myOCR: Optical Character Recognition for Myanmar language with Post-OCR Error Correction

Thura Aung*
LU Lab., Myanmar
Software Engineering
SIIE, KMITL.
Bangkok, Thailand.
66011606@kmitl.ac.th

Ye Kyaw Thu
LU Lab., Myanmar
LST, AINRU,
NECTEC, NSTDA.
Pathum Thani, Thailand.
yekyaw.thu@nectec.or.th

Myat Noe Oo
Robotics and AI
Engineering
SIIE, KMITL.
Bangkok, Thailand.
66011066@kmitl.ac.th

***Abstract*—This paper presents the Myanmar Optical Character Recognition (OCR), named myOCR. It utilizes a synthetic text image dataset with 14 different font styles that contains 25,790 text images. The system includes Convolutional Neural Networks (CNN) for feature extraction, Bidirectional Long-Short Term Memory (BiLSTM) networks for sequence modeling, and Connectionist Temporal Classification (CTC) for decoding, evaluated across various iterations (3,000, 6,000, 9,000) and hidden states (64, 128, 256). Statistical Post-OCR correction methods involve N(3,4,5)-grams and edit distances with the Symmetric Delete Spelling correction algorithm (SymSpell). For Neural Machine Translation-based correction, BiLSTM and Transformer models are employed, while the mT5-base and mBART-50 models are used for LLM-based correction. The best base (optical) model is the model with 9,000 iterations that achieved a CHRF^{++} score of over 97.90 and a Word Error Rate (WER) of 9.18%. Transformer correction improved its CHRF^{++} to 99.31 and reduced the WER to 0.66%.**

***Index Terms*—Optical Character Recognition, Myanmar language, Synthetic data, Post-OCR, low-resource**

I. Introduction

Optical Character Recognition (OCR) technology makes a big impact on modern society in digitization by converting physical manuscripts to the digital environment, i.e., from not machine readable to machine-readable. Extracting and analyzing information contained in physical documents with different page layouts and graphical information lies in the accuracy of OCR [1].

For the optical models, deep learning approaches such as Convolutional Neural Networks (CNN) ([2], [3], [4]) and bidirectional Recurrent Neural Network (BiRNN) [5] improved results compared to traditional machine learning algorithms. CRNN [6], which is a CNN and RNN variant, Long Short-term Memory (LSTM) network hybrid architecture [8] also shows good results for image-based sequence recognition tasks. It was also studied that CRNN-based architectures could improve recognition using connectionist temporal classification (CTC) [9].

Many scenarios and font variations challenge the recognition performances of the OCR system and lead to Post-OCR errors. Therefore, we plan to use N-gram similarity-based correction [10] with a monolingual corpus and edit distance-based correction using the SymSpell algorithm [11] as statistical Post-OCR correction methods. Sequence-to-Sequence (S2S) Neural Machine Translation (NMT) models based on BiLSTM [12] and Transformer [13], and Large Language Models (LLM) such as mT5 [14] and mBART [15] were also applied for the correction of Post-OCR errors.

II. Image Dataset Preparation

NLP researchers in Myanmar are encountering numerous challenges due to the scarcity of free resources. With no open-source dataset available for Myanmar OCR, we generated text images using monolingual text data rendering with mytext2pic.sh[1] bash program.

TABLE I: myOCR IMAGE DATASET PARTITIONING FOR THE EXPERIMENTS

	Images	**Words**	**Characters**
Train	18,052	30,008	1,884,829
Valid	5,802	9,475	599,906
Test	1,936	15,975	201,764
Total	25,790	55,458	2,686,499

Table II shows the synthetic text images with different fonts, generated from the text နည်းပညာကဏ္ဍ ("Technology sector" in English).

The monolingual text data used is a part of myPOS corpus version 3.0 [16], which was originally intended for Myanmar Part-Of-Speech tagging. The Myanmar text images were generated with 14 different font styles to mitigate data bias. Also for statistical Post-OCR correction, myPOS is used to build N-gram dictionaries. Figure 1 describes the pie chart visualization, which shows

*Majority of the work done during the internship at the Language Understanding Lab., Myanmar.

979-8-3315-0991-0/24/$31.00 ©2024 IEEE

[1]https://github.com/ye-kyaw-thu/tools/blob/master/bash/mytext2pic.sh

TABLE II: SYNTHETIC IMAGES OF နည်းပညာကဏ္ဍ (“TECHNOLOGY SECTOR” IN ENGLISH) WITH DIFFERENT FONTS

Font Name	Text Image	Font Name	Text Image
Burmese Handwriting Style 04	နည်းပညာကဏ္ဍ	Kamjing	နည်းပညာကဏ္ဍ
Myanmar Ayar3	နည်းပညာကဏ္ဍ	Z01-UMoe	နည်းပညာကဏ္ဍ
Z03-Press	နည်းပညာကဏ္ဍ	Z09-LatYaySat	နည်းပညာကဏ္ဍ
Masterpiece Spring Revolution	နည်းပညာကဏ္ဍ	Masterpiece Uni Type	နည်းပညာကဏ္ဍ
Myanmar Chatulight	နည်းပညာကဏ္ဍ	Myanmar Phiskel	နည်းပညာကဏ္ဍ
Myanmar Sanpya	နည်းပညာကဏ္ဍ	Myanmar Yin Mar	နည်းပညာကဏ္ဍ
NKSSmart3	နည်းပညာကဏ္ဍ	Pyidaungsu	နည်းပညာကဏ္ဍ

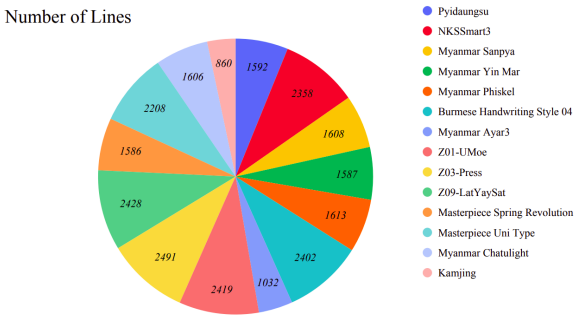

Fig. 1: Number of lines of each font used in the synthetic text images

the number of lines of each font used in the synthetic text images. Table I shows the dataset splitting for the experiments.

III. Methodologies

A. Optical Models

The optical models have **three** main stages - feature extraction, feature sequence modeling, and sequence decoding. Feature Extraction is done on an input grayscale image by mapping the image to a feature vector focusing on character recognition and neglecting other irrelevant information such as color, size, and background using the CNN model with VGG architecture [17]. The convolution operation in the CNN model takes place by applying some filters on the input image, where the filter values are initialized randomly to extract the visual features of the characters. BiLSTM is used for feature sequence modeling [7]. The BiLSTM captures the character sequence information from left to right and then from right to left, encoding feature sequences. CTC is used for the sequence decoding or prediction stage [7].

B. Post-OCR Correction

Post-OCR error correction experiments were conducted to reduce error rates in OCR results from optical models. We explored both statistical and neural methods, including character-level N-gram similarity and SymSpell edit distance approaches for detection and correction, as well as NMT- and LLM-based S2S models for neural correction.

1) Statistical Approaches: The N-gram similarity-based spelling correction [10] computes the similarity between a misspelled word w and a candidate correction c based on their N-gram representations. We used character-level N-gram for Post-OCR correction. In character N-gram representation, a word w is broken down into its constituent N-grams, which are subsequences of n characters.

Symmetric Delete spelling correction (SymSpell) algorithm is useful for spelling correction [23]. SymSpell simplifies the process of generating edit candidates and looking up the dictionary for a given Damerau-Levenshtein distance. Instead of transpositions, replacements, and insertions of the input phrase, it transforms them into deletions of the dictionary term. The algorithm's speed comes from the low-cost delete-only edit candidate generation and pre-calculation. In terms of computational complexity, the SymSpell algorithm can operate in constant time (O(1) time) due to the use of a Hash table-based index, which typically offers an average search time complexity of O(1).

2) Neural Machine Translation Approaches: The BiLSTM (Bidirectional Long Short-Term Memory) network [18] was employed to capture long-range dependencies in both directions within the OCR-generated text. This approach benefits from processing input sequences both forward and backward, making it well-suited for noisy and

inconsistent OCR output. The BiLSTM network learns to map sequences of OCR text to their corrected form by considering contextual relationships within the text, enhancing the error detection and correction process.

The Transformer architecture [19], known for its attention mechanism, addresses the limitations of recurrent networks by focusing on all words of the input sequence simultaneously, rather than processing them sequentially. For OCR correction, the self-attention mechanism helps the model focus on relevant parts of the text, identifying patterns of errors and suggesting corrections based on a global context. This global context-awareness makes the Transformer particularly powerful in scenarios where OCR errors exhibit complex patterns that are difficult to capture with traditional sequence models.

3) LLM Approaches: LLMs excel in handling complex, non-localized patterns. The mT5 (Multilingual T5) model [20], based on the T5 architecture, is specifically designed for multilingual tasks. mT5 leverages a pre-trained sequence-to-sequence transformer that is fine-tuned for the specific task of OCR correction. We used mT5-base which is a variant of mT5. The mBART (Multilingual BART) model [21], based on the BART architecture, is specifically designed for multilingual denoising and text generation tasks. mBART leverages a pre-trained sequence-to-sequence transformer with a denoising autoencoder objective, making it particularly well-suited for tasks like OCR error correction, where noisy input data is common. We utilized the mBART-50 model, a variant fine-tuned for multiple languages.

IV. Experiments

A. Optical Model Training

Our myOCR dataset is used to train optical models with 3,000, 6,000, and 9,000 iterations and 64, 128, and 256 hidden states respectively. Then the model performances were tested on the open test data. We trained models with the VGG feature extractor, both with and without BiLSTM feature sequence modeling and CTC decoding. For the experiments with 3,000 iterations, our models cannot decode the extracted feature maps but we found out that iterating 6,000 iterations and 9,000 iterations can decode the extracted feature maps with some OCR errors.

B. OCR Error Analysis

The structure of the Myanmar language follows a hierarchical formation where characters combine to form grapheme clusters, which in turn form syllables, and these syllables come together to create words, as illustrated in Figure 2. We have done error analysis in the most basic unit which is character level using the SCLITE (score speech recognition system output) program from the NIST scoring toolkit (Version 2.4.11) [24]. It is used to align the OCR result hypothesis text with ground-truth text. This program shows the recognition rate at the sequence level and word level and also gives the confusion pairs.

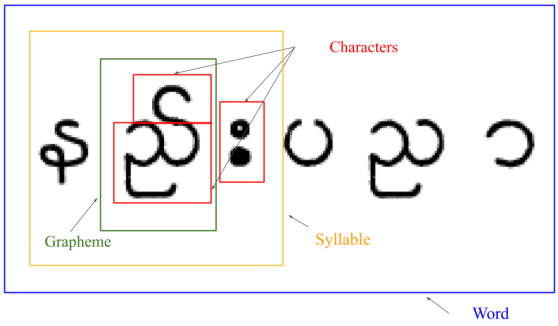

Fig. 2: Grapheme Composition of Myanmar word နည်းပညာ ("Technology" in English) in which Characters in Red, Grapheme-cluster in Green, Syllable in Yellow, and Word in Blue boxes

TABLE III: SAMPLES FOR EACH ERROR TYPE RANDOMLY EXTRACTED FROM THE OPTICAL MODEL RESULTS

Error Type	Error and Ground-truth
Selection	လ ု ံ ဦ ြ ု ံ ရ ေ ား လ ု ံ ခ ြ ု ံ ရ ေ ား
Insertion	က ြ ည ု ် က ြ ည ် ု
Deletion	အ က ိ ု ား မ ပ ေ ား အ က ျ ိ ု ား မ ပ ေ ား

For error analysis, the SCLITE scoring method first aligns the hypothesis and ground-truth text and then calculates a minimum Levenshtein distance. For each type of error type, we extracted sample error points randomly from OCR model outputs as shown in Table III, where the first line is OCR output and the second line is the Ground-truth result.

TABLE IV: THE TOP 10 VISUAL ERROR CONFUSION PAIRS FROM OCR MODELS (CORRECT → OCR ERROR)

Frequency	Confusion Pair
361	သ → လ
236	ဝ → ဝ
152	ီ → ဦ
136	ဘ → တ
112	ီ → ိ
107	ထ → ဆ
104	ဆ → ထ
94	ဝ → ထ
85	ဗျ → ျ
81	? → ရ

[25] explained that in Unicode systems, Myanmar vowels are typed and stored after the consonant due to their pronunciation but are rendered in the correct order.

For example, in Myanmar language vowel sign E ေ is typed and stored after the consonant as in နေ (Sun in English) = န (Consonant Na) + ေ (Vowel E), even though it is rendered in front of the consonant. Apart from visual errors as shown in Table IV, in our optical models, the sequence decoding stage made such kind of error that Vowel E + Consonants, mostly in the models without feature sequence labeling.

To reduce any type of Post-OCR errors, the decoded output text sequences from the optical models were fed into N-gram similarity score calculation and SymSpell algorithm for statistical and into the S2S models for the NMT and LLM correction approaches.

C. Post-OCR Experiments

The N-gram was examined with N values of 3, 4, and 5 because of the nature of the Myanmar language. SymSpell was used with different edit distances 2, 3, and 4. Despite no significant differences exist, we had the best result with character trigram (N=3) for N-gram and edit distance 4 for SymSpell approaches.

The OCR errors generated by the optical models from inferencing on the test data were merged and segmented into syllables using sylbreak[2] then compiled into a parallel corpus, which was then used to train the S2S models for both NMT and LLM approaches.

For the NMT models, the maximum syllable-segmented sequence length for both source and target sentences is 100 tokens. The batch size is set to 64 with a dropout rate of 0.5 for BiLSTM and token-based batching batch size 8 with a dropout rate of 0.1 and normalization for the Transformer model. To prevent overfitting, early stopping is set to 10 validations. For the BiLSTM-based S2S models, the number of hidden layers is 2 with a hidden size of 256 for each model. For Transformer-based S2S models, both the encoder and decoder have a text vector size of 128 and a hidden size of 128. Both the encoder and the decoder comprise 4 layers each. The transformer feed-forward dimension is 512, and attention is distributed across 4 heads. The learning rate for the BiLSTM model is 0.01 and for the Transformer model is 0.1 respectively.

For the LLM approach, we used mT5-base and mBART-50. Both models have a maximum sequence length of 96 tokens. mT5-base was trained for 10 epochs with a batch size of 4, while mBART-50 was trained for 5 epochs with a batch size of 1. Evaluations were conducted every 1,000 steps for both models. Mixed precision training (FP16) was enabled to handle memory limitations, with output directories being overwritten and input data reprocessed at each run. Checkpoints were not saved during evaluation to optimize speed and resource usage.

D. System Configuration

The optical models were trained on a system with AMD Ryzen 9 3900X, 12-core, 24 threads processor, NVIDIA Quadro RTX 4000, 8 GB VRAM graphics card, and 32 GB RAM and the Post-OCR correction experiments were conducted on a server equipped with an Intel Core i9-14900KF processor (64-bit, 5701 MHz), 64 GB of system memory, and an NVIDIA RTX A6000 GPU featuring 48 GB of GDDR6 VRAM and a power consumption of 300W.

Pytorch[3] was used to train the optical models. N-gram correction was implemented using ngram[4] python library. We used the official implementation of SymSpell[5] algorithm for SymSpell error correction. OpenNMT[6] framework was used to train Post-OCR NMT models and the simpletransformers[7] library was used to fine-tune the LLMs. We used Jiwer[8] for calculating Word Error Rate (WER) and chrF^{++} tool [22] for Character N-gram F-score (chrF^{++}).

E. Evaluation

1) Word Error Rate (WER): Word Error Rate (WER) is the percentage of words, which are to be inserted, deleted, or substituted to convert the hypothesis-predicted text of the source to its actual (ground-truth) text. WER can be computed using $WER = \frac{(I+D+S)\times 100}{N}$ where I, D, S, and C refer to the number of insertions, deletions, substitutions, and correct words respectively. N refers to the number of words in the reference translation (target), which can be calculated as $N = S + D + C$.

2) Character N-gram F-score (chrF^{++}): chrF^{++} has calculated the F-score of the predicted output text sequence with respect to the reference sequence (target) based on character N-gram. The general formula for calculating the chrF^{++} score $= (1+\beta^2)(\frac{chrp.chrr}{\beta.chrp+chrr})$ where chrp and chrr stand for character N-gram precision and recall arithmetically averaged over all N-grams.

F. Experimental Results

Table V and VI show the evaluation results for our base model (optical only, without Post-OCR correction), and models with Post-OCR processing. Base in the Table V and VI means optical models without any Post-OCR correction and N/A means there was no pre-trained optical model. Bolded results are the best from comparisons in each column for each optical model type (None, +BiLSTM). Speaking about evaluation, it has been found that training with more iterations and more hidden states can improve the model performance of the optical models.

1) Statistical Post-OCR correction: Statistical Post-OCR correction with both N-gram and SymSpell can reduce WER. With feature sequence modeling included, WER were reduced nearly fivefold. N-gram Post-OCR correction is slightly better than the SymSpell algorithm mostly.

[2]https://github.com/ye-kyaw-thu/sylbreak

[3]https://pytorch.org/

[4]https://pypi.org/project/ngram/

[5]https://github.com/wolfgarbe/SymSpell

[6]https://opennmt.net/OpenNMT-py

[7]https://simpletransformers.ai/

[8]https://pypi.org/project/jiwer/

TABLE V: WORD ERROR RATE (WER) SCORES FOR EACH MODEL. THE LOWER THE WER, THE BETTER THE MODEL IS.

	Iteration	3,000			6,000			9,000		
	Hidden State	64	128	256	64	128	256	64	128	256
None	Base	N/A	N/A	N/A	30.20	23.92	35.47	23.46	18.94	19.15
	+N-gram	N/A	N/A	N/A	14.65	10.03	17.25	9.42	7.16	7.40
	+SymSpell	N/A	N/A	N/A	13.98	9.81	15.95	9.49	7.53	7.65
	+BiLSTM-S2S	N/A	N/A	N/A	15.07	10.83	19.60	10.20	7.54	7.54
	+Transformer-S2S	N/A	N/A	N/A	**9.89**	**6.37**	12.95	**6.12**	**3.57**	**3.24**
	+mT5-base	N/A	N/A	N/A	32.19	30.98	33.60	30.74	30.09	30.57
	+mBART-50	N/A	N/A	N/A	11.24	11.01	**11.69**	10.94	10.57	10.63
+BiLSTM	Base	15.87	17.32	12.30	10.74	11.05	9.69	10.04	10.11	9.18
	+N-gram	5.25	6.43	2.98	2.33	2.58	1.79	1.98	1.99	1.59
	+SymSpell	5.26	6.06	3.17	2.40	2.54	1.83	2.04	1.99	1.53
	+BiLSTM-S2S	7.05	6.56	5.31	4.58	4.27	3.83	4.10	4.05	3.71
	+Transformer-S2S	**3.65**	**3.35**	**1.69**	**1.32**	**1.36**	**0.83**	**1.05**	**1.00**	**0.66**
	+mT5-base	30.41	30.68	29.73	29.56	29.75	29.51	29.69	29.61	29.54
	+mBART-50	10.73	10.87	10.64	10.54	10.61	10.62	10.60	10.61	10.64

TABLE VI: CHARACTER N-GRAM F-SCORES (CHRF^{++}) FOR EACH MODEL. THE HIGHER THE CHRF^{++} VALUE, THE BETTER THE MODEL IS.

	Iteration	3,000			6,000			9,000		
	Hidden State	64	128	256	64	128	256	64	128	256
None	Base	N/A	N/A	N/A	80.54	86.17	75.09	86.76	90.32	89.86
	+N-gram	N/A	N/A	N/A	88.03	92.19	84.98	92.39	94.66	94.24
	+SymSpell	N/A	N/A	N/A	88.84	92.22	86.36	92.26	94.28	93.99
	+BiLSTM-S2S	N/A	N/A	N/A	83.56	87.84	79.30	88.69	91.34	90.98
	+Transformer-S2S	N/A	N/A	N/A	**90.84**	**94.15**	87.93	**94.45**	**96.89**	**96.85**
	+mT5-base	N/A	N/A	N/A	70.46	71.48	69.48	71.33	71.89	71.53
	+mBART-50	N/A	N/A	N/A	89.23	89.50	**88.89**	89.52	89.81	89.76
+BiLSTM	Base	91.81	91.25	95.26	96.55	96.14	97.53	97.21	97.05	97.90
	+N-gram	95.31	94.32	97.36	97.91	97.63	98.40	98.24	98.19	98.54
	+SymSpell	95.20	94.82	97.32	97.85	97.71	98.37	98.17	98.19	98.31
	+BiLSTM-S2S	91.14	91.49	93.18	93.98	93.96	94.52	94.26	94.25	94.58
	+Transformer-S2S	**96.10**	**96.56**	**98.25**	**98.81**	**98.58**	**99.07**	**98.94**	**99.00**	**99.31**
	+mT5-base	71.43	71.44	72.19	72.15	72.05	72.19	71.97	72.04	72.11
	+mBART-50	89.65	89.51	89.79	89.82	89.77	89.77	89.80	89.82	89.80

2) Neural Machine Translation (NMT) Post-OCR correction: NMT-based approaches, including BiLSTM-S2S and Transformer-S2S, have shown promising results in reducing WER in Post-OCR correction tasks. BiLSTM-S2S, while effective, provides moderate improvements over base (optical) models. Transformer-S2S, however, consistently outperforms other methods, significantly lowering the WER. These methods leverage the power of sequence-to-sequence learning, where the model captures context and structural dependencies better than statistical models.

3) Large Language Model (LLM) Post-OCR correction: Large language models like mT5 and mBART have been applied to Post-OCR correction but underperform compared to Transformer-S2S. mT5 struggles with WER reductions, likely due to its broad generalization, which can lead to translation or substitution of words rather than domain-specific corrections. mBART-50 shows modest improvements in lower optical model iterations (3,000) but struggles in higher iterations (6,000 and 9,000) where fewer OCR errors exist, likely due to its tendency to generalize.

Based on the evaluated results, we also found out that with Post-OCR correction models, we can reduce our model complexity by reducing training iterations and hidden states. The performance differences between different hidden states are only around 1 to 2 points for all evaluation scores when the models were trained with iterations 6,000 and 9,000. According to the experimental results, Transformer-S2S demonstrates substantial improvements with the least WER, 0.66 for the optical model with feature sequence modeling and gained CHRF^{++} value of 99.31. Experimental results show that while LLMs can be applied to Post-OCR correction, it does not perform as efficiently as other specialized neural approaches.

V. Conclusion and Future Work

We evaluated deep OCR models across iterations (3,000, 6,000, 9,000) and hidden states (64, 128, 256), incorporating Post-OCR correction from statistical, NMT, and LLM approaches. Error analysis of the optical models revealed that the CTC decoding stage follows visual rather than writing order. Notably, Post-OCR correction significantly enhanced the performance of base optical models, with Transformer-S2S consistently outperforming all other approaches, especially in handling minimal OCR errors across different model iterations.

Looking ahead, we aim to test our models on manually annotated text image datasets with greater real-world variability and explore neural correction methods to address new error types. We will also investigate by tuning the hyperparameter such as maximum length, epoch, etc., and the additional LLMs tailored to the Myanmar language's unique characteristics. Additionally, we plan to publicly release the dataset, along with the best OCR models and Post-OCR correction configurations, to support further research and development in the low-resource Myanmar language community (https://github.com/ye-kyaw-thu/myOCR).

References

[1] D. Doermann, "The indexing and retrieval of document images," *Computer Vision and Image Understanding*, vol. 70, no. 3, pp. 287–298, 1998. [Online]. Available: https://doi.org/10.1006/cviu.1998.0692

[2] A. El-Sawy, M. Loey, and H. El-Bakry, "Arabic handwritten characters recognition using convolutional neural network," *WSEAS Transactions on Computer Research*, vol. 5, 2017. [Online]. ISSN: 2415-1521.

[3] H. A. Alwzwazy, H. M. Albehadili, Y. S. Alwan, and N. E. Islam, "Handwritten digit recognition using convolutional neural network," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 4, no. 2, pp. 1101–1106, 2016.

[4] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural network," in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR)*, Nov. 2012.

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

[6] K. Dutta, P. Krishnan, M. Mathew, and C. V. Jawahar, "Improving convolutional neural network, recurrent neural network hybrid networks for handwriting recognition," in *16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2018.

[7] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 31, pp. 855–868, 2009.

[8] 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 (TPAMI)*, vol. 39, pp. 2298–2304, 2017.

[9] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2006, pp. 369–376.

[10] P. Stefanović, O. Kurasova, and R. Štrimaitis, "The N-Grams Based Text Similarity Detection Approach Using Self-Organizing Maps and Similarity Measures," *Applied Sciences*, vol. 9, 2019, Art. no. 1870.

[11] W. Garbe, "1000x Faster Spelling Correction Algorithm," 2012. [Online]. Available: https://towardsdatascience.com/symspellcompound-10ec8f467c9b. [Accessed: 24-May-2024].

[12] A. Kayabas, A. E. Topcu, and Ö. Kiliç, "OCR Error Correction Using BiLSTM," in 2021 International Conference on Electrical, Computer and Energy Technologies (ICECET), 2021, pp. 1-5, doi: 10.1109/ICECET52533.2021.9698712.

[13] N. Yasin, I. Siddiqi, M. Moetesum, and S. A. Rauf, "Transformer-Based Neural Machine Translation for Post-OCR Error Correction in Cursive Text," in Document Analysis and Recognition – ICDAR 2023 Workshops: San José, CA, USA, August 24–26, 2023, Proceedings, Part II, Berlin, Heidelberg: Springer-Verlag, 2023, pp. 80–93, doi: 10.1007/978-3-031-41501-2_6.

[14] G. Madarász, N. Ligeti-Nagy, A. Holl, and T. Váradi, "OCR Cleaning of Scientific Texts with LLMs," in Natural Scientific Language Processing and Research Knowledge Graphs. NSLP 2024, G. Rehm, S. Dietze, S. Schimmler, and F. Krüger, Eds., vol. 14770, Cham: Springer, 2024, doi: 10.1007/978-3-031-65794-8_4.

[15] E. Soper, S. Fujimoto, and Y.-Y. Yu, "BART for Post-Correction of OCR Newspaper Text," in Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021), Online: Association for Computational Linguistics, 2021, pp. 284–290, doi: 10.18653/v1/2021.wnut-1.31.

[16] Z. Z. Hlaing, Y. K. Thu, T. Supnithi, and P. Netisopakul, "Improving Neural Machine Translation with POS-tag features for low-resource language pairs," *Heliyon*, 2022. [Online]. Available: https://doi.org/10.1016/j.heliyon.2022.e10375

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015, vol. 4, pp. 12.

[18] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, no. 8, pp. 1735-1780, 1997.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is All You Need," in Advances in Neural Information Processing Systems 30 (NIPS 2017), 2017.

[20] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel, "mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer," in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Online: Association for Computational Linguistics, 2021, pp. 483–498, doi: 10.18653/v1/2021.naacl-main.41.

[21] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer, "Multilingual Denoising Pre-training for Neural Machine Translation," Transactions of the Association for Computational Linguistics, vol. 8, pp. 726-742, 2020, doi: 10.1162/tacl_a_00343.

[22] M. Popović, "chrF," 2017. [Online]. Available: https://github.com/m-popovic/chrF. [Accessed: 24-May-2024].

[23] E. P. P. Mon, Y. K. Thu, T. T. Yu, and A. O. Wai Oo, "SymSpell4Burmese: Symmetric Delete Spelling Correction Algorithm (SymSpell) for Burmese Spelling Checking," in *2021 16th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)*, 2021, pp. 1-6. [Online]. Available: https://doi.org/10.1109/iSAI-NLP54397.2021.9678171

[24] NIST, "SCTK, the NIST Scoring Toolkit," [Online]. Available: https://github.com/usnistgov/SCTK. [Accessed: 19-Sept-2022].

[25] M. Hosken and M. Tuntunlwin, "Representing Myanmar in Unicode Details and Examples," 2004. [Online]. Available: https://api.semanticscholar.org/CorpusID:8745749