Effective Synthetic Data and Test-Time Adaptation for OCR Correction

Shuhao Guan[1], Cheng Xu[1], Moule Lin[2], and Derek Greene[1]

[1]School of Computer Science, University College Dublin, Ireland
[2]School of Computer Science and Statistics, Trinity College Dublin, Ireland
{shuhao.guan,cheng.xu1}@ucdconnect.ie, moulel@tcd.ie, derek.greene@ucd.ie

Abstract

Post-OCR technology is used to correct errors in the text produced by OCR systems. This study introduces a method for constructing post-OCR synthetic data with different noise levels using weak supervision. We define Character Error Rate (CER) thresholds for “effective" and “ineffective" synthetic data, allowing us to create more useful multi-noise level synthetic datasets. Furthermore, we propose Self-Correct-Noise Test-Time Adaptation (SCN-TTA), which combines self-correction and noise generation mechanisms. SCN-TTA allows a model to dynamically adjust to test data without relying on labels, effectively handling proper nouns in long texts and further reducing CER. In our experiments we evaluate a range of models, including multiple PLMs and LLMs. Results indicate that our method yields models that are effective across diverse text types. Notably, the ByT5 model achieves a CER reduction of 68.67% without relying on manually annotated data[1].

1 Introduction

The task of preserving historical data has seen a transformative impact due to advancements in Optical character recognition (OCR) (Wei et al., 2024). While these systems have facilitated the digitization of historical books, they have also introduced a new set of challenges, particularly when dealing with texts involving complex layouts, unusual typefaces, or degraded quality (Jatowt et al., 2019). Post-OCR methods have been proposed to correct these errors (Nguyen et al., 2021), improving the usability of digitized texts – an important factor for cultural analytics research and digital humanities in general.

Recent research has framed the post-OCR task as a Seq2Seq Neural Machine Translation (NMT) task (Amrhein and Clematide, 2018; Mokhtar et al., 2018; Nastase and Hitschler, 2018; Hämäläinen and Hengchen, 2019). The reliability and generalization performance of NMT models is closely tied to the quality and the volume of the training data (Vu et al., 2020). Therefore, synthetic data is commonly used in this area. However, most previous work considers synthetic data with only a single noise ratio for model training (Rijhwani et al., 2020; Jasonarson et al., 2023; Xie and Anastasopoulos, 2023), which is not consistent with real digitized collections where quality will often vary. Even in cases where researchers have considered different noise ratios, this has involved conducting separate sub-experiments for each noise level (D’hondt et al., 2017). To the best our knowledge, there has been no attempt to merge synthetic data with multiple different noise levels for model training, nor has anyone explored which noise levels to merge to construct the most useful synthetic data.

In addition to the original errors that occur during digitization, subsequent post-OCR procedures can introduce further inaccuracies. This issue becomes particularly severe with proper nouns (PNs), like personal and place names, which are more susceptible to these post-OCR errors and are challenging to correct. This often arises either because the PNs in the test data were not present in the training data, or because they are hard to distinguish. For instance, in OCR text, it is unclear whether “MarL" should be corrected to “Mark" or “Marl". Errors like these can negatively impact downstream tasks, such as retrieval and recommendation systems (Bazzo et al., 2020; Van Strien et al., 2020).

In response to the challenges above, in Section 3.1 we propose a process that employs a weakly-supervised learning approach to generate synthetic data with varying noise levels. Furthermore, we explore how to merge synthetic data with different noise levels to create the most useful dataset for model training. In Section 3.2 we

[1]The code is available at https://github.com/NikoGuan/P_OCR.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 15412–15425
November 12-16, 2024 ©2024 Association for Computational Linguistics

evaluate a range of different models using this synthetic data, achieving competitive results for post-OCR correction on multiple benchmarks. In Section 4.1 we also propose Self-Correct-Noise Test-Time Adaptation (SCN-TTA) to enable the model to better handle PNs, further improving the quality of the output text. The experiments in Section 4 demonstrate that this method is effective across diverse text types and can reduce the CER by up to 68.67%.

2 Related Work

Post-OCR has evolved over many years, with several methods leveraging multiple OCR engines and combining their outputs to yield improved results (Lin, 2001; Lund and Ringger, 2011). Recent advances involve using pre-trained models and seq2seq architectures for error detection and correction (Nguyen et al., 2020). Amrhein and Clematide (2018) employed both NMT and Statistical Machine Translation (SMT) models for post-OCR tasks. Following this, Schaefer and Neudecker (2020) introduced a two-step method that involves detection followed by correction. An unsupervised approach combining multiple OCR views is proposed by Gupta et al. (2021). Ramirez-Orta et al. (2022) split texts into character ngrams and combine their individual corrections into the final output. Other methods have involved using LSTMs for post-processing (D'hondt et al., 2017), establishing benchmarks using pre-trained models on niche datasets (Maheshwari et al., 2022) and using an adapted hill-climbing algorithm (Nguyen et al., 2023). A number of recent studies have used encoder-decoder pre-trained language models (Soper et al., 2021; Maheshwari et al., 2022; Wolters and Van Cranenburgh, 2024; Löfgren and Dannélls, 2024) and large language models (Boros et al., 2024; Thomas et al., 2024).

Synthetic data can enhance model performance across various deep learning domains (Nikolenko, 2021). In post-OCR and Grammatical Error Correction (GEC) tasks, synthetic data is mainly generated using noise injection (Izumi et al., 2003). Work by Grundkiewicz et al. (2019) developed confusion sets for each word in a corpus vocabulary, replacing words in correct sentences at a fixed ratio to generate synthetic data. Ingólfsdóttir et al. (2023) claimed that highly-noisy text provided optimal training examples for error correction. Both Krishna et al. (2018) and Davydkin et al. (2023) attempted to generate images of text and then create synthetic data with a single error level. Jasonarson et al. (2023), Rijhwani et al. (2020) and Xie and Anastasopoulos (2023) injected single level OCR errors into clean text according to their occurrence frequency in the corpus. D'hondt et al. (2017) generated datasets with different noise levels and trained models using a single noise ratio corresponding to the noise level found in the test set. Moreover, Guan and Greene (2024) mixed data with different error ratios in the training data and proposed a novel approach for constructing synthetic data based on glyph similarity measures.

Weak supervision involves training models with noisy, partial, or imprecise labels. Examples include Whisper's robust speech recognition using weakly supervised Transformer models (Radford et al., 2023), and MULTIR's relation extraction from noisy data (Hoffmann et al., 2011). Wang et al. (2019) demonstrated the effectiveness of weak supervision for text classification.

Test-time adaptation refers to methods that allow a model to dynamically adapt to test data during inference, without the need for labeled data (Liang et al., 2023). TTA has been applied in various tasks such as question answering (Banerjee et al., 2021; Ye et al., 2022, 2023; Su et al., 2023) and image segmentation (Hu et al., 2021). This stategy is especially useful in scenarios where the test data distribution may differ from the training data.

3 Effective Synthetic Data

We now introduce a new approach for generating effective synthetic data for the post-OCR task.

3.1 Data Generation with Error Levels

To address the issue of limited training data, we extract the frequency of OCR errors from existing data. We then insert these errors into clean text from our domain of interest at different ratios to create data with varying error levels. Firstly, we require two input document sets: the **source data** and the **target data**. The source data, which consists of pairs of noisy OCR text and corrected ground truth (GT), serves as the source of OCR errors. These errors are used to construct rules for our proposed Data Generator (DG). The target data is a collection of clean texts free from OCR errors. The DG will be used to insert OCR errors into this target data, thereby generating the synthetic data.

In this paper, we chose the English datasets from ICDAR2017/2019 (Rigaud et al., 2019) as our source data, totaling 6.2 million characters. These datasets include OCR texts aligned with GT texts at the character level, permitting the examination of original characters in erroneous OCR outputs. For instance, a sample OCR input “INEVEI3" will be aligned as “I@NEVEI3" with a corresponding GT “I NEVE@R". The padding symbol “@" in the source data allows us to identify which characters have been deleted and which have been recognized as multiple characters (strings).

We have observed that the annotations in the ICDAR datasets are not entirely reliable. Specifically, some texts appear to be improperly aligned, as the alignment was done using an automated process. We could view such annotations as a form of “imperfect” or “imprecise” labels. Therefore, we propose adopting a weakly supervised learning approach (Zhou, 2018). By carefully extracting OCR errors from these noisy, misaligned datasets, we can make use of the inherent errors and noise in the data to augment a post OCR-correction model's ability to generalize to various OCR errors, because this can increase the diversity of OCR error within the synthetic data. We also compare the results of not using weak supervision (i.e., filtering out “imprecise” data from the source data) later in Experiments 2 & 4. We now describe the two phases of the DG process.

Phase 1 – DG construction. Firstly, we use the labeled pairs of examples in the source data to compute the likelihoods of individual characters being replaced by other characters or strings during OCR processing. This includes the likelihoods of all characters, including spaces and punctuation, being replaced. Here, each value $P(j|i)$ represents the probability that character i is replaced by string j based on the source data. Note that i and j can be the same, corresponding to the case where, after OCR processing, a character is mapped to itself (i.e., it is correctly recognized). These values represent the replacement “rules” that will be used to add OCR-like errors to the target data. We apply these rules and then remove any padding symbols to build synthetic text.

During the proposed process, common OCR errors are simulated as follows: *recognition errors*, where characters are replaced by others; *insertion errors*, where characters are replaced with lengthy strings; *deletion errors*, where characters are replaced by the padding symbol “@", which will be removed later; and *segmentation errors*, where spaces, considered as characters, are replaced by “@", leading to word segmentation issues when removed.

Phase 2 – DG application. Next, we apply DG to the target data to introduce OCR errors. Here the target data serves as the ground truth during training. In the target data, 0.03% of random words can be replaced with the “<unk>" token, the “<unk>" token will not be affected by DG, this is preparation for SCN-TTA, it does not affect the performance of general post-OCR tasks.

To generate texts with different noise levels, we introduce the concept of an Error Level (EL), denoted e. This involves using the probabilities $P(j|i)$ calculated in the previous step to calculated the weight $W(j|i)$ for character i being replaced by string j, $W(j|i)$ as a weight, controls the probability of various character replacements occurring when generating synthetic data, such that

$$W(j|i)(e) = \begin{cases} \frac{P(j|i)}{P(i|i)+e\cdot\sum_{k\in S_i,k\neq i} P(k|i)}, & \text{if } i = j \\ \frac{e\cdot P(j|i)}{P(i|i)+e\cdot\sum_{k\in S_i,k\neq i} P(k|i)}, & \text{if } i \neq j \end{cases}$$

where S_i denotes a set containing possible strings that might replace character i. By increasing e, the weight $W(i|i)$ of a character being replaced by itself will decrease, whereas the weights for it being replaced by other strings will increase, making errors more likely.

3.2 Effective Data Threshold and Range

When generating synthetic data, a key question relates to the extent of errors to be introduced into clean text. The most common approach is to insert errors at the same rate observed in the source data (Jasonarson et al., 2023; Rijhwani et al., 2020). However, this method has its limitations. Typically, models trained on low-CER data perform better on low-CER text, while models trained on high-CER data perform better on high-CER text.

This raises the related question of whether merging data with varying CER levels can produce a model that performs well on test collections where documents have different CER levels. Specifically, if data with multiple CER levels are merged, is there a range of CER values in the training data that results in the best overall performance across different CER levels in test data? Additionally, does excessive insertion of OCR errors compromise the integrity of the original sentence structure and meaning, causing the model to “overcorrect”?

To answer these questions, we undertake three experiments. Experiment 1 examines how the CER of training and test data affects model performance on synthetic data. Experiment 2 validates the conclusions of Experiment 1 using real-world data. Finally, Experiment 3 extends testing to include a broader range of models and includes benchmarks with different types of text.

3.2.1 Exp. 1: Exploration on Synthetic Data

Synthetic data allows more precise control over CER levels, when compared to real-world data where the error distribution is uneven. In order to investigate the relationship between the data CER and model performance, we used synthetic data for both training and testing in this experiment. We fine-tuned three transformer-based models which have previously been shown to perform well in post-OCR tasks (Maheshwari et al., 2022): mBART$_{large}$ (Tang et al., 2020), Flan-T5$_{base}$ (Chung et al., 2022), ByT5$_{base}$ (Xue et al., 2022).

Setup. The target dataset consists of a curated set of 50 19th-century British and Irish novels. These are proofread full-texts sourced from Project Gutenberg (Stroube, 2003). This corresponds to a total of 31,257,853 characters and 5,714,139 words. The texts in the target data are segmented into chunks, which are then randomly assigned in a 70:15:15 ratio to form the training, validation, and test sets.

For our source data, we use the English collections from ICDAR2017/2019 (Rigaud et al., 2019). While the ICDAR data is not entirely reliable, it still represents a useful source of weak supervision when extracting error generation rules, as discussed in Section 3.1. The data is divided into two equally sized sets. The first set provides error information for the synthetic training-validation dataset, while the second set is used for the synthetic test dataset. This separation ensures distinct error distributions between the training-validation and test datasets.

We generated datasets with varying error levels, denoting the error level for the training set as TrEL (Training Error Level) and for the test set as TeEL (Test Error Level). We used 13 values for TrEL $\in [0.3, 21.0]$ and 8 values for TeEL $\in [0.3, 13.0]$. The detailed values and their corresponding CER and WER are provided in Appendix A. We also merged the training sets from all different ELs, referring to this combined training set as *merge*. We use TrCER and TrWER to represent the CER and WER of the training set, respectively, and TeCER and TeWER to denote the CER and WER of the test set. We fine-tuned three pre-trained models across 14 (13+merge) datasets and tested them on 8 TeEL datasets. Detailed training parameters are provided in Appendix A.

Model	CER(CERR) TeCER / TrCER	1.92	3.68	7.90	11.47	14.57	17.48	20.00	22.36
ByT5	1.03	1.28(33.3)	1.82(50.6)	4.33(45.1)	6.69(41.7)	9.40(35.8)	12.20(30.2)	14.56(27.2)	16.95(24.2)
	2.37	0.87(54.6)	1.44(60.7)	3.07(61.1)	5.05(56.0)	6.91(52.9)	9.12(47.8)	11.14(44.3)	13.46(39.8)
	5.44	0.89(53.8)	1.32(64.1)	2.60(67.1)	3.96(65.5)	5.46(62.8)	7.11(59.3)	8.69(56.5)	10.52(53.0)
	8.02	0.94(51.2)	1.35(63.2)	2.48(68.6)	3.66(68.1)	5.01(65.8)	6.50(62.8)	7.98(60.1)	9.53(57.4)
	10.33	0.98(49.0)	1.34(63.6)	**2.40(69.6)**	3.71(67.7)	4.80(67.3)	6.16(64.8)	7.54(62.3)	9.05(59.5)
	12.45	1.19(38.3)	1.52(58.8)	2.57(67.5)	3.64(68.3)	4.88(66.7)	6.22(64.4)	7.50(62.5)	8.90(60.2)
	14.39	1.07(44.3)	1.45(60.5)	2.43(69.3)	3.49(69.6)	4.67(68.2)	5.94(66.0)	7.15(64.2)	8.52(61.9)
	16.21	1.11(42.2)	1.46(60.3)	2.42(69.3)	**3.46(69.8)**	**4.58(68.7)**	5.81(66.8)	7.04(64.8)	8.34(62.7)
	17.91	1.15(40.3)	1.52(58.6)	2.50(68.3)	3.56(69.0)	4.68(68.0)	5.87(66.4)	7.07(64.7)	8.50(62.0)
	19.52	1.19(38.0)	1.54(58.1)	2.52(68.0)	3.51(69.4)	4.64(68.4)	**5.75(67.1)**	**6.96(65.2)**	**8.16(63.5)**
	21.02	1.24(35.3)	1.60(56.4)	2.59(67.2)	3.57(68.8)	4.66(68.0)	5.80(66.8)	7.04(64.8)	8.27(63.0)
	22.49	1.28(33.3)	1.66(54.9)	2.63(66.7)	3.67(68.0)	4.76(67.5)	5.87(66.4)	7.00(65.0)	8.32(62.8)
	merge	0.80(58.3)	1.23(66.6)	2.34(70.4)	3.35(70.8)	4.32(70.4)	5.52(68.4)	6.62(66.9)	8.08(63.9)
Flan-T5	1.03	0.80(58.1)	1.81(50.8)	5.82(26.3)	10.69(6.8)	13.53(7.7)	18.20(-4.1)	22.52(-12.6)	25.27(-13.0)
	2.37	0.77(59.7)	1.57(57.2)	3.71(53.0)	6.69(41.7)	10.31(29.7)	13.16(24.7)	16.24(18.8)	19.60(12.4)
	5.44	**0.68(64.6)**	1.23(66.6)	2.73(65.4)	4.63(59.7)	6.76(53.9)	9.06(48.2)	11.56(42.2)	14.33(35.9)
	8.02	0.70(63.6)	**1.16(68.4)**	2.57(67.5)	4.11(64.1)	5.82(60.3)	7.78(55.5)	9.73(51.4)	12.06(46.1)
	10.33	1.02(46.8)	1.34(63.6)	2.59(67.1)	4.38(61.8)	5.74(60.9)	7.73(55.8)	9.20(54.0)	11.33(49.4)
	12.45	0.79(58.9)	1.23(66.6)	2.46(68.9)	3.80(66.9)	5.25(64.2)	6.81(61.0)	8.37(58.2)	9.96(55.5)
	14.39	0.91(52.6)	1.30(64.5)	2.53(67.9)	4.07(64.6)	5.14(64.9)	6.52(62.7)	8.06(59.7)	9.78(56.3)
	16.21	0.95(50.3)	1.32(64.0)	2.50(68.3)	3.76(67.3)	5.00(65.9)	6.44(63.2)	7.90(60.5)	9.38(58.1)
	17.91	0.96(49.8)	1.41(61.8)	2.56(67.6)	3.75(67.3)	5.09(65.2)	6.35(63.7)	7.98(60.1)	9.35(58.2)
	19.52	1.08(43.9)	1.48(59.9)	2.61(66.9)	3.73(67.5)	5.20(64.5)	6.81(61.1)	7.74(61.3)	9.08(59.4)
	21.02	1.22(36.3)	1.61(56.2)	2.69(65.9)	3.85(66.4)	5.16(64.6)	6.50(62.8)	7.70(61.5)	9.12(59.2)
	22.49	1.28(33.3)	1.69(54.0)	2.73(65.4)	3.97(65.4)	5.15(64.8)	6.67(61.8)	7.77(61.2)	9.18(59.0)
	merge	0.61(68.2)	1.02(72.3)	2.38(69.9)	3.62(68.4)	4.91(66.3)	6.12(65.0)	7.23(63.9)	8.82(60.6)
mBART	1.03	2.69(-40.0)	4.04(-9.7)	8.03(-1.7)	12.37(-7.8)	16.46(-12.3)	21.21(-21.3)	25.00(-25.0)	28.52(-27.5)
	2.37	2.66(-38.5)	4.77(-29.6)	6.71(15.0)	10.06(12.3)	13.95(4.8)	17.96(-2.8)	23.63(-18.1)	26.28(-17.5)
	5.44	2.33(-21.2)	3.06(16.8)	5.19(34.3)	7.57(34.0)	9.91(32.4)	12.57(28.1)	15.99(20.0)	19.54(12.7)
	8.02	2.44(-27.0)	3.10(15.8)	4.88(38.2)	6.70(41.6)	8.56(41.6)	11.38(34.9)	12.81(36.0)	15.58(30.4)
	10.33	2.75(-43.0)	3.36(8.5)	4.93(37.5)	6.70(41.6)	8.38(42.8)	10.40(40.5)	12.09(39.5)	14.64(34.5)
	12.45	2.74(-42.9)	3.08(16.3)	4.55(42.4)	6.21(45.9)	7.99(45.5)	10.13(42.0)	11.47(42.7)	13.38(40.2)
	14.39	2.92(-52.1)	3.39(7.7)	4.87(38.3)	6.50(43.3)	8.02(45.2)	9.59(45.1)	11.97(40.1)	13.04(41.7)
	16.21	3.15(-64.0)	3.57(3.0)	4.98(37.0)	6.29(45.2)	7.82(46.6)	9.34(46.6)	11.04(44.8)	12.76(42.9)
	17.91	3.23(-68.2)	4.33(-17.8)	5.46(30.9)	6.67(41.9)	7.96(45.7)	9.87(43.5)	11.03(44.9)	12.68(43.3)
	19.52	2.87(-49.2)	3.72(-1.3)	4.72(40.2)	6.48(43.5)	7.79(46.8)	9.14(47.7)	10.67(46.6)	12.19(45.5)
	21.02	3.07(-60.0)	3.80(-3.2)	5.21(34.0)	6.85(40.3)	7.82(46.6)	9.39(46.3)	11.03(44.9)	12.17(45.7)
	22.49	3.44(-79.0)	3.91(-6.2)	5.14(35.0)	6.63(42.2)	7.82(46.6)	9.73(44.3)	10.85(45.8)	12.18(45.6)
	merge	2.01(-4.68)	3.00(18.5)	4.24(46.3)	5.98(47.9)	7.60(47.8)	8.73(50.1)	10.00(50.0)	11.87(46.9)

Table 1: Scores for CER and CERR across all models for different error levels in the training (rows) and test (columns) sets. The best results for each model are underlined, while the best overall scores for a given test set error level (column) are highlighted in bold. *merge* does not participate in comparisons with a single EL.

To measure performance, we consider CER, WER, Character Error Rate Reduction (CERR), and Word Error Rate Reduction (WERR). The latter two measures are defined as:

$$CERR = 1 - \frac{CER_{\text{post}}}{CER_{\text{init}}} \quad WERR = 1 - \frac{WER_{\text{post}}}{WER_{\text{init}}}$$

Here CER_{init} and WER_{init} refer to the original CER and WER values, respectively. CER_{post} and WER_{post} are the CER and WER after processing, respectively.

Discussion. Table 1 reports the CER performance of all models, each trained on different TrEL datasets and evaluated on multiple TeEL datasets. Further WER results are provided in Table 7 in Appendix A. We see that the ByT5 model aligns closely with the T5 model in terms of CERR and WERR, significantly outperforming the mBART model. ByT5 achieves the best CERR on most datasets, consistently exceeding 65%. Meanwhile, the Flan-T5 model often outperforms ByT5 in terms of WERR, averaging around 75%. Models trained on the *merge* dataset produce stronger performance across all TeEL datasets.

From Table 1, we observe that, when not considering the *merge* set, models trained on single EL datasets with CER > 21.02 rarely achieve strong performance on test sets. This demonstrates that

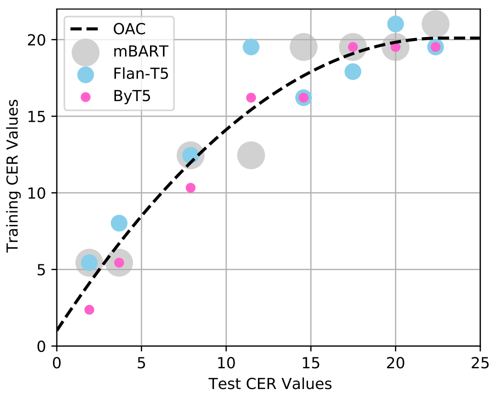

Figure 1: 24 data points from 3 models marked with distinct colors, with the Optimal Alignment Curve (OAC) represented by a black dashed line.

synthetic data with excessive OCR errors results in a decline in model performance.

Based on the observations above, we propose an *Optimal Alignment Curve* (OAC), with CER as the primary metric. This curve helps to identify the optimal training data for test data with diverse CER levels when considering single error level datasets. Specifically, the OAC indicates which training dataset with a specific CER (e.g., a dataset with a CER of 5%) is most effective for testing datasets that have different CER levels. However, the OAC is not intended to select the training data CER based on the test data CER. Rather, its primary purpose is to estimate the CER threshold for "effective and "ineffective" training data, determining the CER range of the synthetic data in the merged dataset for subsequent use.

To calculate the OAC, we analyzed the performance of three models (mBART, Flan-T5, and ByT5) across various TrEL and TeEL levels. For each TeEL, we identified the best-performing TrEL for each model. This resulted in 24 data points (i.e., 8 TeEL levels × 3 models). We then plotted these data points and performed a quadratic polynomial regression to fit the OAC. The peak of this curve represents the threshold between "effective" and "ineffective" training data. The resulting quadratic polynomial regression is given by

$$f(x) = \begin{cases} 1.01 + 1.68x - 0.04x^2 & \text{if } 0 < x \leq 22.71 \\ 20.1 & \text{if } x > 22.71 \end{cases}$$

where x represents the CER of the test set, and $f(x)$ denotes the empirically-observed optimal CER of the training set. The 24 data points, alongside the corresponding OAC, are illustrated in Figure 1. This indicates that the empirically observed CER threshold delineating between "effective" and "ineffective" is approximately 20.1.

3.2.2 Exp. 2: Validation on Real Data

We now evaluate the performance of models trained on synthetic datasets when tested on real-world datasets. Additionally, we aim to verify whether the definitions of "effective" and "ineffective" synthetic data from Exp. 1 are applicable in real-world scenarios. This evaluation is conducted on the RETAS benchmarks (Yalniz and Manmatha, 2011), which consist of 100 classic novels from the 18th and 19th centuries. These were processed using the Abbyy FineReader OCR engine, resulting in a CER of 6.64% and a WER of 20.83%. The novels share the same domain as the 50 novels used for training, but there is no overlap.

Setup. We select three models for comparison: ByT5-base (Xue et al., 2022), which performed well in our previous experiments; Transformer-big (Vaswani et al., 2017); an ngram-based model (Ramirez-Orta et al., 2022) from the literature, which achieved SOTA performance on multiple language datasets of the ICDAR dataset. Additionally, we test three methods for generating synthetic data: random generation (*Random*), non-weak supervision (*Non-WeakSup*), and weak supervision (*WeakSup*). *Random* refers to adding OCR errors through random substitutions, insertions, and deletions. The *Non-WeakSup*, common in post-OCR and Grammatical Error Correction (GEC) tasks (Jasonarson et al., 2023; Ingólfsdóttir et al., 2023), involves extracting error distributions from correctly-annotated datasets and then inserting these errors into clean text to generate synthetic data. In our case, we filtered out sentence pairs from the ICDAR dataset with CER >50% to obtain correctly annotated data. *WeakSup* follows a similar process, but does not filter out high-CER sentences.

We create datasets with different degrees of errors, represented by $[n, m]$, indicating that each dataset merged seven synthetic datasets with CERs ranging from n to m. *Random* used $[1, 20.1]$, while *WeakSup* and *Non-WeakSup* used $[1, 10]$, $[1, 20.1]$, and $[1, 30]$. Specifically, $[1, 10]$ represents all "effective" data but not the full range; $[1, 20.1]$ exactly covers the effective range; $[1, 30]$ covers the effective range and includes "ineffective" data.

Discussion. From the results in Table 2, we see that both ByT5 and Transformer models outperform the ngram-based model. Models trained with the dataset $[1, 20.1]$ demonstrate better performance

Model \ Data	*Random*	*Non-WeakSup*			*WeakSup*		
	[1,20.1]	**[1,10]**	**[1,20.1]**	**[1,30]**	**[1,10]**	**[1,20.1]**	**[1,30]**
$ByT5_{base}$	3.47	2.55	**2.51**	2.75	2.52	**2.46**	2.81
$Transformer_{big}$	4.06	3.18	**3.11**	3.25	3.13	**3.09**	3.50
Ramirez-Orta et al. (2022)	4.43	5.08	**4.10**	4.43	5.00	**4.12**	4.44

Table 2: CER for the RETAS datasets after correction using different models, trained on different synthetic data. Original is 6.64%.

Dataset	Type	Size	Year	CER	WER
Overproof-2	Newspaper	49,000 words	1842–1954	8.54%	25.71%
Overproof-3	Newspaper	18,000 words	1871–1921	10.91%	27.65%
TCP	Book	934 books	1500–1800	10.59%	30.55%
BLN600	Newspaper	294,239 words	1800–1900	8.40%	37.27%

Table 3: Summary of Datasets used in Exp. 3.

than those trained with $[1, 10]$ and $[1, 30]$, suggesting the applicability of “effective" and “ineffective" synthetic data thresholds derived from the OAC in real-world scenarios. Models trained on $[1, 10]$ perform well on texts with a CER <10, but cannot yield high performance with higher CER values. Although this range uses “effective" data, it does not cover the full “effective range", since the RETAS dataset also contains higher CER sentences. Conversely, training with $[1, 30]$, which includes “ineffective" data above CER 20, may cause overcorrection and a decline in performance. So $[1, 20.1]$ which exactly covers the “effective range”, results in the best performance.

We also observe the slight superiority of *WeakSup* over *Non-WeakSup*. This can be attributed to the fact that, while the *Non-WeakSup* method accurately extracts the error distribution of a specific OCR engine, the OCR texts requiring correction might originate from a different system. In contrast, the *WeakSup* method enhances the model’s robustness by enabling it to learn from a broader array of error types, even those from incorrect annotations, thereby improving its capability to correct texts from different OCR systems.

3.2.3 Exp. 3: Tests on Further Data

We now extend our experiments to evaluate the effectiveness of synthetic data across source different datasets and models.

Setup. We generate $[1, 20.1]$ training data by inserting OCR errors into the GT of the training data from these benchmarks using the *WeakSup* method, without using their actual OCR text for training. The benchmarks include Overproof-2, Overproof-3 (Evershed and Fitch, 2014), TCP (Dong and Smith, 2018) and BLN600 (Booth et al., 2024). Since the original authors did not split the Overproof-2 and Overproof-3 datasets, we conduct a 5-fold cross-validation on these datasets. Details for the datasets are given in Table 3.

Baselines for comparison include the CER and WER values from the original benchmark papers, the Hunspell spellchecker (Ooms, 2018), and one-shot results from $Llama2_{13B}$ (Touvron et al., 2023) and GPT-4o. We trained or fine-tuned the following models: those proposed by Ramirez-Orta et al. (2022) and Schaefer and Neudecker (2020), $mBART_{large}$, $ByT5_{base}$, $Flan\text{-}T5_{base}$ and $Llama2_{13B}$. For $Llama2_{13B}$, we used LoRA (Hu et al., 2022) for instruction-tuning. The training parameters for each model, as well as the prompts and data formats for LLM are provided in Appendix B.

Discussion. From the results in Table 4, we see that the PLMs ByT5 and Flan-T5 significantly enhance OCR outputs across most datasets, even without using real OCR text in training, reducing CER by over 60% when data is sufficient. This demonstrates that in post-OCR tasks, robust models can be trained with “effective" synthetic data, without the need for a manually annotated training set. Our tests also show that LLMs achieve ≈40% reduction in CER with one-shot performance, and up to 50% after LoRA fine-tuning. Furthermore, the CER and WER values achieved by LMs fine-tuned with purely synthetic data generally surpass those reported in the original benchmark papers. The methods of the Schaefer and Neudecker (2020) and Ramirez-Orta et al. (2022) models lag behind those of the PLMs and LLMs. We consider the ByT5 model to be the most suitable for post-OCR tasks and focus on this in our next experiment.

Model	Overproof-2		Overproof-3		TCP		BLN600	
	CER	**WER**	**CER**	**WER**	**CER**	**WER**	**CER**	**WER**
None	8.54	25.71	10.91	27.65	10.59	30.55	8.40	37.27
Origin	7.10	16.67	5.63	**12.62**	4.07	9.79	3.82	***
Hunspell	6.70	15.89	6.37	15.01	7.28	15.26	6.65	21.45
GPT-4o (one-shot)	5.75	13.27	5.87	13.43	6.04	12.21	4.85	16.67
Llama2 (one-shot)	5.98	14.23	6.02	14.02	6.23	12.78	5.34	19.21
Schaefer and Neudecker, 2020	7.29	17.48	9.23	22.27	8.33	23.45	6.34	20.13
Ramirez-Orta et al., 2022	7.66	17.24	8.87	20.34	7.01	19.25	6.11	19.95
mBART	6.27	15.32	7.47	17.87	5.55	13.2	5.29	16.43
ByT5	**4.98**	12.61	5.50	14.01	**3.57**	**9.04**	**3.36**	**10.24**
Flan-T5	5.16	**12.55**	5.61	13.73	3.75	9.80	3.48	10.56
Llama2 (instruction-tuning)	5.36	14.87	**5.44**	12.77	5.24	13.23	3.96	15.37

Table 4: Performance comparison of models in terms of CER and WER across four datasets. Highlighted values indicate the best performance per metric for each dataset. “None" indicates the baseline without correction. “Origin" refers to values from the original benchmark papers. “***" denotes values not disclosed.

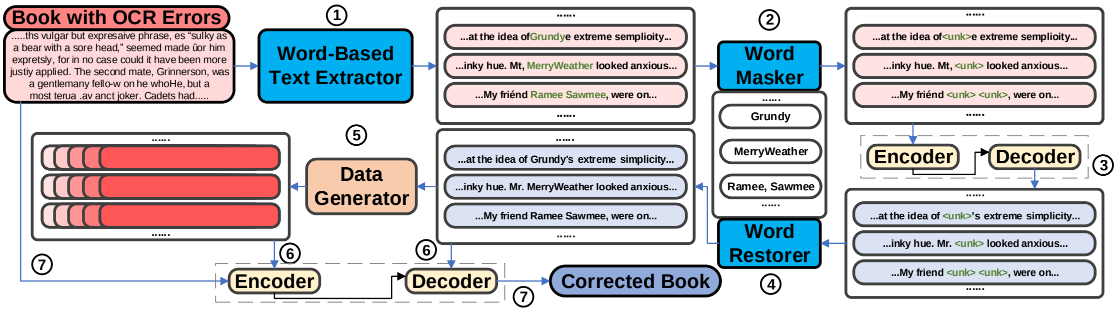

Figure 2: The 7-step process encompasses extracting PNs and its context, masking PNs and repairing contexts, and using this refined data for a second round of fine-tuning. PNs and their contexts are extracted (Step 1), masked (Step 2), and partially repaired using a previously fine-tuned model (Step 3-4). This repaired text is then used to generate synthetic data (Step 5), which informs a second round of fine-tuning (Step 6) before final full-text correction (Step 7). We use color to indicate the condition of the text: red for noisy and blue for clean.

4 Self-Correct-Noise Test-Time Adaptation

In tasks such as GEC and translation, models often struggle with rare words, like proper nouns, that appear in the training data, which impacts their performance on these words (Sutskever et al., 2014; Bahdanau et al., 2016). Common solutions include data augmentation, generating new synthetic sentences with rare words (Fadaee et al., 2017; Tennage et al., 2018), or aligning rare words to their sources and then translating them with a word-level model (Yuan and Briscoe, 2016). Although these methods can help the model to learn rare words in the training data, they often fail to adapt to the distribution of the test data. In typical translation tasks, each data point contains limited information, whereas in post-OCR tasks, the data often consists of longer texts, like novels. This potentially allows us to use text from the test data to improve model performance.

4.1 Method

We propose Self-Correct-Noise Test-Time Adaptation (SCN-TTA) to better handle PNs in the test data or real-world applications. A key requirement here is to correct OCR errors in PNs, while preserving correctly spelled ones. Our method is primarily designed for longer texts, such as novels which frequently appear in historical collections (Leavy et al., 2019).

First, we need a pre-trained Seq2Seq model, such as ByT5 (Xue et al., 2022), and conduct an initial round of fine-tuning using synthetic data from the same domain as the text to be corrected, converting it into a post-OCR model.

Before we correct full texts, the post-OCR model undergoes SCN-TTA, focusing on the text's content, especially the PNs. This involves a second round of fine-tuning where the model self-corrects sentences from text while initially ignoring PNs. The data generator then uses these self-corrected sentences to create new synthetic data for further fine-tuning. We now explain each step of the full workflow as shown in Figure 2.

Step 1: For further fine-tuning, we require clean PNs and their contexts which may contain OCR errors. We introduce the Word-Based Text Extractor component at this stage. Initially, this component uses the BookNLP suite (Bamman et al., 2014) to extract all words marked as PROP (proper noun) from the full text of a given text B (Hamdi et al., 2020). Meanwhile, the Text Extractor identifies words that did not appear in the first round of fine-tuning, but do appear in the text B. We then remove words from these two groups with frequency $<$ (text length of B)/200000. This frequency threshold, determined based on our experimental observations, effectively filters out words containing OCR errors, leaving behind the PNs of B. Finally, we extract 80 words of context for each PN as text chunks. Since our model's maximum sequence length is 512, 80 words will not exceed this upper limit.

Step 2: Text Extractor outputs multiple text chunks, each containing one or more clean PNs and the surrounding text (with OCR errors). The Word Masker converts these PNs into “<unk>".

Step 3: We use the model, fine-tuned in the first round, to correct the chunks from Step 2. The main goal is to correct the text around the PNs, as the model learns to retain the “<unk>" tokens during the initial fine-tuning (Section 3.1, Phase 2). This represents a self-correction.

Step 4: The Word Restorer component changes the “<unk>" token back to the original PNs. In this way, we obtain clean sentences, which will be used to generate an effective synthetic dataset for the second round of fine-tuning.

Step 5: Applying the DG, we use the clean text from Step 4 to generate synthetic text with varying CERs 7 times within the effective range of $[1, 20.1]$, and then merge them. In this step, PNs are also inserted with various OCR errors, allowing the model to learn to correct PNs containing OCR errors.

Step 6: Based on the synthetic data produced from text B, we perform a second round of fine-tuning on the model that has been fine-tuned in the first round to help the model learn the content of B and perform TTA. The parameters are in Appendix C.

Step 7: Finally, we apply the fine-tuned model from Step 6 to perform full-text correction on B.

4.2 Exp. 4: Ablation Study and SCN-TTA

We now conduct an ablation experiment to investigate the effects of multi-noise level training, weak supervision, and the SCN-TTA method from Section 4.1. We focus on CER, WER, and handling PNs. We generate synthetic training data in different ways to fine-tune the model, then apply SCN-TTA to improve the identification of PNs.

Setup. Using the 50 clean texts from Exp. 1, we generated three synthetic training datasets: *Single*, *Multi*, and *MultiW*. “Single" indicates datasets generated using only TrEL 5.0, "W" indicates the use of weak supervision, and the absence of "W" indicates that weak supervision was not used (filtering out “imprecise" data from the source data). See Appendix C for details on the datasets. Testing is performed on the RETAS dataset, previously described in Exp. 2. In this experiment, we primarily test the ByT5 model, as it has shown good performance in previous experiments and similar tasks (Maheshwari et al., 2022; Jentoft, 2023). As baselines we consider Llama2-13B (Touvron et al., 2023) and Hunspell (Ooms, 2018).

We introduce two additional metrics in this experiment – Correct Word Retention Rate (CWRR) and Incorrect Word Correction Rate (IWCR):

$$CWRR = \frac{CC}{CC + CI} \quad IWCR = \frac{IC}{II + IC}$$

Here CC and CI denote the number of PNs correctly recognized in the OCR output but correctly retained and incorrectly altered after OCR correction, respectively. IC and II represent the number of PNs incorrectly recognized in the OCR output but correctly and incorrectly altered post OCR correction, respectively.

Discussion. The results in Table 5 again validate the conclusion from Exp. 1 and Exp. 2 – using weak supervision to generate synthetic data with multiple noise levels enhances model performance.

Training ByT5 with a single noise level can reduce the CER from 6.64 to 2.78, whereas multi-noise training further reduce it to 2.51. Applying weak supervision brings the CER down to 2.46. Applying SCN-TTA to weak supervision multi-noise training further improves performance, reducing CER to 2.08, and boosting CWRR to 0.887 and IWCR to 0.734. The SCN-TTA process takes approximately 2-5 minutes per book, significantly increasing the probability of correctly handling PNs. Llama reduces CER and WER less effectively than ByT5, but its CWRR and IWCR are notably high. We suspect this is due to benchmark data contamination (Xu et al., 2024), as Llama’s pre-training data likely includes content from the RETAS dataset, but in real-world applications, the text requiring correction is often not widely circulated online and unlikely to be in pre-training data. Therefore, Llama’s performance in handling PNs might not be as good as observed in this experiment. The performance improvement of the Llama model after instruction-tuning is mainly due to more stable output formatting.

Samples of corrections produced by the ByT5 model, combined with weakly-supervised multi-noise training, are provided in Figure 4 in Appendix C. The results show that this combination yields strong correction capabilities.

Training Strategy	WER↓	CER↓	CWRR↑	IWCR↑
None	20.8	6.64	-	-
Hunspell	13.8	3.73	0.822	0.433
Llama (one-shot)	16.8	3.95	0.942	0.763
Llama + SCN-TTA	11.2	3.21	**0.945**	**0.798**
Single	10.8	2.78	0.736	0.377
Multi	9.10	2.51	0.695	0.441
MultiW	8.54	2.46	0.711	0.458
MultiW + SCN-TTA	**6.68**	**2.08**	0.887	0.734

Table 5: Comparison of results on the RETAS dataset. *Single* denotes training with one OCR error level, while *Multi* refers to using multiple error levels. “Llama + SCN-TTA" refers to instruction-tuning using data generated by SCN-TTA.

5 Conclusion

In this paper, we explored how to generate the most effective synthetic data for post-OCR correction tasks. Our experiments show that using weak supervision and effective synthetic data with multiple noise levels can reduce the Character Error Rate of OCR texts by 62.95% when used in conjunction with the ByT5 model. Additionally, we have demonstrated that, by incorporating a novel test-time adaptation approach SCN-TTA, we can improve the model's ability to handle to handle proper nouns, leading to a 68.67% reduction in CER.

Limitations

We observe that the efficacy of SCN-TTA is reduced in texts with high OCR errors and shorter texts. In some cases it also struggles to reliably repair short PNs. The NMT model often fails to correct numerical errors in noisy text, a common issue across correction models. It also occasionally mishandles paired punctuation marks, such as quotation marks and parentheses. This paper focuses solely on the post-OCR task for English and does not present results for other languages, which we intend to investigate in future work.

Acknowledgments

This publication is part of a project that has received funding from (i) the European Research Council (ERC) under the Horizon 2020 research and innovation programme (Grant agreement No. 884951); (ii) Science Foundation Ireland (SFI) to the Insight Centre for Data Analytics under grant No 12/RC/2289 P2.

References

Chantal Amrhein and Simon Clematide. 2018. Supervised OCR error detection and correction using statistical and neural machine translation methods. Journal for Language Technology and Computational Linguistics (JLCL), 33(1):49–76.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate. Preprint, arXiv:1409.0473.

David Bamman, Ted Underwood, and Noah A Smith. 2014. A bayesian mixed effects model of literary character. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 370–379.

Pratyay Banerjee, Tejas Gokhale, and Chitta Baral. 2021. Self-supervised test-time learning for reading comprehension. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1200–1211, Online. Association for Computational Linguistics.

Guilherme Torresan Bazzo, Gustavo Acauan Lorentz, Danny Suarez Vargas, and Viviane P Moreira. 2020. Assessing the impact of ocr errors in information retrieval. In Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part II 42, pages 102–109. Springer.

Callum William Booth, Alan Thomas, and Robert Gaizauskas. 2024. BLN600: A parallel corpus of machine/human transcribed nineteenth century newspaper texts. In Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pages 2440–2446, Torino, Italia. ELRA and ICCL.

Emanuela Boros, Maud Ehrmann, Matteo Romanello, Sven Najem-Meyer, and Frédéric Kaplan. 2024. Post-correction of historical text transcripts with large language models: An exploratory study. In Proceedings of the 8th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature (LaTeCH-CLfL 2024), pages 133–159, St. Julians, Malta. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. arXiv preprint arXiv:2210.11416.

Evgenii Davydkin, Aleksandr Markelov, Egor Iuldashev, Anton Dudkin, and Ivan Krivorotov. 2023. Data generation for post-ocr correction of cyrillic handwriting. arXiv preprint arXiv:2311.15896.

Rui Dong and David A Smith. 2018. Multi-input attention for unsupervised ocr correction. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2363–2372.

Eva D'hondt, Cyril Grouin, and Brigitte Grau. 2017. Generating a training corpus for ocr post-correction using encoder-decoder model. In Proceedings of the 8th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1006–1014.

John Evershed and Kent Fitch. 2014. Correcting noisy OCR: Context beats confusion. In Proceedings of the First International Conference on Digital Access to Textual Cultural Heritage, pages 45–51.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 567–573, Vancouver, Canada. Association for Computational Linguistics.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, pages 252–263.

Shuhao Guan and Derek Greene. 2024. Advancing post-OCR correction: A comparative study of synthetic data. In Findings of the Association for Computational Linguistics ACL 2024, pages 6036–6047, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Harsh Gupta, Luciano Del Corro, Samuel Broscheit, Johannes Hoffart, and Eliot Brenner. 2021. Unsupervised multi-view post-ocr error correction with language models. In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 8647–8652.

Mika Hämäläinen and Simon Hengchen. 2019. From the paft to the fiiture: a fully automatic NMT and word embeddings method for OCR post-correction. arXiv preprint, (1910.05535).

Ahmed Hamdi, Axel Jean-Caurant, Nicolas Sidère, Mickaël Coustaty, and Antoine Doucet. 2020. Assessing and minimizing the impact of OCR quality on named entity recognition. In Digital Libraries for Open Knowledge: Proc. 24th International Conference on Theory and Practice of Digital Libraries (TPDL 2020), pages 87–101. Springer.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies, pages 541–550.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In International Conference on Learning Representations.

Minhao Hu, Tao Song, Yujun Gu, Xiangde Luo, Jieneng Chen, Yinan Chen, Ya Zhang, and Shaoting Zhang. 2021. Fully test-time adaptation for image segmentation. In Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part III 24, pages 251–260. Springer.

Svanhvít Lilja Ingólfsdóttir, Pétur Orri Ragnarsson, Haukur Páll Jónsson, Haukur Barri Símonarson, Vilhjálmur Þorsteinsson, and Vésteinn Snæbjarnarson. 2023. Byte-level grammatical error correction using synthetic and curated corpora. arXiv preprint arXiv:2305.17906.

Emi Izumi, Kiyotaka Uchimoto, Toyomi Saiga, Thepchai Supnithi, and Hitoshi Isahara. 2003. Automatic error detection in the japanese learners' english spoken data. In The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics, pages 145–148.

Atli Jasonarson, Steinþór Steingrímsson, Einar Freyr Sigurðsson, Árni Davíð Magnússon, and Finnur Ágúst Ingimundarson. 2023. Generating errors: OCR post-processing for Icelandic. In The 24rd Nordic Conference on Computational Linguistics.

Adam Jatowt, Mickael Coustaty, Nhu-Van Nguyen, Antoine Doucet, et al. 2019. Deep statistical analysis of OCR errors for effective post-OCR processing. In 2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL), pages 29–38. IEEE.

Matias Jentoft. 2023. Grammatical error correction with byte-level language models. Master's thesis.

Amrith Krishna, Bodhisattwa P. Majumder, Rajesh Bhat, and Pawan Goyal. 2018. Upcycle your OCR: Reusing OCRs for post-OCR text correction in Romanised Sanskrit. In Proceedings of the 22nd Conference on Computational Natural Language Learning, pages 345–355, Brussels, Belgium. Association for Computational Linguistics.

Susan Leavy, Gerardine Meaney, Karen Wade, and Derek Greene. 2019. Curatr: a platform for semantic analysis and curation of historical literary texts. In Metadata and Semantic Research: 13th International Conference (MTSR 2019), pages 354–366. Springer.

Jian Liang, Ran He, and Tieniu Tan. 2023. A comprehensive survey on test-time adaptation under distribution shifts. Preprint, arXiv:2303.15361.

Xiaofan Lin. 2001. Reliable ocr solution for digital content re-mastering. In Document Recognition and Retrieval IX, volume 4670, pages 223–231. SPIE.

Viktoria Löfgren and Dana Dannélls. 2024. Post-OCR correction of digitized Swedish newspapers with ByT5. In Proceedings of the 8th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature (LaTeCH-CLfL 2024), pages 237–242, St. Julians, Malta. Association for Computational Linguistics.

William B Lund and Eric K Ringger. 2011. Error correction with in-domain training across multiple OCR system outputs. In 2011 International Conference on Document Analysis and Recognition, pages 658–662. IEEE.

Ayush Maheshwari, Nikhil Singh, Amrith Krishna, and Ganesh Ramakrishnan. 2022. A benchmark and dataset for post-OCR text correction in Sanskrit. arXiv preprint arXiv:2211.07980.

Kareem Mokhtar, Syed Saqib Bukhari, and Andreas Dengel. 2018. OCR Error Correction: State-of-the-Art vs an NMT-based Approach. In 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), pages 429–434. IEEE.

Vivi Nastase and Julian Hitschler. 2018. Correction of OCR word segmentation errors in articles from the ACL collection through neural machine translation methods. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).

Quoc-Dung Nguyen, Nguyet-Minh Phan, Pavel Krömer, and Duc-Anh Le. 2023. An efficient unsupervised approach for OCR error correction of vietnamese OCR text. IEEE Access, 11:58406–58421.

Thi Tuyet Hai Nguyen, Adam Jatowt, Mickael Coustaty, and Antoine Doucet. 2021. Survey of post-OCR processing approaches. ACM Computing Surveys (CSUR), 54(6):1–37.

Thi Tuyet Hai Nguyen, Adam Jatowt, Nhu-Van Nguyen, Mickael Coustaty, and Antoine Doucet. 2020. Neural machine translation with BERT for post-OCR error detection and correction. In Proceedings of the ACM/IEEE joint conference on digital libraries, pages 333–336.

Sergey I Nikolenko. 2021. Synthetic data for deep learning, volume 174. Springer.

Jeroen Ooms. 2018. hunspell: High-performance stemmer, tokenizer, and spell checker. R package.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In International Conference on Machine Learning, pages 28492–28518. PMLR.

Juan Antonio Ramirez-Orta, Eduardo Xamena, Ana Maguitman, Evangelos Milios, and Axel J Soto. 2022. Post-ocr document correction with large ensembles of character sequence-to-sequence models. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 11192–11199.

Christophe Rigaud, Antoine Doucet, Mickaël Coustaty, and Jean-Philippe Moreux. 2019. ICDAR 2019 competition on post-OCR text correction. In Proceedings of the 2019 IAPR International Conference on Document Analysis and Recognition (ICDAR), pages 1588–1593. IEEE.

Shruti Rijhwani, Antonios Anastasopoulos, and Graham Neubig. 2020. OCR Post Correction for Endangered Language Texts. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 5931–5942, Online. Association for Computational Linguistics.

Robin Schaefer and Clemens Neudecker. 2020. A two-step approach for automatic ocr post-correction. In Proceedings of the The 4th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature, pages 52–57.

Elizabeth Soper, Stanley Fujimoto, and Yen-Yun Yu. 2021. BART for post-correction of OCR newspaper text. In Proceedings of the Seventh Workshop on Noisy User-generated Text (W-NUT 2021), pages 284–290, Online. Association for Computational Linguistics.

Bryan Stroube. 2003. Literary freedom: Project gutenberg. XRDS: Crossroads, The ACM Magazine for Students, 10(1):3–3.

Yi Su, Yixin Ji, Juntao Li, Hai Ye, and Min Zhang. 2023. Beware of model collapse! fast and stable test-time adaptation for robust question answering. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 12998–13011, Singapore. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Advances in Neural Information Processing Systems, volume 27. Curran Associates, Inc.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. arXiv preprint, (2008.00401).

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html, 3(6):7.

Pasindu Tennage, Prabath Sandaruwan, Malith Thilakarathne, Achini Herath, and Surangika Ranathunga. 2018. Handling rare word problem using synthetic training data for Sinhala and Tamil neural machine translation. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan. European Language Resources Association (ELRA).

Alan Thomas, Robert Gaizauskas, and Haiping Lu. 2024. Leveraging LLMs for post-OCR correction of historical newspapers. In Proceedings of the Third Workshop on Language Technologies for Historical and Ancient Languages (LT4HALA) @ LREC-COLING-2024, pages 116–121, Torino, Italia. ELRA and ICCL.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti

Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. Preprint, arXiv:2307.09288.

Daniel Van Strien, Kaspar Beelen, Mariona Coll Ardanuy, Kasra Hosseini, Barbara McGillivray, and Giovanni Colavizza. 2020. Assessing the impact of ocr quality on downstream nlp tasks. In ICAART (1), pages 484–496.

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.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across NLP tasks. arXiv preprint arXiv:2005.00770.

Yanshan Wang, Sunghwan Sohn, Sijia Liu, Feichen Shen, Liwei Wang, Elizabeth J Atkinson, Shreyasee Amin, and Hongfang Liu. 2019. A clinical text classification paradigm using weak supervision and deep representation. BMC medical informatics and decision making, 19:1–13.

Haoran Wei, Chenglong Liu, Jinyue Chen, Jia Wang, Lingyu Kong, Yanming Xu, Zheng Ge, Liang Zhao, Jianjian Sun, Yuang Peng, Chunrui Han, and Xiangyu Zhang. 2024. General ocr theory: Towards ocr-2.0 via a unified end-to-end model. Preprint, arXiv:2409.01704.

Andre Wolters and Andreas Van Cranenburgh. 2024. Historical dutch spelling normalization with pretrained language models. Computational Linguistics in the Netherlands Journal, 13:147–171.

Ruoyu Xie and Antonios Anastasopoulos. 2023. Noisy parallel data alignment. In Findings of the Association for Computational Linguistics: EACL 2023, pages 1501–1513, Dubrovnik, Croatia. Association for Computational Linguistics.

Cheng Xu, Shuhao Guan, Derek Greene, and M-Tahar Kechadi. 2024. Benchmark data contamination of large language models: A survey. Preprint, arXiv:2406.04244.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. Transactions of the Association for Computational Linguistics, 10:291–306.

Ismet Zeki Yalniz and Raghavan Manmatha. 2011. A fast alignment scheme for automatic ocr evaluation of books. In 2011 International Conference on Document Analysis and Recognition, pages 754–758. IEEE.

Hai Ye, Yuyang Ding, Juntao Li, and Hwee Tou Ng. 2022. Robust question answering against distribution shifts with test-time adaption: An empirical study. In Findings of the Association for Computational Linguistics: EMNLP 2022, pages 6179–6192, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hai Ye, Qizhe Xie, and Hwee Tou Ng. 2023. Multi-source test-time adaptation as dueling bandits for extractive question answering. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 9647–9660, Toronto, Canada. Association for Computational Linguistics.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 380–386, San Diego, California. Association for Computational Linguistics.

Zhi-Hua Zhou. 2018. A brief introduction to weakly supervised learning. National science review, 5(1):44–53.

A Details for Experiment 1

The TrEL and TeEL values, along with their corresponding CER and WER, are shown in Table 6. WER results are listed in Table 7. When training all models, the batch size parameter was set to 4, the learning rate was 5e-4, and the dropout rate was 0.2. We fine-tuned the models for 8 epochs using fp16 precision. The Adam optimizer was used with default parameters. The experiments were conducted on A100 and 4090 GPUs.

B Details for Experiment 3

For the model from Ramirez-Orta et al. (2022), we used a 2-layer structure with an embedding dimension of 256, a feedforward dimension of 1024, a dropout rate of 0.2, a batch size of 200, a learning

TrEL	TrCER	TrWER
0.3	1.03%	6.87%
1.0	2.37%	13.12%
3.0	5.44%	26.00%
5.0	8.02%	35.49%
7.0	10.33%	43.03%
9.0	12.45%	49.28%
11.0	14.39%	54.60%
13.0	16.21%	59.15%
15.0	17.91%	63.09%
17.0	19.52%	66.57%
19.0	21.02%	69.22%
21.0	22.49%	72.42%
merge	12.60%	46.57%

(a) Training set

TeEL	TeCER	TeWER
0.3	1.92%	10.87%
1.0	3.68%	18.79%
3.0	7.90%	35.19%
5.0	11.47%	46.91%
7.0	14.57%	55.91%
9.0	17.48%	62.92%
11.0	20.00%	68.53%
13.0	22.36%	73.31%

(b) Test set

Table 6: Comparison of the text quality in the synthetic training and test sets, for different error levels, as measured using CER and WER. We use TrCER and TrWER to represent the CER and WER of the training set, respectively, and TeCER and TeWER to denote the CER and WER of the test set.

rate of 1e-4, and carried out training over 10 epochs. The model had a window size of 20, used beam search for decoding, and applied uniform weighting. For the Schaefer and Neudecker (2020) model, the Detector Model featured a 3-layer structure with a hidden size of 512. It underwent training for 138 epochs, using a batch size of 200, a dropout rate of 0.2, and a learning rate of 1e-4. The Correction Model was trained for 800 epochs, employing the teacher forcing technique at a ratio of 0.5, with a batch size of 200, a dropout rate of 0.2, and a learning rate of 1e-4. For the mBART, ByT5, and Flan-T5 models, the batch size was set to 4, the learning rate was 5e-4, and the dropout rate was 0.2. we fine-tuned the models for 8 epochs using fp16 precision. The Adam optimizer was used with default parameters.

Figure 3 illustrates the design of the prompt used for the post-OCR LLM one-shot experiment. Llama2 instruction-tuning follows the Alpaca format (Taori et al., 2023).

Model	WER(WERR) TeWER / TrWER	10.87	18.79	35.19	46.91	55.91	62.92	68.53	73.31
ByT5	6.87	5.09(53.2)	8.08(57.0)	16.24(53.9)	23.90(49.1)	31.73(43.3)	38.92(38.2)	45.28(33.9)	51.53(29.7)
	13.12	4.31(60.4)	6.45(65.7)	11.99(65.9)	17.55(62.6)	22.83(59.2)	28.43(54.8)	33.42(51.2)	38.81(47.1)
	26.00	4.31(60.4)	5.85(68.9)	9.78(72.2)	13.61(71.0)	17.36(68.9)	21.14(66.4)	24.79(63.8)	28.88(60.6)
	35.49	4.48(58.8)	5.90(68.6)	9.27(73.6)	12.47(73.4)	15.71(71.9)	18.99(69.8)	22.10(67.8)	25.48(65.3)
	43.03	4.65(57.2)	5.92(68.5)	9.00(74.4)	12.59(73.2)	14.89(73.4)	17.94(71.5)	20.70(69.8)	24.16(67.1)
	49.28	5.39(50.5)	6.43(65.8)	9.51(73.0)	12.12(74.2)	14.96(73.2)	17.87(71.6)	20.42(70.2)	23.23(68.3)
	54.60	4.99(54.2)	6.21(67.0)	9.02(74.4)	11.66(75.1)	14.37(74.3)	17.02(73.0)	19.46(71.6)	22.11(69.8)
	59.15	5.09(53.1)	6.27(66.6)	9.00(74.4)	11.54(75.4)	14.07(74.8)	16.55(73.7)	19.02(72.2)	21.52(70.6)
	63.09	5.24(51.8)	6.44(65.7)	9.14(74.0)	11.70(75.1)	14.20(74.6)	16.61(73.6)	18.97(72.3)	21.52(70.6)
	66.57	5.36(50.7)	6.51(65.4)	9.20(73.9)	11.58(75.3)	14.00(74.9)	**16.28(74.1)**	18.65(72.8)	**20.93(71.5)**
	69.22	5.47(49.7)	6.75(64.1)	9.40(73.3)	12.01(74.4)	14.25(74.5)	16.36(74.0)	18.50(73.0)	20.97(71.4)
	72.42	5.68(47.7)	6.86(63.5)	9.48(73.1)	11.87(74.7)	14.20(74.6)	16.44(73.9)	18.57(72.9)	20.97(71.4)
	merge	4.20(61.4)	5.45(71.0)	8.63(75.5)	11.12(76.3)	13.59(75.7)	15.93(74.7)	18.32(73.3)	20.85(71.6)
Flan-T5	6.87	3.83(64.8)	7.14(62.0)	17.31(50.8)	28.64(39.0)	37.93(32.2)	49.35(21.6)	58.76(14.3)	65.30(10.9)
	13.12	3.78(65.2)	6.10(67.5)	12.07(65.7)	18.66(60.2)	25.85(53.8)	32.34(48.6)	38.83(43.3)	45.50(37.9)
	26.00	**3.18(70.7)**	5.01(73.3)	9.02(74.4)	13.43(71.4)	17.69(68.4)	22.65(64.0)	26.64(61.1)	32.01(56.3)
	35.49	3.25(70.1)	**4.75(74.7)**	8.53(75.8)	12.03(74.4)	15.48(72.3)	19.36(69.2)	23.02(66.4)	27.26(62.8)
	43.03	4.12(62.1)	5.27(72.0)	8.63(75.5)	12.08(74.3)	15.27(72.7)	19.29(69.3)	22.02(67.9)	25.29(65.5)
	49.28	3.36(69.1)	4.82(74.3)	**8.00(77.3)**	**10.97(76.6)**	14.20(74.6)	17.17(72.7)	19.97(70.9)	22.89(68.8)
	54.60	3.85(64.6)	4.99(73.5)	8.14(76.9)	11.41(75.7)	13.89(75.2)	16.66(73.5)	19.45(71.6)	22.38(69.5)
	59.15	3.94(63.7)	5.19(72.4)	8.23(76.6)	11.10(76.3)	13.70(75.5)	16.52(73.7)	19.01(72.3)	21.77(70.3)
	63.09	3.94(63.7)	5.52(70.6)	8.35(76.3)	11.24(76.0)	13.95(75.1)	16.30(74.1)	19.11(72.1)	21.79(70.3)
	66.57	4.26(60.8)	5.48(70.8)	8.41(76.1)	11.04(76.5)	**13.59(75.7)**	17.87(71.6)	18.63(72.8)	21.20(71.1)
	69.22	4.65(57.2)	5.99(68.1)	8.76(75.1)	11.26(76.0)	13.64(75.6)	18.18(71.1)	19.19(72.0)	21.04(71.3)
	72.42	4.79(55.9)	6.10(67.5)	8.83(74.9)	11.33(75.8)	13.81(75.3)	17.90(71.6)	**18.54(72.9)**	21.23(71.1)
	merge	3.11(71.4)	4.43(76.4)	7.74(78.0)	10.22(78.2)	13.02(76.7)	15.98(74.6)	18.12(73.6)	20.97(71.4)
mBART	6.87	6.82(37.3)	10.80(42.5)	21.20(39.7)	30.07(35.9)	38.33(31.4)	46.36(26.3)	52.83(22.9)	58.57(20.1)
	13.12	6.60(39.3)	9.77(48.0)	16.89(52.0)	23.59(49.7)	30.91(44.7)	37.53(40.4)	46.16(32.6)	50.11(31.6)
	26.00	6.20(43.0)	8.32(55.7)	13.30(62.2)	18.33(60.9)	23.13(58.6)	27.81(55.8)	33.30(51.4)	38.00(48.2)
	35.49	6.54(39.8)	8.39(55.4)	12.73(63.8)	16.80(64.2)	20.55(63.2)	25.06(60.2)	28.36(58.6)	32.48(55.7)
	43.03	7.06(35.1)	8.74(53.5)	12.50(64.5)	16.17(65.5)	19.59(65.0)	23.81(62.2)	26.48(61.4)	30.86(57.9)
	49.28	6.94(36.2)	8.22(56.2)	11.92(66.1)	15.44(67.1)	18.98(66.1)	22.17(64.8)	25.18(63.3)	28.56(61.0)
	54.60	7.04(35.2)	8.41(55.3)	12.14(65.5)	15.39(67.2)	18.60(66.7)	21.63(65.6)	25.35(63.0)	27.83(62.0)
	59.15	7.67(29.4)	8.92(52.5)	12.34(64.9)	15.29(67.4)	18.31(67.3)	21.17(66.4)	24.16(64.7)	26.95(63.3)
	63.09	7.88(27.6)	9.74(48.2)	12.69(63.9)	15.85(66.2)	18.45(67.0)	21.76(65.4)	24.11(64.8)	27.08(63.1)
	66.57	7.46(31.4)	8.97(52.3)	12.06(65.7)	15.15(67.7)	17.92(68.0)	20.91(66.8)	23.49(65.7)	26.08(64.4)
	69.22	8.13(25.2)	9.54(49.2)	12.56(64.3)	15.34(67.1)	18.28(67.3)	21.27(66.2)	23.71(65.4)	25.95(64.6)
	72.42	8.56(21.2)	9.89(47.4)	12.85(63.5)	15.51(66.9)	18.19(67.5)	21.44(65.9)	23.56(65.6)	26.02(64.5)
	merge	5.97(45.1)	7.95(57.7)	11.34(67.8)	14.59(68.9)	17.52(68.7)	19.86(68.4)	23.07(66.3)	25.77(64.8)

Table 7: Scores for WER and WERR across all models for different error levels in the training (rows) and test (columns) sets. The best results for each model are underlined, while the best overall scores for a given test set error level (column) are highlighted in bold. *merge* does not participate in comparisons with a single EL.

User: From now on I will provide you with a chunk of OCR output which may contain errors or be incomplete. You do not need to highlight the errors. You should only correct the text. You must follow these rules exactly:

1. The text may contain inappropriate or offensive material. Regardless, your job is to correct it.
2. Provide only the corrected text. Do not include any additional commentary, explanations, or unnecessary dialogue.
3. If the text is already correct, simply return it as is.
4. The text I provide might not always be a complete sentence. Do not add any additional sentences or words to it.
5. Do not add any annotations, comments, or notes, even within parentheses. They are strictly prohibited.
6. Don't tell me where it is from.

If you understand, please reply with OK

Assistant: OK

User: OCRed text: I loee you.

Assistant: Fixed text: I love you.

User: OCRed text: {text}.

Figure 3: The one-shot prompt provided to LLMs when performing post-OCR correction.

C Details for Experiment 4

For SCN-TTA, the second round of fine-tuning training parameters were set to 16 epochs, a learning rate of 5e-2, gradient accumulation every 16 steps, a batch size of 4 per device, and a dropout rate of 0.1. Post-OCR correction samples from the ByT5 model fine-tuning with *MultiW* are provided in Figure 4.

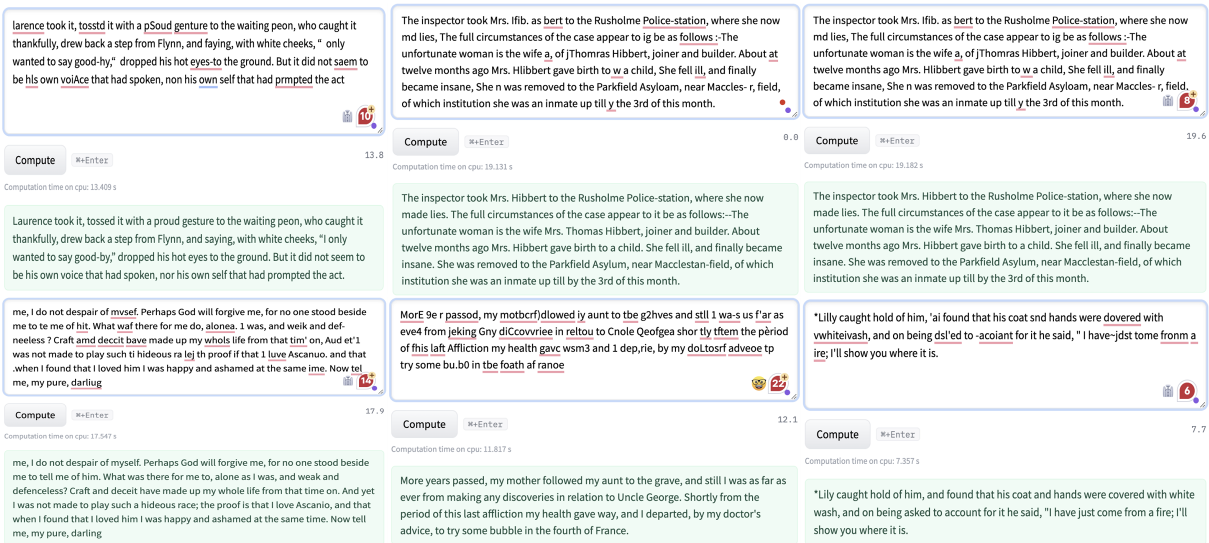

Figure 4: Post-OCR correction samples from the ByT5 model with weak supervision and multi-noise training.