Enhancing OCR Performance through Post-OCR Models: Adopting Glyph Embedding for Improved Correction

Yung-Hsin Chen*
University of Zurich
yung-hsin.chen@uzh.ch

Yuli Zhou*
University of Zurich
yuli.zhou@uzh.ch

Abstract

The study investigates the potential of post-OCR models to overcome limitations in OCR models and explores the impact of incorporating glyph embedding on post-OCR correction performance. In this study, we have developed our own post-OCR correction model. The novelty of our approach lies in embedding the OCR output using CharBERT and our unique embedding technique, capturing the visual characteristics of characters. Our findings show that post-OCR correction effectively addresses deficiencies in inferior OCR models, and glyph embedding enables the model to achieve superior results, including the ability to correct individual words.

1 Introduction

Converting images into digital formats has become increasingly practical for preserving ancient documents (Avadesh and Goyal, 2018; Narang et al., 2020), understanding real-time road signs (Wu et al., 2005; Kim, 2020), multimodal information extraction (Liu et al., 2019; Sun et al., 2021) and evaluating text-guided image generation (Petsiuk et al., 2022; Zhang et al., 2023; Rodríguez et al., 2023), just to name a few. These digital representations can be further processed using language models to extract underlying features. A high-quality OCR model can benefit various domains. However, the performance of optical character recognition (OCR) is often limited due to factors such as image quality or inherent model limitations.

This study seeks to explore the potential of post-OCR correction, specifically examining whether it can compensate for the deficiencies of OCR models. The study comprises two main components. Firstly, we evaluated a range of OCR models and their post-OCR correction methods using language models to determine whether the correction process could mitigate weaknesses in the OCR results. Secondly, we introduce our own embedding and post-OCR correction model, leveraging CharBERT (Ma et al., 2020) for character embedding, as well as a custom glyph embedding developed by our team. This glyph embedding captures the visual characteristics of the characters, rather than solely focusing on semantic aspects like most conventional embeddings.

2 Data

In the initial phase of our study, we employ the extensively researched and widely employed ICDAR 2013 dataset as our primary data source for OCR tasks. The results obtained from this phase will subsequently serve as inputs for the next step, where we evaluate the performance of our post-OCR correction model. Additionally, to facilitate training of the glyph embedding in our post-OCR correction model, we rely on the Chars74K dataset[1].

The ICDAR 2013 dataset (Karatzas et al., 2013) and the recently introduced ICDAR 2023 dataset (Long et al., 2023) serve as prominent benchmarks for evaluating Optical Character Recognition (OCR) models.

The ICDAR 2013 dataset encompasses a diverse collection of document images, encompassing both printed and handwritten text captured under a variety of conditions, including different languages, font styles, sizes, and distortions. It serves as a robust benchmark for assessing the accuracy and robustness of OCR systems in a wide array of real-world scenarios.

In contrast, the ICDAR 2023 dataset expands upon the achievements of its predecessor by introducing additional challenges that push the bound-

* These authors contributed equally to this work.

[1] Available at http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/

aries of OCR technology. It incorporates more intricate document layouts, degraded text, low-resolution images, and challenging background clutter, with the goal of evaluating OCR model performance in increasingly demanding scenarios.

Both datasets will undergo evaluations using three models: EasyOCR[2], PaddleOCR[3], and TrOCR (Transformer-based OCR) (Li et al., 2021). The EasyOCR model will be utilized specifically for single word detection box, while the other two models will provide single line detection box functionality. Consequently, the outputs from EasyOCR will consist solely of single words, whereas PaddleOCR will generate output in the form of sentences.

The Chars74K dataset is employed for training the glyph embedding. This dataset encompasses scene images of English and Kannada characters, although for this particular experiment, only English alphabets are utilized. In addition to the Chars74K dataset, we capture screenshots of Korean and Hebrew characters to serve as samples for the garbage class in open-set classification.

For training the post-OCR correction model, the data consists of the outputs obtained from EasyOCR, PaddleOCR, and TrOCR in the initial phase.

3 Method

In this section, we will outline the methodologies employed to conduct the examinations of state-of-the-art (SOTA) models as well as our post-OCR correction model.

SOTA OCR models In recent years, OCR models have leveraged advanced machine learning and computer vision techniques to achieve remarkable levels of accuracy and efficiency. EasyOCR has emerged as a popular open-source OCR library. Within EasyOCR, the detection execution employs the CRAFT algorithm (Baek et al., 2019), while the recognition model is based on CRNN (Shi et al., 2016). The CRNN model comprises three key components: feature extraction utilizing ResNet (He et al., 2016) and VGG, sequence labeling employing LSTM, and decoding through CTC. PaddleOCR, another widely adopted OCR framework, is built on the PaddlePaddle deep learning platform and showcases state-of-the-art performance across various text recognition tasks. PaddleOCR supports multiple languages, scene text recognition, and text detection based on PP-OCRv3 (Li et al., 2021). Furthermore, TrOCR stands as an OCR model that capitalizes on transformer-based architectures to achieve exceptional recognition accuracy. By harnessing the power of transformers, TrOCR effectively captures contextual information and delivers impressive results across diverse OCR tasks.

Post-OCR Correction by Language Models Post-OCR correction using Language Models (LMs) has emerged as a robust methodology for enhancing the accuracy and quality of Optical Character Recognition (OCR) outputs. LMs, such as transformer-based architectures like BERT (Devlin et al., 2018) or GPT (Brown et al., 2020), have revolutionized the field of natural language processing. Within the realm of post-OCR correction, LMs leverage their contextual understanding and language modeling capabilities to detect and rectify errors present in the recognized text. By analyzing the neighboring words, sentence structure, and semantic context, LMs can effectively identify and rectify OCR mistakes, encompassing misspellings, punctuation errors, and grammatical inconsistencies. This approach has demonstrated its efficacy in significantly enhancing OCR accuracy, particularly in scenarios where OCR models face challenges associated with complex layouts, degraded text, or intricate font styles.

The post-OCR correction model is composed of two parallel encoders and one decoder. The encoders receive inputs in the form of sentences that have been embedded using CharBERT and the glyph embedding that we developed through training.

CharBERT and Glyph Embedding CharBERT is a pre-trained language model that takes a string as input and produces its corresponding embedding. The model combines BERT embedding with character-level embedding. It undergoes two pre-training tasks: masked word prediction[4] and noise deduction[5]. We chose this embedding for our post-OCR correction model because the denoising pre-training task is expected to aid in the correction process.

[2] https://github.com/JaidedAI/EasyOCR

[3] https://github.com/PaddlePaddle/PaddleOCR

[4] Similar to BERT's pretraining task, CharBERT randomly masks words for prediction.

[5] CharBERT introduces random noise, such as random characters, into the corpus and trains the model to denoise the text.

On the other hand, the glyph embedding is trained using open-set classification with a garbage class[6] and adapted softmax. ResNet18 and ResNet50 models are utilized, with the last fully-connected layer (fc) replaced and an additional fc layer appended for the purpose of fine-tuning. During inference, this additional fc layer is removed to generate an embedding of size 768 (Table 1). In fine-tuning, we experiment with unfreezing the last two fc layers as well as the last four layers. Unlike charBERT, which captures the semantic information of words, glyph embedding represents the visual glyph of characters.

After fine-tuning, images of the same character are passed through the model to generate embeddings. The averaged embedding is considered as the glyph embedding for that specific character.

Post-OCR Correction Model Once the glyph embedding training is complete, it becomes one of the inputs for the post-OCR correction model. This model is composed of two CNN encoders and one transformer decoder (Figure 1). CNN is a commonly employed architecture for processing character embeddings (Kim et al., 2016), while the transformer decoder offers a significant advantage in handling long-distance dependencies (Vaswani et al., 2017). Furthermore, the positional encoding within the transformer decoder aids in preserving the spatial information of the input sentence.

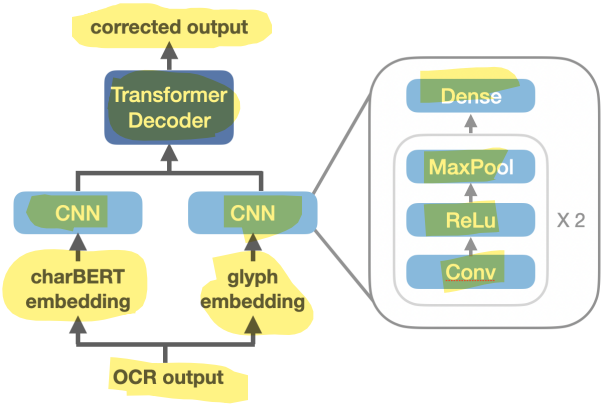

Figure 1: The post-OCR correction model consists of two parallel CNN encoders and a transformer decoder with CharBERT and glyph embedding as inputs.

[6]Data categorized as "garbage" refers to information that does not hold significance within the context of our study. Its function lies in training the model to classify irrelevant characters (specifically, non-English alphabets and non-digits in this instance) into a designated category termed the "garbage" class. In the scope of this research, the training data for the garbage class encompasses Hebrew and Korean characters.

4 Results

Evaluation Criteria To assess the recognition performance of the OCR model, we employ a word-level evaluation using full matching. In addition, for line-level evaluation, two metrics are utilized: Character Error Rate (CER) and Word Error Rate (WER). These metrics can be calculated using the equations 1, where S represents the number of substitutions, D represents the number of deletions, I represents the number of insertions, and C represents the total number of characters in the reference text. Similarly, WER can be calculated using equation 2, where N represents the total number of correct words. A lower value indicates better performance.

$$CER = \frac{S + D + I}{S + D + I + C} \quad (1)$$

$$WER = \frac{S_w + D_w + I_W}{N_w} \quad (2)$$

Evaluation on OCR Models with GPT Table 2 presents the performance of various OCR models on the word-level correctness of the ICDAR2013 dataset (about more than 800 images), considering both the original ground truth and the fixed ground truth. Our analysis focuses on EasyOCR, a slightly inferior OCR model, as well as two superior OCR models, PaddleOCR and TrOCR. Utilizing gpt-3.5-turbo-0301 for post-OCR correction demonstrates a noticeable improvement in the OCR results, indicating that the language model can enhance and compensate for the deficiencies to some extent.

In Table 3, We demonstrate the post-OCR correction process based on ICDAR 2023 at line-level (we have selected about 50 images with hierarchical text). Similarly, the language model proves beneficial in rectifying errors by leveraging contextual information.

Evaluation on Glyph Embedding Models Table 4 illustrates the performance of various models used for training glyph embedding. Unfreezing more than the two modified fc layers can lead to disturbances in the pre-trained ResNet model, resulting in decreased accuracy. Furthermore, in our specific case, the garbage class approach yields superior results compared to the adapted softmax approach.

Evaluation on Post-OCR Correction Models Table 5 demonstrates that the inclusion of glyph

Layer	in dim	out dim
Resnet50 last fc	2048	768
additional fc	768	63

Table 1: The dimensions of the last two layers of the modified ResNet50 architecture were adjusted for the pre-trained task.

embedding leads to a significant enhancement in the WER and CER scores at the sentence level. However, the impact on the performance at the single-word level is minimal. Notably, the model exhibits proficient post-correction capabilities at the single-word level, as evidenced by the low CER and WER scores.

5 Discussion

During the initial phase of the study, we observed that post-OCR correction possesses the capacity to mitigate the limitations of OCR models to a certain degree. This becomes particularly evident when processing inputs comprising complete sentences. The application of post-OCR correction by GPT to both PaddleOCR and TrOCR outputs resulted in notable and comparable improvements. However, in scenarios involving inputs containing a single word, such as those generated by EasyOCR, the impact of post-correction on performance enhancement is relatively minimal.

In the subsequent step, the comparative performance of the post-OCR correction models developed by us is examined. Notably, these models achieve results comparable to those generated by GPT (Table 3), while employing a significantly lighter weight model. It is worth mentioning that the aforementioned models are capable of effectively post-correcting individual words. Additionally, the inclusion of the glyph embedding leads to a substantial improvement in overall performance for all inputs.

However, the scope of this study is limited to characters within the ranges of 0-9, a-z, and A-Z. As a result, punctuations have not been included in the training process or considered during the evaluation. Nevertheless, this issue can be addressed and improved by incorporating additional training data for special characters and punctuations into the glyph embedding framework.

Model	Original GT	Fixed GT
EasyOCR	0.788	0.800
PaddleOCR	0.948	0.958
TrOCR	**0.952**	**0.965**
EasyOCR+GPT*	-	0.841
PaddleOCR+GPT*	-	0.967
TrOCR+GPT*	-	**0.969**

Table 2: The evaluation of word-level text was performed using various models, with accuracy calculated based on full matching. The term "Original GT" denotes the ground truth initially provided by the dataset, while recognizing that it may contain errors. Consequently, we manually rectified some of these errors, resulting in a revised ground truth referred to as "Fixed GT". "*GPT" specifically represents the GPT-3.5-turbo-0301 model.

Model	WER	CER
EasyOCR	0.67	0.58
PaddleOCR	0.45	**0.11**
TrOCR	**0.22**	0.12
EasyOCR+GPT*	0.66	0.55
PaddleOCR+GPT*	**0.11**	**0.08**
TrOCR+GPT*	**0.11**	0.09

Table 3: The evaluation of line-level text was conducted using diverse models, with accuracy measured through the utilization of Word Error Rate (WER) and Character Error Rate (CER). In this context, "*GPT" specifically denotes the GPT-3.5-turbo-0301 model.

Model	Acc (%)
RN18, garbage class, unfreeze L2	85.08
RN50, garbage class, unfreeze L2	**87.37**
RN50, garbage class, unfreeze L4	83.90
RN50, adapted softmax, unfreeze L4	83.19

Table 4: This table shows models employed for training glyph embedding and provides a comprehensive comparison of their performance. (RN refers to ResNet; L refers to number of layers)

OCR Output	Glyph	WER	CER
EasyOCR	False	-	0.0959
EasyOCR	True	-	**0.0292**
PaddleOCR	False	0.2642	0.1771
PaddleOCR	True	**0.0630**	**0.0283**
TrOCR	False	0.0859	0.0478
TrOCR	True	**0.0067**	**0.0013**

Table 5: The table provides a comprehensive comparison of post-OCR correction models that integrate glyph embedding with those that exclusively employ CharBERT embedding. The comparison encompasses various OCR model outputs.

References

Meduri Avadesh and Navneet Goyal. 2018. Optical character recognition for sanskrit using convolution neural networks. In *2018 13th IAPR International Workshop on Document Analysis Systems (DAS)*, pages 447–452.

Youngmin Baek, Bado Lee, Dongyoon Han, Sangdoo Yun, and Hwalsuk Lee. 2019. Character region awareness for text detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9365–9374.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazàn Almazàn, and Lluís Pere de las Heras. 2013. Icdar 2013 robust reading competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1484–1493.

Jaejoon Kim. 2020. Application on character recognition system on road sign for visually impaired: Case study approach and future. *International Journal of Electrical and Computer Engineering (IJECE)*, 10(1):778–785.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. 2016. Character-aware neural language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Minghao Li, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. 2021. Trocr: Transformer-based optical character recognition with pre-trained models. *arXiv preprint arXiv:2109.10282*.

Xiaojing Liu, Feiyu Gao, Qiong Zhang, and Huasha Zhao. 2019. Graph convolution for multimodal information extraction from visually rich documents. In *Proceedings of the NAACL-HLT*, pages 32–39, Minneapolis, Minnesota. Association for Computational Linguistics.

Shangbang Long, Siyang Qin, Dmitry Panteleev, Alessandro Bissacco, Yasuhisa Fujii, and Michalis Raptis. 2023. Icdar 2023 competition on hierarchical text detection and recognition. *arXiv preprint arXiv:2305.09750*.

Wentao Ma, Yiming Cui, Chenglei Si, Ting Liu, Shijin Wang, and Guoping Hu. 2020. CharBERT: Character-aware pre-trained language model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 39–50, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Sonika Rani Narang, Manish Kumar Jindal, and Munish Kumar. 2020. Ancient text recognition: a review. *Artificial Intelligence Review*, 53:5517–5558.

Vitali Petsiuk, Alexander E. Siemenn, Saisamrit Surbehera, Zad Chin, Keith Tyser, Gregory Hunter, Arvind Raghavan, Yann Hicke, Bryan A. Plummer, Ori Kerret, Tonio Buonassisi, Kate Saenko, Armando Solar-Lezama, and Iddo Drori. 2022. Human evaluation of text-to-image models on a multi-task benchmark.

Juan A. Rodríguez, David Vazquez, Issam Laradji, Marco Pedersoli, and Pau Rodriguez. 2023. Ocr-vqgan: Taming text-within-image generation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3689–3698.

Baoguang Shi, Xiang Bai, and Cong Yao. 2016. 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*, 39(11):2298–2304.

Lin Sun, Jiquan Wang, Kai Zhang, Yindu Su, and Fangsheng Weng. 2021. Rpbert: A text-image relation propagation-based BERT model for multimodal NER. In *Proceedings of AAAI 2021*, pages 13860–13868. AAAI Press.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wen Wu, Xilin Chen, and Jie Yang. 2005. Detection of text on road signs from video. *IEEE Transactions on Intelligent Transportation Systems*, 6(4):378–390.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. 2023. Magicbrush: A manually annotated dataset for instruction-guided image editing.