# Offline ICR with Deep Learning

Ting XU＊, Wei MING＊, Masahiro OZAWA＊＊

## 要旨

手書き文字認識（Intelligent Character Recognition: ICR）の研究は，一文字単位の英数字の認識から始まり，筆記体の単語単位の認識まで発展してきている。しかし，現在でも実用的な商用システムは数えられる程度である。しかも，これらの商用システムは性能を向上するために，辞書，または言語モデルを用いていることが多い。この「語彙内」制約により，ICR技術を多くの実社会の文書（臨床記録，財務書類，流通の取引記録など）に適用する上で制限が発生している。それは，辞書に登録されていない語句が頻繁に含まれることで，認識率の低下に繋がるためである。

この問題に対処するために，辞書に登録されていない手書きの語句を認識するICRエンジンを開発した。これにより幅広い分野へのICR技術の適用が可能になり，辞書に登録されていない名前，住所，略語，数字または通貨を含む文書に対しても，既存の商用システム以上の高精度な認識を実現できる。開発したエンジンの有効性は，複数の適用分野で確認できている。RPA（Robotic Process Automation）システムに統合することで，作業の生産性と従業員の満足度が大幅に向上することが分かっている。

## Abstract

Handwriting recognition, or Intelligent Character Recognition (ICR), has started from recognizing single alphanumeric character to recognizing cursive whole words, which few commercial systems can recognize well. Moreover, these systems are often guided by a dictionary or a language model to enhance performance. With the success of Deep Learning, many recent works on ICR extract handwriting features using Convolutional Neural Networks (CNN) provided that all possible words are prescribed by a dictionary. This “in-vocabulary” constraint, however, prevents ICR’s application from clinical forms, financial reports, logistic transaction records, etc., which frequently contain named entities, address, acronyms, digits, or currency that are not found in a dictionary.

To address this problem, we developed a dictionary-free ICR engine that opens broader ICR applications, including documents containing named entities, address, acronyms, digits, or currency, with better performance compared to leading competitors. The effectiveness of the proposed engine has been proved in multiple applications. When integrated into RPA (Robotic Process Automation) system, the work productivity and employee satisfaction are increased significantly.

＊ Konica Minolta Laboratory U.S.A., Inc
＊＊ Konica Minolta, Inc.

## 1 Introduction

Despite the widespread creation of electronic documents and the promotion of paperless in office environment, paper documents are still heavily used in many fields, such as medical interview forms, government office application forms, banking application forms, etc. The operations, like data entry and verification on these paper-based documents are the tedious daily tasks performed by human workers. The resource turnover on data processing is also a severe issue. Thus, automating these operations has great economic and social needs. It will not only result in time reduction, cost reduction and error reduction, but also solve the problem of resource shortage and relieve people from boring tasks to creative work.

Automating office data processing is getting more and more attentions these years. There are satisfactory solutions on market that convert cleanly printed documents into digital form through Optical Character Recognition (OCR). Yet, the solution to convert handwritten documents (ICR) still face many challenges. Needless to say, the technology of handwriting recognition is the top challenge among them. Although there are some solutions on market that support written documents, they stay at the level of handprint character recognition of words that can be found in a dictionary.

Putting user needs on top of head, we target to the recognition of a word or a phrase that is an arbitrary sequence of characters that include digits and special characters. We focused on the development of recognition engine with deep learning technology, and meantime developed entire processing workflow, from pre-processing, layout analysis to post-processing, in order to provide higher readability. In experiments, we have achieved the state-of-the-art recognition accuracy of 96% and 83% for character and word level respectively. In this report, we focus on a deep CNN based method for handwritten recognition engine, targeting the challenging problem of unconstrained transcription of handwritten word images.

The rest of the paper is organized as follows: In Section 2, we briefly introduce past research and development on handwriting recognition technology. In Section 3, we describe the proposed solution to recognizing words in which characters have no correlation with each other. In Section 4, we present preliminary evaluation results. Conclusions are given in Section 5.

## 2 Previous Work

Handwriting recognition is the task of transforming a language represented in its spatial form into its symbolic representation. The brief introduction on the recognition system and approaches will be given as follows.

### 2. 1 Online Recognition vs. Offline Recognition

Based on input device, handwriting recognition can be classified into two types, offline character recognition and online character recognition. Online recognition refers to the methods and techniques that automatically convert a writing trajectory into symbolic representation when it is written using a digitizer or a PDA device, where the pen positions with temporal order information are recorded. Fig. 1 gives an example of touch based and tracking based online systems. Since the rich information and distinguishable features can be obtained and used in recognition, the online recognition has achieved satisfactory accuracy and is widely used in various PDA devices.

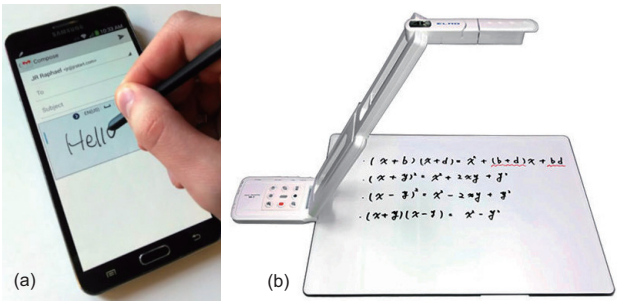



Fig. 1 (a) Touch based input device.
(b) Tracking based input device.

In contract to online recognition, offline handwriting recognition refers to the process of recognizing writings that have been scanned or captured with camera from a piece of paper or a surface (ex. whiteboard), after the writing process. Thus, the temporal and other helpful information, such as the order of pen-on and pen-off, the number of strokes, movements, the direction and speed of writing, the pressure applied, etc. do not exist. The only information available in recognition is a static image.

Since the information obtained in online and offline handwriting processes are very different, the recognition approaches are very different as well. Generally, the time-sequence based analysis and image feature based analysis are the major differences in these two cases. Fig. 2 gives an example of different features used in two different analysis.

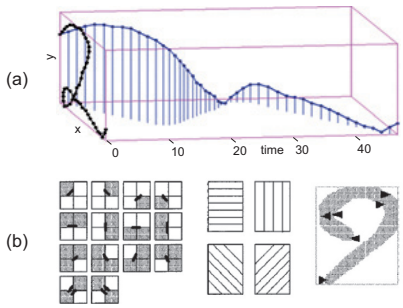



Fig. 2 (a) Time sequence features.
(b) Topologic and statistic features.

Technically, offline handwriting recognition is much more challenging than online recognition. It is not only because less information can be used in offline recognition than in online recognition, but also because inconsistent or unstable capture conditions make the recognition much more difficult. So far, satisfactory products are limited to a few domain applications under specific conditions, such as recognition of postal address and amount on bank checks on specified locations or within boxes.

### 2. 2 Related Work on Offline Recognition

As mentioned above, offline handwriting recognition is one of the most challenging topics in pattern recognition and image processing fields. Although the recognition of isolated characters has become mature (Convolutional Neural network based technology can achieve about 99 % of recognition rate for isolated characters for both western and Asian languages [1-3].), the word or character sequence recognition has yet reached to this level of accuracy. Moreover, most of current work on offline handwritten recognition assume all the possible words to be recognized are known beforehand.

Most of the current word recognition methods can be classified into three types:

1. Lexicon-based approaches where word recognition is modeled as a classification problem and solved using Convolutional Neural Networks (CNN). Here all words that need to be recognized are among a predefined dictionary or lexicon;
2. Sequence based approach where a word is treated as a character sequence which is recognized using Recurrent Neural Networks (RNN);
3. Attribute based approach where lexical attributes of a word are learnt by Convolutional Neural Networks (CNN). Learned attributes are then mapped to a word string using RNN or optimizing a specific objective function.

Lexicon-based approaches have been shown effective on scene text recognition (OCR on natural images) [4, 5]. However, the requirement of a predefined dictionary poses serious limitation for many real-word applications. For example, transcribing people's names, phone number, date, address, email, etc. requires recognizing out-of-vocabulary (OOV) words.

Attribute based methods achieve state-of-the-art recognition accuracy across several standard handwritten word recognition benchmarks, which are previously dominated by sequence based methods. Compared to RNN, CNN has less training and inference time because the computation can be parallelized. Meanwhile, attribute based learning uses training data more efficiently than lexicon-based approach because many attributes are shared thus not every word needs to appear in the training set. For example, in a training set, a word "abstraction" may appear only 3 times, but the attribute "does TION appears in the second half of the word?" may appear many more times.

Even though CNN predicted attributes can achieve over 90% word accuracy in IAM and RIMES benchmarks [6], a predefined lexicon is still required for transcription because by design, these attributes cannot be directly converted back into word strings. In fact, all current designs of lexical attribute vectors and their label embedding processes require predefined lexicons ([7-9]).

## 3 Unconstrained Word Recognition

Unconstrained word refers to a text or digit string that does not exist in a dictionary, like bank account numbers, claim numbers, etc. We propose here an invertible label embedding (encoding) algorithm to embed character strings into an Euclidean vector space as attribute vectors, and uses a CNN to learn and predict attribute vectors of handwriting images in this Euclidean vector space, and then directly decodes a predicted attribute vector into a character string using a decoding algorithm without requiring a lexicon.

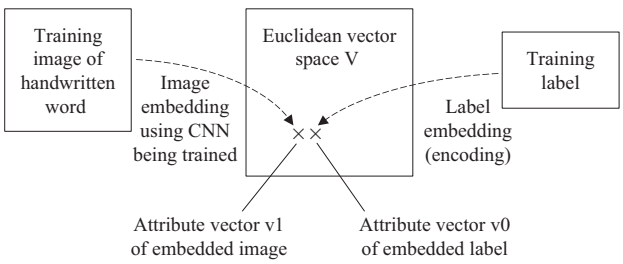



Fig. 3 Image embedding and label embedding during training.

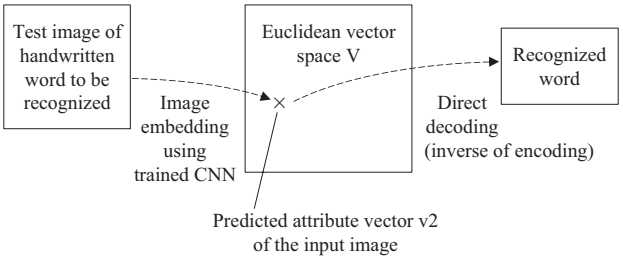



Fig. 4 Direct decoding of lexical attribute vectors during inference.

As shown in Fig. 3, a convolutional neural network (CNN) is trained to perform image embedding to embed images of handwritten words into an Euclidean vector space V. During training, in each iteration, a training image is inputted into the neural network to calculate a first attribute vector v1 in the Euclidean vector space V ("image embedding"), and the training label (the word) is embedded into the same Euclidean vector space V using the invertible encoding algorithm (described in detail later) as a second attribute vector v0 ("label embedding (encoding)"). The weights of the neural network are updated to minimize a loss function which measures the distance between the attribute vectors v1 and v0 in the Euclidean vector space. Thus, during training, the label embedding step is used to construct the ground truth of training samples: the ground truth word label is encoded into its corresponding attribute vector as ground truth. The trained neural network is able to predict an attribute vector from an input word image. This neural network training process is summarized in Fig. 5.

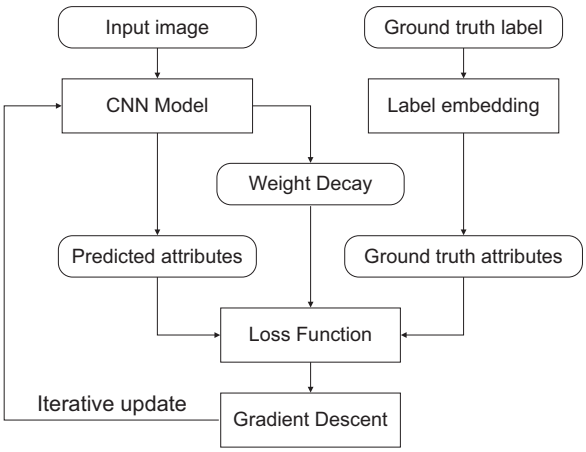



Fig. 5 Training process of a CNN model.

In this development, a L_2 loss function and stochastic gradient descent are used to train a 11-layer VGG-based CNN [10], which includes a horizontal Spatial Pyramid Pooling layer [11] before the fully-connected layers to enable arbitrary input image size in the horizontal direction. This is helpful because a CNN would otherwise require input images to have the same size, while the length of the word image may vary greatly as compared to its height.

To recognize a target handwriting image (Fig. 4), the target image is inputted into the trained neural network to predict an attribute vector v2 in the Euclidean vector space V ("image embedding"). A decoding process is then applied to the predicted attribute vector v2, using a decoding algorithm which is the inverse of the encoding (label embedding) algorithm used in the training process. The result of the decoding process is the recognition result, i.e. the character string that is recognized. This image recognition process is summarized in Fig. 6.

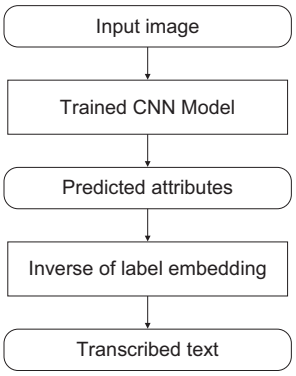
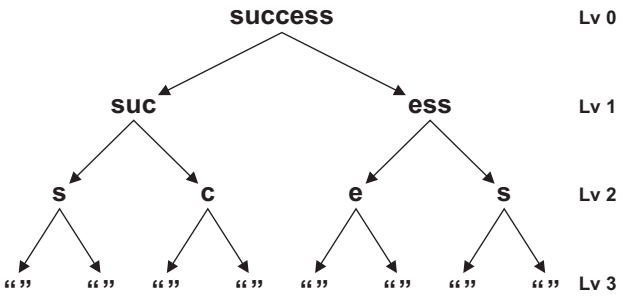


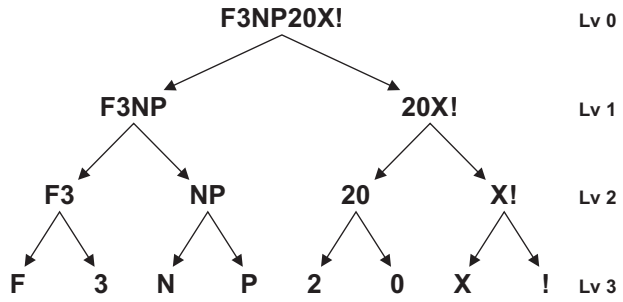

Fig. 6 Recognition process (inference) using a trained model.

The invertible encoding (label embedding) and decoding algorithms will be referred to as "invertible Pyramidal Histogram of Characters" (iPHOC) from now on. It is assumed that all characters of the string being encoded belong to a known and fixed alphabet $\Sigma$. The encoding of a character string into an iPHOC attribute vector uses a recursive bisection and histogram computation process. In each bisecting step, if the string being bisected has an odd number of characters, its middle character is omitted in the next level child strings. Thus, the child strings at each level always have the same number of characters. If the string being bisected has an even number of characters, it is deemed to have an omitted middle character that is an empty character.

For example, in Fig. 7 (a), "success" (level 0) is bisected into two child strings "suc" and "ess" (omitting the middle character "c") (level 1), which are further bisected into smaller child strings "s" and "c", and "e" and "s" (again omitting the respective middle characters) (level 2). The next level bisections (level 3) are all empty, as indicated by the quotation marks. Fig. 7 (b) shows the bisection of a string that is not a common word.




(a)




(b)

Fig. 7 Bisection process of a word.
(a) A common word of length 7.
(b) An out-of-vocabulary word of length 8.

This bisecting process can be represented as a binary tree, where the root of the tree is the original string and the other nodes are the child strings. This binary tree can also be seen as a coarse-to-fine pyramid where each level focuses on smaller and smaller child strings.

For each node of the binary tree, a histogram of characters is calculated from the character string of that node, which is a histogram with n values (n being the size of the alphabet Σ), each value being the number of times the corresponding character occurs in the string. Fig. 8 shows the histograms of levels 0 to 2 for the example of Fig. 7 (a) (in this example, all child strings at level 3 are empty, so all level 3 histograms have zero values and are not shown in Fig. 8.

Note that the omission of the middle character when bisecting odd-length strings does not cause any loss of information. During decoding, the omitted middle characters can always be recovered by finding the difference between a histogram of a node and the sum of the two histograms of its left and right child node (and if there is no difference, then the omitted middle character is empty). For example, the central "c" in "success" can be found by subtracting the sum of two level 1 histograms (for "suc" and "ess") from the level 0 histogram (for "success") (see Fig. 8).

After the bisection is completed, all histograms for all nodes of the tree (including the zero histograms) are concatenated to form the attribute vector.

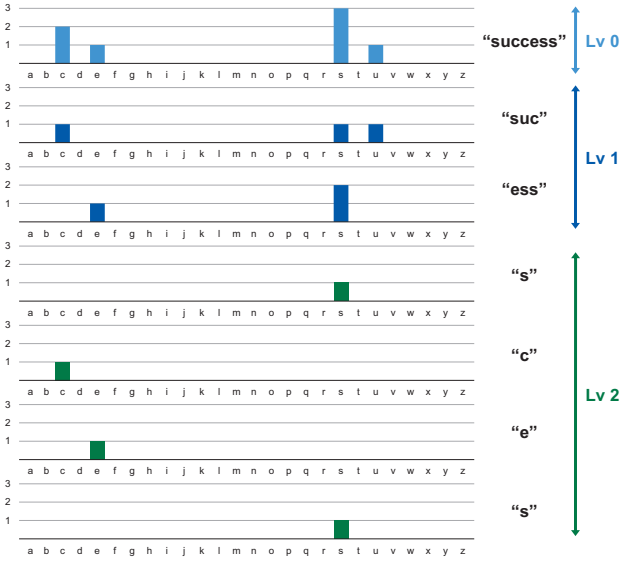



Fig. 8 Character histogram of each node in the binary bisection tree.

For an iPHOC encoding of k levels, there will be $(2^k-1)$ histograms; and with an alphabet of size n, the attribute vector's dimension will be $(2^k-1)*n$. To decode a CNN-predicted iPHOC attribute vector of dimension $(2^k-1)*n$, the vector is divided to obtain $(2^k-1)$ individual histograms each of size n, using the same order in which the histograms are concatenated in the encoding algorithm.

For each node of the decoding binary tree, the histograms of its left and right child nodes are subtracted from the histogram of the current node to obtain a difference histogram. The difference histogram is decoded to obtain a decoded character. If the maximum histogram value is greater than the threshold of confidence $\tau$, the decoded character is the character having the maximum histogram value; if the maximum histogram value is less than or equal to the threshold of confidence $\tau$, the decoded character is an empty character.

As a result, a decoded character (which may be an empty character) is generated for each node of the decoding binary tree. Fig. 9 illustrates the decoded characters organized in a "decoding binary tree", corresponding to the example of Fig. 7 (a).

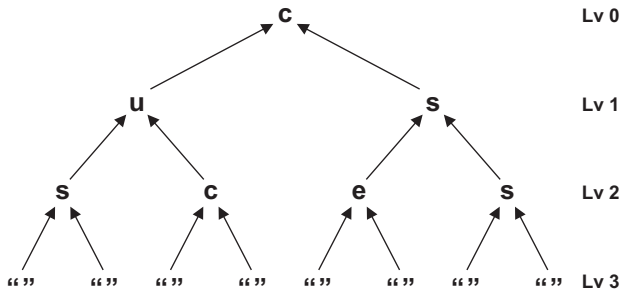



Fig. 9 A decoding binary tree.

The decoded characters for all the nodes of the decoding binary tree are concatenated together, based on an order that is the reverse of the recursive bisection used in the encoding algorithm, to obtain a character string that is the final decoding result.

As can be seen, the invertible label encoding and decoding method is a one-to-one mapping between character strings and a set of grid points of the Euclidean vector space. This method is simple and fast, as it does not require an optimization (nearest neighbor search) process for decoding. To the contrary, the SPOC [8] and PHOC [7] do not provide an invertible decoding method. In these methods, each words of the lexicon is mapped to a grid point, but not uniquely.

## 4 Experiments

To test the performance of the proposed iPHOC attributes with CNN, we report the word recognition accuracy on 7 test samples from writers whose handwritings have never been seen (Fig. 10). Each sample has 56 handwritten words that contain digital values and special characters (“$”, “%”, “,”, “.”, “-”, “(”, “)”). Some samples may have less words because miswritten words are not counted. The samples are scanned at 300 ppi.

Table 1 shows the word recognition accuracy on 7 samples. Comparing with a leading competitor, our model shows significant better accuracy, a 22% higher on average. Meanwhile, the accuracy of our model is on par with start-of-the-art handwritten digital sequence recognition, which uses a similar test dataset [12].

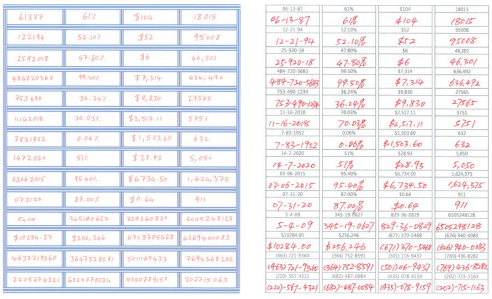


Fig. 10 Example of test samples.

Table 1 Evaluation results in comparison with a leading competitor.

| Sample (#) | Leading Competitor | Ours› | Difference |
|---|---|---|---|
| 1 | 48% | 73% | +25% |
| 2 | 77% | 93% | +16% |
| 3 | 59% | 86% | +27% |
| 4 | 56% | 75% | +19% |
| 5 | 77% | 89% | +12% |
| 6 | 63% | 89% | +26% |
| 7 | 50% | 79% | +29% |
| AVG | 61% | 83% | +22% |

## 5 Conclusions and Future Work

In this report, we proposed a handwriting recognition method that enables unconstrained transcription of handwritten word images. The method resolves the technical difficulty of transcribing a textual image that may contain arbitrary text. The method provides a deterministic procedure to encode and decode text embedding without using optimization or machine learning based methods as reported in the literature. This is possible because the label embedding uses an invertible coding scheme which is a one-to-one mapping between valid grid points of the Euclidean vector space and character strings. In the future, we would like to extend this method to recognizing a whole line of handwritten text. This would be desirable in the situation where word segmentation is hard.

## References


[1] Wang Yutao, Qin Tingting, Tian Ruixia, Yang Gang, “Recognition of license plate character based on wavelet transform and generalized regression neural network”, Control and Decision Conference (CCDC), 2012 24th Chinese, 1881-1885

[2] Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, Zhenbo Luo, Cheng-Lin Liu: Unsupervised Adaptation of Neural Networks for Chinese Handwriting Recognition. ICFHR 2016: 512-517

[3] Alex Graves, J. Schmidhuber, “Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks”, NIPS 2009

[4] Jaderberg, Max, et al. “Synthetic data and artificial neural networks for natural scene text recognition.” In *NIPS Deep Learning workshop* (2014).

[5] Jaderberg, Max, et al. “Reading text in the wild with convolutional neural networks.” *International Journal of Computer Vision* 116.1 (2016): 1-20.

[6] Poznanski, Arik, and Lior Wolf. “Cnn-n-gram for handwriting word recognition.” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[7] Almazán, Jon, et al. “Word spotting and recognition with embedded attributes.” *IEEE transactions on pattern analysis and machine intelligence* 36.12 (2014): 2552-2566.

[8] Rodriguez-Serrano, Jose A., Florent Perronnin, and France Meylan. “Label embedding for text recognition.” *Proceedings of the British Machine Vision Conference*. 2013.

[9] Wilkinson T, Brun A. Semantic and verbatim word spotting using deep neural networks. In Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on 2016 Oct 23 (pp. 307-312). IEEE.

[10] K. Simonyan et al., Very Deep Convolutional Networks For Large-Scale Image Recognition, ICLR 2015.

[11] He, Kaiming, et al. “Spatial pyramid pooling in deep convolutional networks for visual recognition.” *European conference on computer vision*. Springer, Cham, 2014.

[12] Wang, Qingqing, and Yue Lu. “A sequence labeling convolutional network and its application to handwritten string recognition.” *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. AAAI Press, 2017.