems for new languages.

We demonstrated an average increase in recognition accuracy from 48.67% to 60.48% using a new set of retraining rules. In addition to being much faster, the new rules perform slightly better than current transcription-based approaches.

Future work includes experiments on historic data sets and an integration of this approach into a complete bootstrapping framework for new recognition systems. In order to further increase the performance, co-training approaches using different keyword spotting methodologies might also be investigated.

## Acknowledgments

We thank Alex Graves for kindly providing us with the BLSTM Neural Network source code. This work

has been supported by the European project FP7-PEOPLE-2008-IAPP: 230653, the Swiss National Science Foundation (Project CRSI22\_125220), the Spanish project TIN2009-14633-C03-03, and the Spanish MICINN under the MIPRCV "Consolider Ingenio 2010" CSD2007-00018 project.

## References

- [1] G. R. Ball and S. N. Srihari. Semi-supervised Learning for Handwriting Recognition. In *10th Int'l Conf. on Document Analysis and Recognition*, pages 26–30, 2009.
- [2] J. Chan, C. Ziftci, and D. Forsyth. Searching Off-Line Arabic Documents. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1455–1462, 2006.
- [3] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, 2006.
- [4] A. Fischer, A. Keller, V. Frinken, and H. Bunke. Lexicon-Free Handwritten Word Spotting Using Character HMMs. *Pattern Recognition Letters*, accepted for publication, 2011.
- [5] V. Frinken and H. Bunke. Self-Training for Handwritten Text Line Recognition. In *15th Iberoamerican Congress on Pattern Recognition*, pages 104–112, 2010.
- [6] V. Frinken, A. Fischer, and H. Bunke. Combining Neural Networks to Improve Performance of Handwritten Keyword Spotting. In N. El Gayar, J. Kittler, and F. Roli, editors, *9th Int'l Workshop on Multiple Classifier Systems*, LNCS, pages 215–224, 2010.
- [7] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke. A Novel Word Spotting Method Based on Recurrent Neural Networks. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2011, preprint.
- [8] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber. A Novel Connectionist System for Unconstrained Handwriting Recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 31(5):855–868, 2009.
- [9] U.-V. Marti and H. Bunke. Using a Statistical Language Model to Improve the Performance of an HMM-Based Cursive Handwriting Recognition System. *Int'l Journal of Pattern Recognition and Artificial Intelligence*, 15:65–90, 2001.
- [10] U.-V. Marti and H. Bunke. The IAM-Database: An English Sentence Database for Offline Handwriting Recognition. *Int'l Journal on Document Analysis and Recognition*, 5:39–46, 2002.
- [11] K. Nigam and R. Ghani. Understanding the Behavior of Co-training. In *KDD-2000 Workshop on Text Mining*, pages 105–107, 2000.
- [12] G. Salton. *The SMART Retrieval System – Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1971.
- [13] A. Vinciarelli. A Survey On Off-Line Cursive Word Recognition. *Pattern Recognition*, 35(7):1433–1446, 2002.