https://doi.org/10.2352/ISSN.2470-1173.2018.2.VIPC-176

Approach for Machine-Printed Arabic Character Recognition: the-state-of-the-art deep-learning method

Daegun Ko, Changhyung Lee, Donghyeop Han, Hyeongsu Ohk, Kimin Kang and Seongwook Han
S-Printing Solution at Hewlett Packard Seoul, 130, Samsung-ro, Yeongtong-gu, Suwon-si, Gyeonggi-do, South Korea
{dg.ko, changhyung.lee, donghyeop.han, ohk, kimin.kang, seongwook.han} @hp.com

Abstract

Optical character recognition (OCR) automatically recognizes texts in an image and converts them into machine codes such as ASCII or Unicode. Compared to many research studied on OCR for other languages, recognizing Arabic language is still a challenging problem due to character connection and segmentation issues. In this work, we propose a deep-learning framework of recognizing Arabic characters based on the multi-dimensional bi-direction long short-term memory (MD-BLSTM) with connectionist temporal classification (CTC). To train this framework, we generate over one-million Arabic text-line images dataset that contains Arabic digits, basic Arabic forms with isolated shape and connected forms. To compare the results, we also measure the performance of other OCR software such as Tesseract made by Hewlett-Packard and Google Inc. Tesseract version 3 and version 4 are used. Results show that deep-learning method outperforms the conventional methods in terms of recognition error rate, although the Tesseract_3.0 system was faster.

1. INTRODUCTION

Optical character recognition (OCR) [1] is to recognize the printed text or hand-written text in an image and convert into a machine code. OCR technologies were first launched on the market in the middle of 1950's. In 1960's to 1970's, OCR systems were able to recognize normal printed text and hand printed text. The new version of OCR, which appeared in the middle of 1970's, could recognized poor quality text and hand-written characters. In recently years, the OCR system is improved its performance and starts to be provided as software package. Nevertheless, the OCR cannot be compared with human reading capabilities. Therefore, in engineering aspects, the OCR capabilities need to be improved. In order to improve OCR technologies, analysis of the current state-of-the-art OCR method is very pertinent.

Arabic character recognition including hand-written has been many studies on the competition [17]. Despite decades of research on the engineering aspects, Arabic character recognition problem is still challenging issue in OCR filed. Since, Arabic letter has not only several shapes it is written connected to other letters in the word but also to appear connection between characters. By the same token, recognizing tasks as Arabic language recently prefer to apply segmentation-free method.

Regarding the performance of OCR, not only error rate but also processing speed has high priority. For example, the automatic document feeder (ADF) from the recently multi-function printer machine can scan 200 images per minute (ipm). Therefore, OCR processing speed is required to follow scanning speed.

Recognition or classification task exploits pattern recognition and machine learning technologies. According to Ko *et al.* [5, 6], they compared OCR methods by error rate and processing speed using convolutional neural network (CNN) [7-9] and Tesseract [3, 4]. In addition, they built their dataset and improved OCR capability using CNN. Plus, they proved that the deep-learning method is suitable for machine-printed character recognition. However, approaching CNN method, image segmentation step certainly is required, even though when Arabic letters are recognizing. Especially, Arabic letters is difficult to apply segmentation method for character recognition, due to connection between letters. Bushofa *et al.* [31] and Elnagar *et al.* [32] studied segmentation based Arabic letters recognizing which had been from each machine-printed and hand-written. To apply segmentation, they made over-segmentation rules that forced chopping thin area of text. In recently years, segmentation-free model has a lot of used to solve this problem, such as recurrent neural network and long short-term memory.

Additionally, Tesseract (version 3.0.1) case, it has been already compared with ABBYY FineReader commercial OCR packages for Polish historical printed documents [33]. From the result, both had exactly different characterization, however, when comparing results of both engines in test, there was not winner that would outperform the second engine in all test cases.

The goal of this paper is to get high performance printed Arabic language character recognition by the-state-of-the-art deep-learning method which composed multi-dimensional bi-directional long short-term memory (MD-BLSTM) [10, 12] with connectionist temporal classification (CTC) [11], and compare the Tesseract_4.0 with neural network [13] version, that is widely known open-source. To compare segmentation risk, we specially included Tesseract_3.0. For performance measurement, we use ISRI Analytic Tools for OCR Evaluation version 5.1 [15] and computes character error rate (CER), word error rate (WER), and processing speed for each method.

Generally, to compare performance, many studying has used to measure for fixed widely known database that are already extracted text-line or segmented one-characters. However, for this, we approached commercial aspects regarding to measure the performance. So, we used the 500 analog papers.

The rest of this paper is organized as follows, in the next section, we describe Arabic language and how we could generate -text-line images. In section 3 is a brief overview of character recognition, and we delineate MD-BLSTM with language model. In section 4, we report the experimental results on the OCR methods, and we also give an analysis of their performance comparison. Finally, in section 5, we derive a conclusion and suggest future work.

2. ARABIC LETTER

Arabic language has been used by more than 500 million people in about 25 countries. Arabic letter is the writing system of the Arabic language and widely used in many other languages including Arabic, Farsi, Urdu and etc.

The Arabic letter has 28 basic letters and multiple forms depending on its position in the word [18]. Some letter is written on an isolated shape when it is written alone. The other case is written in three shape when they is written connected to other letters in the word as begin, middle, end shape [Table 1] [Fig.1].

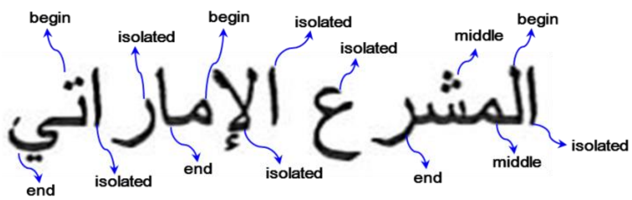


Figure 1. Consists of *Arabic sentence*

Table 1. Arabic language forms

Unicode	isolated	end	middle	begin
0x062A	ت	ـت	ـتـ	تـ
0x062B	ث	ـث	ـثـ	ثـ
0x062C	ج	ـج	ـجـ	جـ
0x062D	ح	ـح	ـحـ	حـ
0x062E	خ	ـخ	ـخـ	خـ

Additionally, Arabic language has features as follows:

The features of Arabic language

① Writing from right to left.

② Always expressing cursive writing types, such as
(a) generally written as “التشريع”
(b) written by all isolated (basic) form: ا ل ت ش ر ي ع

③ Character combination, such as الله, لاً, لا and etc.
: The letter لا is combined letter ل (0x0644) with letter ا (0x0627)

For Arabic character recognition, many researchers have been tried to apply segmentation rules to Arabic words [19]. In particular, Arabic character recognition system of segmentation-based methods not only mostly report only to segment perspective, but also cannot know exactly recognition engine performance due to segmentation error. For this reason, we mainly use and compare segmentation-free methods as MD-BLSTM that has designed by ourselves and Tesseract_4.0 with neural networks which is open-source. However, Tesseract_3.0 is segmentation-based model.

3. METHODS

In this section, we propose our OCR methods based on MD-BLSTM.

3.1 MD-BLSTM

For the past decade, recurrent neural networks (RNN) [22, 23] have emerged as an important area in artificial intelligence, machine learning and computer vision due to rapid development in digital image processing with huge and high-quality datasets.

Long short-term memory (LSTM) [22, 24] is the one of various kinds of RNN that solved vanishing problem. According to Shi *et al.* [25], they used one-dimensional LSTM to treat one-dimension from two-dimensional image. However, MD-LSTM brings clearly to improve recognition accuracy and was proved through several competitions.

In this paper, we generated 548,325 text-line images (around 9.62 GB) [Fig. 2] for training, and it took about 4 weeks on our environments. We obtained error rate every epoch on training step [Fig. 3] for 158 epochs. The sample images of AdobeArabic, Arial, Cour, Tahoma, Times Winsoftpro font types were used for training step, and Micross font was used by validation.

In addition to approach for deep-learning, the method is designed as shown in Fig. 4. Our approach is MD-BLSTM that has five-hidden layers, softmax output layer and CTC cost function [Table 2]. For experiments, we had given as initial learning rate, 0.0003; momentum, 0.9; and received feedback every-epoch. Finally, Stochastic Gradient Descent (SGD) was used by optimizer and updated on weights at neurons every-line processing. Plus, we used *tanh* activation function in sub-sampling layer to improve performance and to avoid over-fitting. For a reason to use this structure is usability that can improve from the processing time aspect. Even if segmentation-free model is used and also use language model as n-gram and word dictionary, our MD-BLSTM model is faster and more accuracy than Tesseract_4.0 with neural networks.

Table 2. The network parameters

Parameters	Value
input block	4 x 1
hidden block	4 x 2, 4 x 2
hidden size	2, 10, 50
sub-sample size	6, 20

Figure 2. *Generated text-line images for deep-learning method*

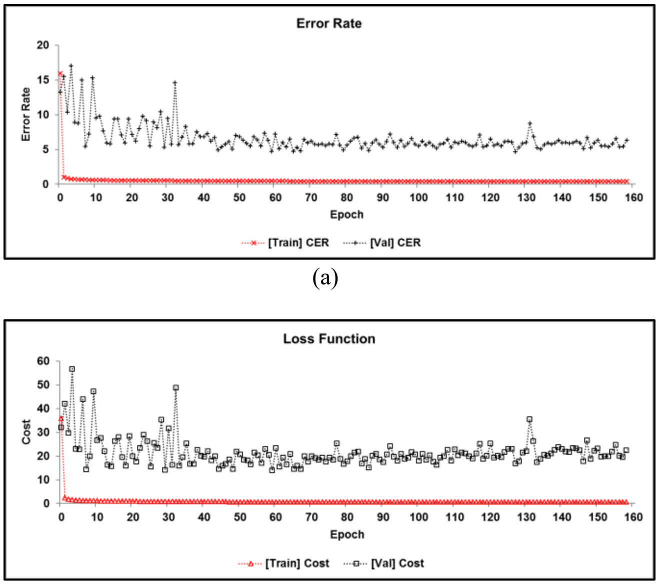
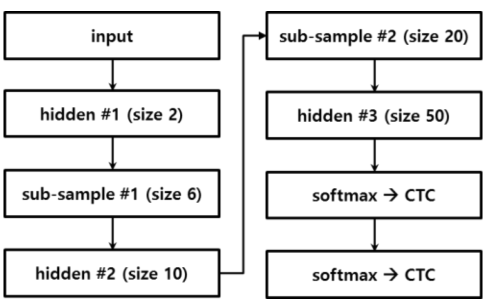
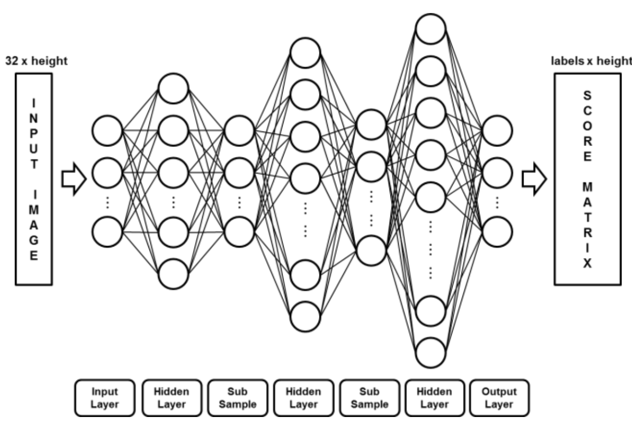


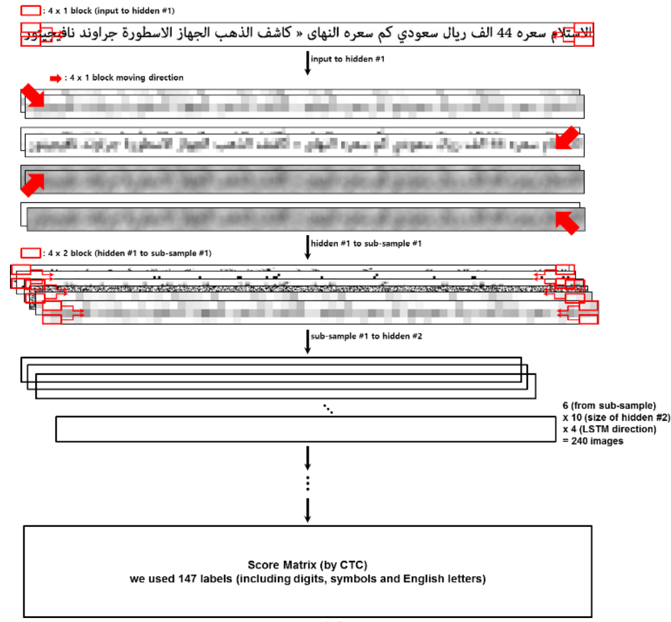
Figure 3. *Training and validation error on training step:*
(a) is shown accuracy rate and (b) is shown loss value (by CTC)
(CER: Character Error Rate, LER: Line Error Rate)



(a)



(b)



(c)

Figure 4. *The structure of MD-BLSTM method:*
(a) is a flow-chart, (b) is an architecture and (c) procedure of MD-BLSTM

Finally, to extract text-line in an image, we exploited text-line finding method from the Tesseract_3.0 (using Tesseract version 3.04 and Leptonica version 1.74.1 [16]), and the alteration of recognized result for several epochs are shown Fig. 9.

3.2 ARABIC TEXT-LINE DATASET

The performance of a learning-based system is primarily dependent on the quality of dataset. To train our MD-BLSTM method, we have built to sequence Arabic text-line images as show in Fig. 5.

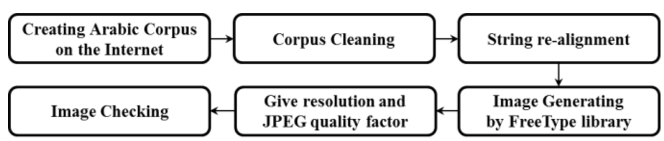


Figure 5. *The architecture of text-line generating*

To generate text-line image, firstly we have to search Arabic corpus from on the Internet. For this, we referred by KACST Arabic corpus [30]. Secondly clean up the corpus as Fig. 6. Lastly, text is adjusted the fixed number of character for a text-line and draw image using FreeType library [26] and inserting noise into image for JPEG quality factor. For building the dataset, we gathered Arabic corpus of around 10 GB files, and used 19 Arabic font files.

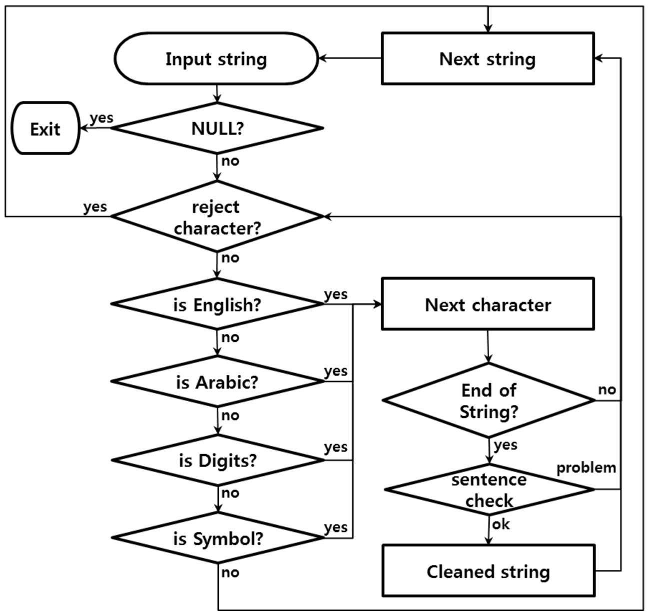


Figure 6. *A flow chart of clean up the corpus*

3.3 LANGUAGE MODEL AND FIXING PREDICT

To enhance recognition performance, language model (which is bi-gram with dictionary word list in our method case) should be applied, before fixing the final predicted string. In this paper, we used the SRILM [14] which is a collection of C++ libraries and freely available for statistical language model about speech or character recognition applications. For getting a language model, we began to process from the cleaned corpus at section 3.2 as following architecture [Fig. 7]. In addition, we had to extra work that exclude other language including digits and symbol, and remove duplicated sentence on the cleaned corpus.

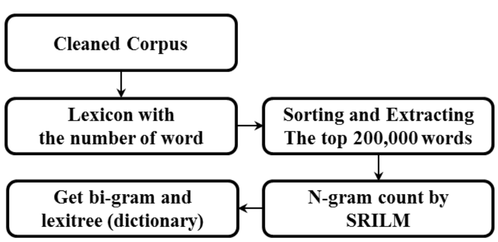
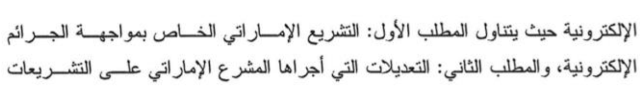
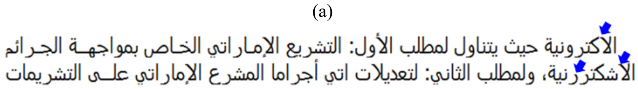


Figure 7. *The architecture of language model*

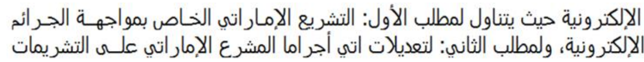
After making a score-matrix from the CTC, we give a bi-gram score each candidate of character using this language model data, and compute score of path using CTC probability score matrix, bi-gram probability and word existence or not in a dictionary. Finally, we choose the sentence of top path score as shown Fig. 8.



(a)



(b)



(c)

Figure 8. *An example of corrected sentence from language model: (a) original image (b) after best path (path 1) from the CTC (c) after adjust path (string) from the language model*

4. PERFORMANCE COMPARISON

MD-BLSTM method needs a training session to get optimized weighting values at each neuron. Thus, best result corresponding to weighting values brings from section 3.1 and doing comparison performance. MD-BLSTM, Tesseract_3.0 and Tesseract_4.0 are measured by PC-environments with processing speed.

4.1 TESSERACT

Tesseract is an open source for OCR that was developed by HP between 1984 and 1994. The engine was sent to UNLV for Annual Test of OCR Accuracy in 1995. In 2005, Tesseract was released as open-source. The simple procedure of Tesseract is as follows:

The procedure of Tesseract
① Binarization: to get a binary image from lightness non-uniformity in an image
② Connected Component (CC): to extract CC (such as labeling) and feature in the binary image
③ Line and word finding: outline are converted into blobs
④ Recognition: the result from step 3 was classified and the rest of the word recognition step applies only to non-fixed-pitch text
⑤ Producing the output text

In 2017, Tesseract version 4.0 added neural network which is long short-term memory (LSTM) and released. Thus, Tesseract_4.0 engine doesn't require segmentation rules any more. To differ from Tesseract_3.0 engine, Tesseract_4.0 has only to seek text-lines in an image instead of processing CC and extracting geometrical features.

4.2 PERFORMANCE MEASURING

To measure error rate, test samples has been made JPEG file up of fully Arabic plain texts as shown in Fig. 9. Test image consists of 500 image files, and it built to printing and scanning steps.

Test samples were printed by Samsung Smart Multi-Xpress 7 series from default option and scanned by also same device for 300 dpi resolution.

Figure 9. *The examples of test samples for accuracy measuring*

4.3 EXPERIMENTAL RESULTS

To measure error rate, we should use the Analytic Tool for OCR Evaluation that modified for us. For this, we added reject character lists as ‘~’, ‘`’, ‘!’, ‘@’, ‘#’, ‘$’, ‘%’, ‘^’, ‘&’, ‘*’, ‘(‘, ‘)’, -‘, ‘_’, ‘+’, ‘=’, ‘|’, ‘[‘, ‘]’, ‘{‘, ‘}’, ‘:’, ‘;’, ‘”’, ‘<’, ‘>’ and ‘/’.

The reading capability results are shown as Table 3. The best result is appeared when MD-BLSTM was used, considering both accuracy and speed. Even though processing time was not good than geometrical feature based Tesseract_3.0. In addition, a few of errors especially were occurred by text-line finding on Tesseract [Fig. 10]. For example, some letters exist around picture, and it has difference font size and colorful letters in the image.

Additionally, we conducted to compare processing speed about MD-BLSTM, Tesseract_3.0 and Tesseract_4.0 on our PC environments that consist of Windows Server 2012 64-bits, Intel Xeon CPU E5-2690v4 @ 2.60GHz and 256GB RAM.

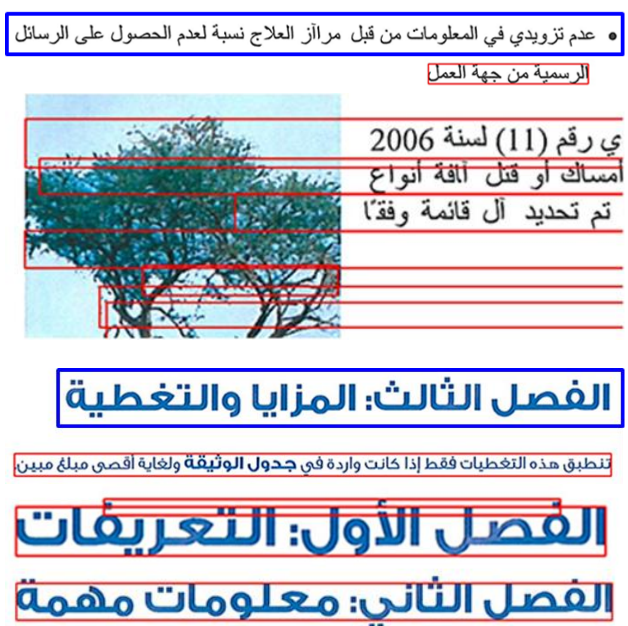


Figure 10. *Text-line finding miss in images (red box: extracted text-line, blue box: unfounded text-line)*

Table 3. Experimental results (Error Rate)

	CER	WER
MD-BLSTM	0.0988	0.2956
Tesseract_3.0	0.2106	0.4887
Tesseract_4.0	0.1029	0.3048

Table 4. Experimental Results (Processing Time)

	Total Time (sec)	Average Time (sec)
MD-BLSTM	3,025	6.05
Tesseract_3.0	2,235	4.50
Tesseract_4.0	9,050	18.1

The capability of recognition error rate and processing speed are shown as Table 3 and Table 4. Processing speed aspect, Tesseract_3.0 feature based processing model showed the best speed among the comparison models.

5. CONCLUSION

Our main approach is to recognize the Arabic language of the-state-of-the-art deep-learning method, and compare with widely known OCR methods. MD-BLSTM, Tesseract_3.0 and Tesseract_4.0 were included to accomplish the task. Additionally, we have built Arabic text-line images to conduct deep-learning method. Experimental results of CER and WER are utilized to compare the performance of the methods as shown in Fig. 12. As the result, MD-BLSTM deep-learning method showed outperforms in terms of error-rate. Especially, in the Tesseract open-source case, we recommend to use Tesseract_4.0 engine to recognize text in images, because it is segmentation-free model and is not segmentation-error.

The result of error-rate showed that printed Arabic character recognition is sufficiently difficult unlike already widely known other studies [17, 20, 21] through various language of hand-written recognition under the time limitation. Since, printed image has many problems such as text-line detecting issue, noise, screen, skew, slant, variance size and font, brightness of color and etc. Therefore, printed character recognition is still challenging parts and need to study about some of languages. Finally, we will need to be improved OCR algorithms about other languages like Hebrew, Farsi and Greek, and also, we will have to overcome text-line detecting problem by [27-29]. Moreover, we should be changing experimental method as n-fold cross validation.

References

[1] R. Mither, S. Indalkar and N. Divekar, “Optical Character Recognition”, *International Journal of Recent Technology & Engineering*, IJRTE, 2013.

[2] S. V. Rice and T. A. Nartker, “The ISRI Analytic Tools for OCR Evaluation”, *Information Science Research Institute*, 1996.

[3] R. Smith, “Tesseract OCR Engine”, *Google Inc.*, OSCON, 2007

[4] R. Smith, "An Overview of the Tesseract OCR Engine", *International Conference on Document Analysis and Recognition*, IEEE, 2007.

[5] D. G. Ko, S. H. Song, K. M. Kang, S. W. Han and J. H. Yi, "Optical Character Recognition Performance Comparison of Convolutional Neural Networks and Tesseract", *The 31st International Technical Conference on Circuits/Systems, Computers and Communications Technical Program*, ITC/CSCC: pp. 871-874, 2016.

[6] D. G. Ko, S. H. Song, K. M. Kang and S. W. Han, "Convolutional Neural Networks for Character-Level Classification", *A publication of the Institute of Electronics and Information Engineers Transactions on Smart Processing and Computing*, IEIE SPC: vol.6 no.1 pp.53-59, 2017.

[7] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", *in Neural Information Processing System*, NIPS, 2012.

[8] D. Ciresan, U. Meier and J. Schmidhuber, "Multi-column Deep Neural Networks for Image Classification", *in Proceeding of Computer Vision and Pattern Recognition*, IEEE, 2012.

[9] Y. LeCun, L. Bottou, Y. Bengoi and P. Haffner, "Gradient based learning applied to document recognition", *in Proceeding of the IEEE*, IEEE, 1998.

[10] A. Graves, S. Fernandez and J. Schmidhuber, "Multi-Dimensional Recurrent Neural Networks", *International Conference on Artificial Neural Networks*, ICANN: pp. 549-558, 2007.

[11] A. Graves, S. Fernandez, F. J. Gomez and J. Schmidhuber, "Connectionist temporal classification: Labeling unsegmented sequence data with recurrent neural nets", *in Proceedings of the 23rd international conference on machine learning*, ICML: pp. 369-376, 2006.

[12] A. Graves and N. Jaitly, "Toward end-to-end speech recognition with recurrent neural networks", *in Proceeding of the 31st International conference on machine learning*, ICML: pp. 1764-1772, 2014.

[13] Tesseract 4.0, https://github.com/tesseract-ocr/tesseract/wiki/4.0-with-LSTM, available.

[14] A. Stolcke, "SRILM – An Extensible Language Modeling Toolkit", *Speech Technology and Research Laboratory*, SRI International.

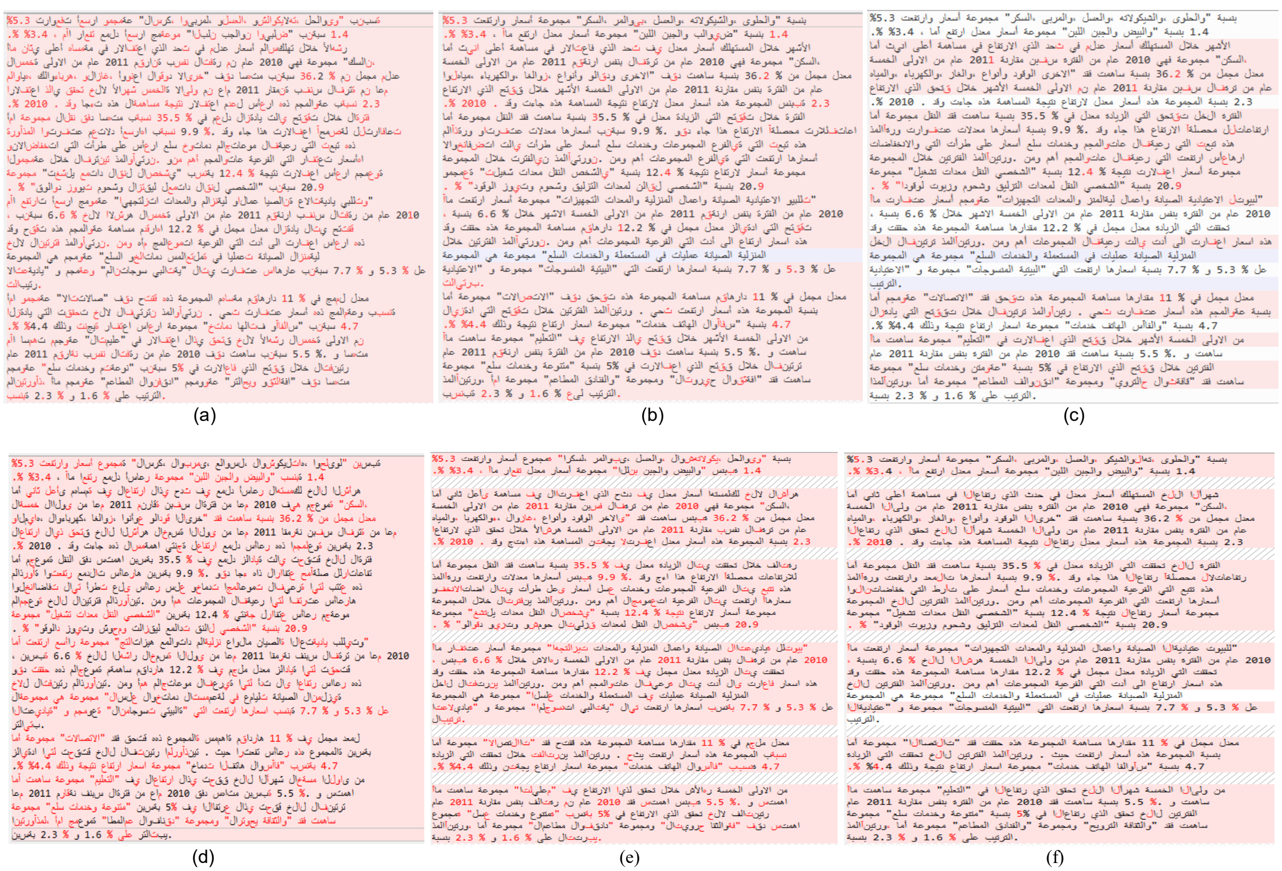
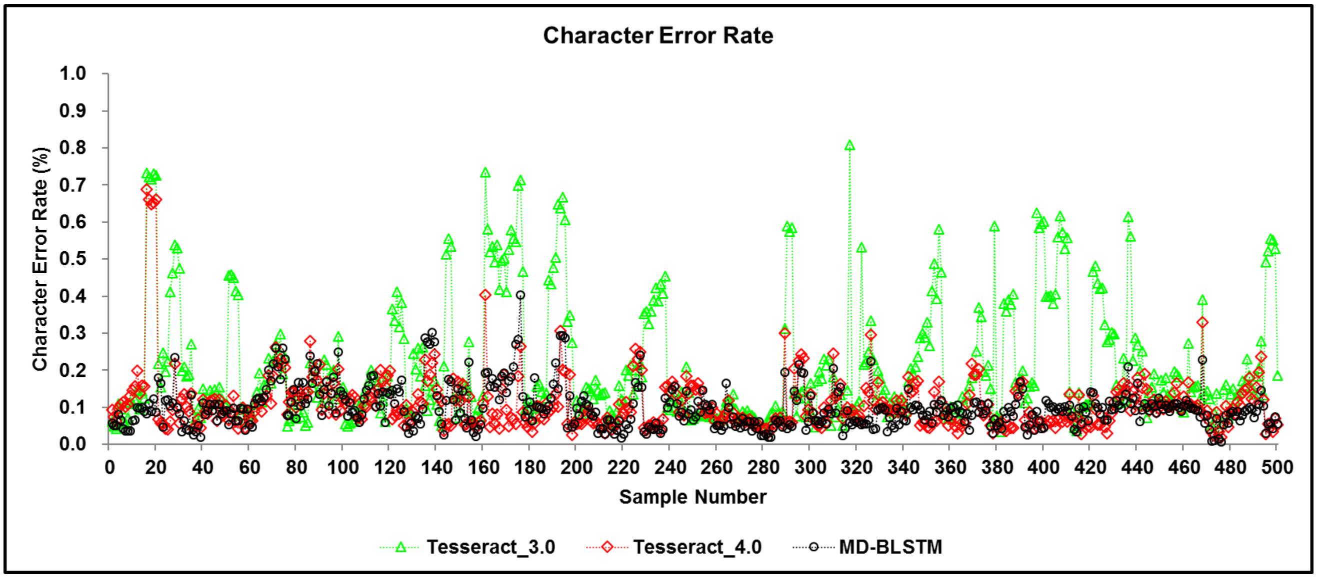
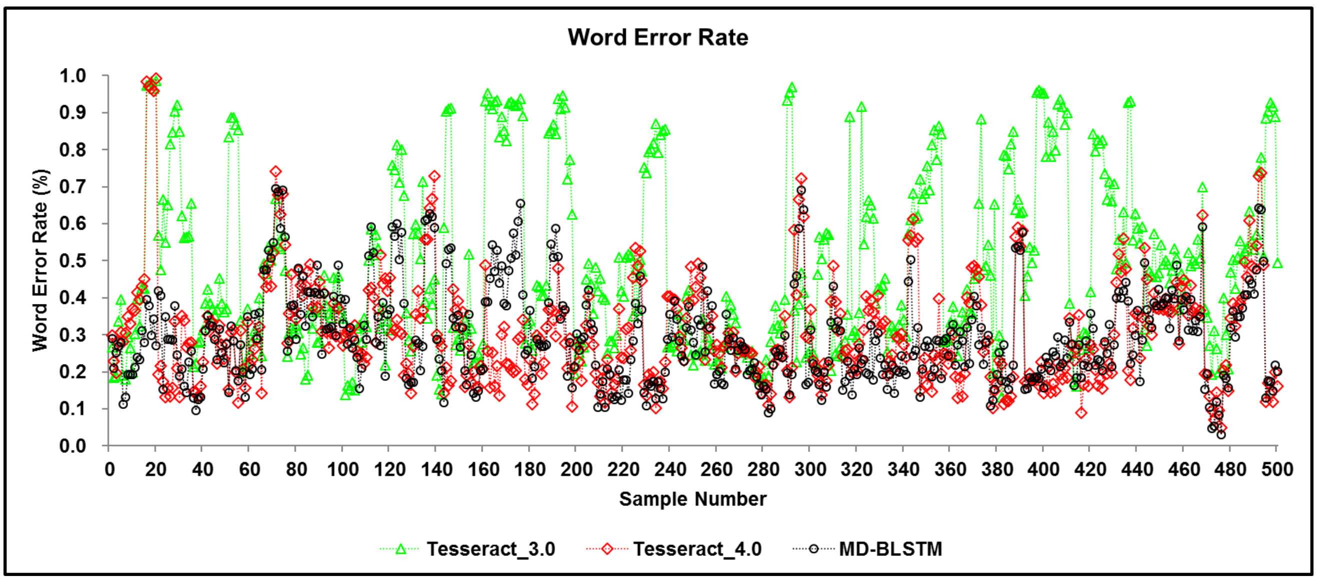


Figure 11. *Accuracy alteration is shown by MD-BLSTM method (black letters are correct and red letters are incorrect, white text-lines are perfect line): (a) is the result of prediction using 12 epoch's weighting values, (b) is 44 epoch, (c) is 148 epoch, (d) is a result of the commercial OCR S/W, (e) is a result of the Tesseract3 and (f) is a result of the Tesseract4*



(a)



(b)

Figure 12. *Recognition error tendency are shown about each sample: (a) CER (b) WER*

[15] S. V. rice and A. Nartker, "The ISRI Analytic Tools for OCR Evaluation Version 5.1", Information Science Research Institute: TR-96-02, 1996.

[16] Leptonica, http://www.leptonica.com/, available.

[17] S. Mozaffari and H. Soltanizadeh, "ICDAR 2009 Handwritten Farsi/Arabic Character Recognition Competition", *in 10th International conference on Document Analysis and Recognition*, IEEE, 2009.

[18] Arabic Unicode, https://en.wikipedia.org/wiki/Arabic_script_in_Unicode, *Wikipedia*, available.

[19] G. Abandah and F. Jamour, "Recognizing handwritten Arabic script through efficient skeleton-based grapheme segmentation algorithm", *in Proceeding International conference Intelligent Systems Design and Applications*, pp. 977-982, 2010.

[20] E. Grosicki and H. El-Abed, "ICDAR 2011 – French Handwriting Recognition Competition", *in 2011 International Conference on Document Analysis and Recognition*, IEEE, 2011.

[21] V. Margner and H. EI-Abed, "Arabic Handwriting Recognition Competition", *in Ninth International Conference on Document Analysis and Recognition*, IEEE, 2007.

[22] F. Gers, "Long Short-Term Memory in Recurrent Neural Networks", PhD Thesis, 2001.

[23] A. Graves, “Sequence Transduction with Recurrent Neural networks”, *in the International Conference of Machine Learning*, ICML, 2012.

[24] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory”, *in Journal of Neural Computation*, Vol. 9 pp. 1735-1780, 1997.

[25] B. Shi, X. Bai and C. Yao, “An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, 2015.

[26] FreeType, https://www.freetype.org/download.html, available.

[27] Y. Boulid, A. Souhar and M. Y. Elkettani, “Arabic handwritten text line extraction using connected component analysis from a multi agent perspective”, *15th International Conference on Intelligent Systems Design and Applications*, IEEE, 2015.

[28] A. Zahour, B. Taconet, P. Mercy and S. Ramdane, “Arabic hand-written text-line extraction”, *6th International Conference on Document Analysis and Recognition*, IEEE, 2001.

[29] Z. Shi, S. Setlur and V. Govindaraju, “A Steerable Directional Local Profile Technique for Extraction of Handwritten Arabic Text Lines”, *10th International Conference on Document Analysis and Recognition*, IEEE, 2009.

[30] A. O. Al-Thubaity, “A 700M+ Arabic corpus: KACST Arabic corpus design and construction”, *Language Resources and Evaluation*, Vol. 49: pp. 721-751, Springer Netherlands, 2015.

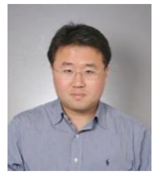
[31] B. M. F. Bushofa and M. Spann, “Segmentation and Recognition of Printed Arabic Characters”, *British Machine Vision Conference*, BMVC, 1995.

[32] A. Elnagar and R. Bentrcia, “A Recognition-Based Approach to Segmentating Arabic Handwritten Text”, *Journal of Intelligent Learning Systems and Applications*, 7: pp. 93-103, SciRes, 2015.

[33] M. Helinski, M. Kmieciak and T. Parkola, “Report on the comparison of Tesseract and ABBYY FineReader OCR Engines”, *Improving Access to Text*, IMPACT.

Author Biography

Daegun Ko *received his BSc in Electronic Engineering and Computer Engineering from Yeongnam University, South Korea, in 2009, and he was a member in Samsung Electronics Software Membership from 2006 to 2009. He received the MSc from the Department of Digital Media and Communications Engineering at Sungkyunkwan University, South Korea, in 2016. He had worked a researcher in the Samsung Electronics Co.,Ltd., South Korea from 2009 to 2017. He has been currently working a researcher at Hewlett Packard S-Printing Solution Co.,Ltd., South Korea. His research interests include image processing, pattern recognition, deep-learning and computer vision systems.*



Changhyung Lee *received his BSc and MSc degrees from Seoul National University, Seoul, South Korea, both in Electrical Engineering, in 1997 and 2000, respectively. He received his PhD degree in Electrical and Computer Engineering from Purdue University, West Lafayette, IN in 2008. In 2009, he joined Samsung Electronics, Co.,Ltd., Suwon, South Korea, where he worked as a principal engineer in the Digital Printing Division. From 2017, he is working as an engineer at the Office Printing Solutions in Hewlett-Packard Korea. His research interests include image rendering, imagine enhancement, and machine learning.*

Donghyeop Han *received BSc in Yonsei University 1996. MSc in 1998 and PhD in Texas A&M University 2009. He had worked at Samsung Electronics Co.,Ltd, South Korea from 1998 to 2017, and has worked as a researcher at Hewlett Packard S-Printing Solution Co.,Ltd. South Korea. His research interests include image processing and optical character recognition.*

Hyeongsu Ohk *had worked at Samsung Electronics Co.,Ltd., South Korea from 2009 to 2017, and he has worked a researcher at Hewlett Packard S-Printing Solution Co.,Ltd., South Korea. His research interests include image processing, document compression and optical character recognition.*

Kimin Kang *received his PhD in Electronical Engineering from Inha University, South Korea, in 2001. He joined Samsung Electronics Co.,Ltd., in 2001, and researched and developed the algorithms and the pipelines related to image enhancement and quantitative quality diagnosis with vision system. Now, he develops documents workflow solutions embedded in the copier machines based on scene analysis and optical character recognition.*

Seongwook Han *received his BSc in Electronics Engineering from Yonsei University, Seoul, South Korea, in 2000, and the MSc in Electronical and Electronics Engineering from Yonsei University, Seoul, South Korea, in 2002. From 2002 to 2004, he was a research engineer at on Timetek Inc., Seoul, where he worked on video compression, video transmission, and pre/post processing for digital broadcasting systems. In 2009, he received his PhD in Electrical Engineering from Purdue University, West Lafayette, IN. Since January 2009, he has been with Samsung Electronics Co.,Ltd., Suwon, South Korea, developing algorithms for electronic imaging systems. His research interests include electronic imaging system, color processing, video coding, image/video analysis and image enhancement.*