Multi-Input Attention for Unsupervised OCR Correction

Rui Dong **David A. Smith**
College of Computer Information and Science
Northeastern University
{dongrui, dasmith}@ccs.neu.edu

Abstract

We propose a novel approach to OCR post-correction that exploits repeated texts in large corpora both as a source of noisy target outputs for unsupervised training and as a source of evidence when decoding. A sequence-to-sequence model with attention is applied for single-input correction, and a new decoder with multi-input attention averaging is developed to search for consensus among multiple sequences. We design two ways of training the correction model without human annotation, either training to match noisily observed textual variants or bootstrapping from a uniform error model. On two corpora of historical newspapers and books, we show that these unsupervised techniques cut the character and word error rates nearly in half on single inputs and, with the addition of multi-input decoding, can rival supervised methods.

1 Introduction

Optical character recognition (OCR) software has made vast quantities of printed material available for retrieval and analysis, but severe recognition errors in corpora with low quality of printing and scanning or physical deterioration often hamper accessibility (Chiron et al., 2017). Many digitization projects have employed manual proofreading to further correct OCR output (Holley, 2009), but this is time consuming and depends on fostering a community of volunteer workers. These problems with OCR are exacerbated in library-scale digitization by commercial (e.g., Google Books, Newspapers.com), government (e.g., Library of Congress, Bibliothèque nationale de France), and nonprofit (e.g., Internet Archive) organizations. The scale of these projects not only makes it difficult to adapt OCR models to their diverse layouts and typefaces but also makes it impractical to present any OCR output other than a single-best transcript.

Existing methods for automatic OCR post-correction are mostly supervised methods that correct recognition errors in a single OCR output (Kolak and Resnik, 2002; Kolak et al., 2003; Yamazoe et al., 2011). Those systems are not scalable since human annotations are expensive to acquire, and they are not capable of utilizing complementary sources of information. Another line of work is ensemble methods (Lund et al., 2013, 2014) combining OCR results from multiple scans of the same document. Most of these ensemble methods, however, require aligning multiple OCR outputs (Lund and Ringger, 2009; Lund et al., 2011), which is intractable in general and might introduce noise into the later correction stage. Furthermore, voting-based ensemble methods (Lund and Ringger, 2009; Wemhoener et al., 2013; Xu and Smith, 2017) only work where the correct output exists in one of the inputs, while classification methods (Boschetti et al., 2009; Lund et al., 2011; Al Azawi et al., 2015) are also trained on human annotations.

To address these challenges, we propose an *unsupervised* OCR post-correction framework both to correct *single* input text sequences and also to exploit *multiple* candidate texts by simultaneously *aligning, correcting, and voting among input sequences*. Our proposed method is based on the observation that significant number of duplicate and near-duplicate documents exist in many corpora (Xu and Smith, 2017), resulting in OCR output containing repeated texts with various quality. As shown by the example in Table 1, different errors (characters in red) are introduced when the OCR system scans the same text in multiple editions, each with its own layout, fonts, etc. For ex-

ample, in is recognized as m in the first output and a is recognized as u in the third output, while the second output is correctly recognized. Therefore, duplicated texts with diverse errors could serve as complementary information sources for each other.

OCR Output	eor**y that I have been slam in battle, for I sorry that I have been slain in battle, for I sorry tha' I have been s_uin in battle, f_r I
Original Text	sorry that I have been slain in battle, for I

Table 1: Example duplicate texts in OCR'd digital corpora

In this paper, we aim to train an unsupervised correction model via utilizing the duplication in OCR output. We propose to map each erroneous OCR'd text unit to either its high-quality duplication or a consensus correction among its duplications via bootstrapping from an uniform error model. The baseline correction system is a sequence-to-sequence model with attention (Bahdanau et al., 2015), which has been shown to be effective in text correction tasks (Chollampatt et al., 2016; Xie et al., 2016).

We also seek to improve the correction performance for duplicated texts by integrating multiple inputs. Previous work on combining multiple inputs in neural translation deal with data from different domains, e.g., multilingual (Zoph and Knight, 2016) or multimodal (Libovický and Helcl, 2017) data. Therefore, their models need to be trained on multiple inputs to learn parameters to combine inputs from each domain. Given that the inputs of our task are all from the same domain, our model is trained on a single input and introduces multi-input attention to generate a consensus result merely for decoding. It does not require learning extra parameters for attention combination and thus is more efficient to train. Furthermore, average attention combination, a simple multi-input attention mechanism, is proposed to improve both the effectiveness and efficiency of multi-input combination on the OCR post-correction task.

We experiment with both **supervised and unsupervised training** and with **single- and multi-input decoding** on data from two manually transcribed collections in English with diverse typefaces, genres, and time periods: newspaper articles from the Richmond (Virginia) Daily Dispatch (RDD) from 1860–1865 and books from 1500–1800 from the Text Creation Partnership (TCP). For both collections, which were manually transcribed by other researchers and are in the public domain, we aligned the one-best output of an OCR system to the manual transcripts. We also aligned the OCR in the training and evaluation sets to other public-domain newspaper issues (from the Library of Congress) and books (from the Internet Archive) to find multiple duplicates as "witnesses", where available, for each line. Experimental results on both datasets show that our proposed averarge attention combination mechanism is more effective than existing methods in integrating multiple inputs. Moreover, our noisy error correction model achieves comparable performance with the supervised model via multiple-input decoding on duplicated texts.

In summary, our contributions are: (1) a scalable framework needing no supervision from human annotations to train the correction model; (2) a multi-input attention mechanism incorporating aligning, correcting, and voting on multiple sequences simultaneously for consensus decoding, which is more efficient and effective than existing ensemble methods; and (3) a method that corrects text either with or without duplicated versions, while most existing methods can only deal with one of these cases.

2 Data Collection

We perform experiments on one-best OCR output from two sources: two million issues from the Chronicling America collection of historic U.S. newspapers, which is the largest public-domain full-text collection in the Library of Congress;[1] and three million public-domain books in the Internet Archive.[2]

For supervised training and for evaluation, we aligned manually transcribed texts to these one-best OCR transcripts: 1384 issues of the Richmond (Virginia) Daily Dispatch from 1860–1865 (RDD)[3] and 934 books from 1500–1800 from the

[1] chroniclingamerica.loc.gov: Historical newspapers also constitute the largest digitized text collections in the Australian National Library (Trove) and the Europeana consortium.

[2] https://archive.org/details/texts. Google Books and the Hathi Trust consortium also hold many in-copyright books and require licensing agreements to access public-domain materials.

[3] dlxs.richmond.edu/d/ddr/: the transcription from the University of Richmond includes all articles but only some advertisements.

Text Creation Partnership (TCP).[4] Both of these manually transcribed collections, which were produced independently from the current authors, are in the public domain and in English, although both Chronicling America and the Internet Archive also contain much non-English text.

To get more evidence for the correct reading of an OCR'd line, we aligned each OCR'd RDD issue to *other* issues of the RDD and other newspapers from Chronicling America and aligned each OCR'd TCP page to other pre-1800 books in the Internet Archive. To perform these alignments between noisy OCR transcripts efficiently, we used methods from our earlier work on text-reuse analysis (Smith et al., 2014; Wilkerson et al., 2015). An inverted index of hashes of word 5-grams was produced, and then all pairs from different pages in the same posting list were extracted. Pairs of pages with more than five shared hashed 5-grams were aligned with the Smith-Waterman algorithm with equal costs for insertion, deletion, and substitution, which returns a maximally aligned subsequence in each pair of pages (Smith and Waterman, 1981). Aligned passages that were at least five lines long in the target RDD or TCP text were output. For each target OCR line—i.e., each line in the training or test set—there are thus, in addition to the ground-truth manual transcript, zero or more **witnesses** from similar texts, to use the term from textual criticism.

In our experiments on OCR correction, each training and test example is a line of text following the layout of the scanned image documents[5]. The average number of characters per line is 42.4 for the RDD newspapers and 53.2 for the TCP books. Table 2 lists statistics for the number of OCR'd text lines with manual transcriptions and additional witnesses. 43% of the manually transcribed lines have witnesses in the RDD newspapers, and 64% of them have witnesses in the TCP books. In the full Chronicling America data, 44% of lines align to at least one other witness. Although not all OCR collections will have this level of repetition, it is notable that these collections, which are some of the largest public-domain digital libraries, do exhibit this kind of reprinting. Similarly, at least 25% of the pages in Google's web crawls are duplicates (Henzinger, 2006). Although we exploit text reuse, where available, to improve decoding and unsupervised training, we also show (Table 5) significant improvements to OCR accuracy with only a single transcript.

Dataset	# Lines w/manual	# Lines w/manual & witnesses
RDD	2.2M	0.95M (43%)
TCP	8.6M	5.5M (64%)

Table 2: Statistics for the number of OCR'd lines in million (M) from the Richmond Dispatch and TCP Books with manual transcriptions (Column 1) or with both transcriptions and multiple witnesses (Column 2).

3 Methods

In this section, we first define our problem in §3.1, followed by model description. In general, we train an OCR error correction model via an attention-based RNN encoder-decoder, which takes a single erroneous OCR'd line as input and outputs the corrected text (§3.2). At decoding time, multi-input attention combination strategies are introduced to allow the decoder to integrate information from multiple inputs (§3.3). Finally, we discuss several unsupervised settings for training the correction model in §3.4.

3.1 Problem Definition

Given a line of OCR'd text $\mathbf{x}$, comprising the sequence of characters $[x_1, \cdots, x_{T_S}]$, our goal is to map it to an error-free text $\mathbf{y} = [y_1, \cdots, y_{T_T}]$ via modeling $p(\mathbf{y}|\mathbf{x})$. Given $p(\mathbf{y}|\mathbf{x})$, we also seek to model $p(\mathbf{y}|\mathbf{X})$ to search for consensus among duplicated texts $\mathbf{X}$, where $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]$ are duplicated lines of OCR'd text.

3.2 Attention-based Seq2Seq Model

Similar to previous work (Bahdanau et al., 2015), the encoder is a bidirectional RNN (Schuster and Paliwal, 1997) that converts source sequence $\mathbf{x} = [x_1, \cdots, x_{T_S}]$ into a sequence of RNN states $\mathbf{h} = [h_1, \cdots, h_{T_S}]$, where $h_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$ is a concatenation of both forward and backward hidden states at time step $i (1 \leq i \leq T_S)$. We have

$$\overrightarrow{h}_i = f(x_i, \overrightarrow{h}_{i-1}); \quad \overleftarrow{h}_i = f(x_i, \overleftarrow{h}_{i+1}), \quad (1)$$

here f is the dynamic function, e.g., LSTM (Hochreiter and Schmidhuber, 1997) or GRU (Cho et al., 2014).

The decoder RNN predicts the output sequence $\mathbf{y} = [y_1, \cdots, y_{T_T}]$, through the following dynamics and prediction model:

$$s_t = f(y_{t-1}, s_{t-1}, c_t); \quad (2)$$

[4] www.textcreationpartnership.org

[5] The datasets can be downloaded from http://www.ccs.neu.edu/home/dongrui/ocr.html

$$p(y_t|y_{<t}, \mathbf{x}) = g(y_{t-1}, s_t, c_t), \quad (3)$$

where s_t is the RNN state and c_t is the context vector at time t. y_t is the predicted symbol from the target vocabulary at time t via prediction function $g(\cdot)$. The context vector is given as a linear combination of the encoder hidden states:

$$c_t = \sum_{i=1}^{T_S} \alpha_{t,i} h_i; \quad \alpha_{t,i} = \frac{e^{\eta(s_{t-1}, h_i)}}{\sum_\tau e^{\eta(s_{t-1}, h_\tau)}} \quad (4)$$

where $\alpha_{t,i}$ is the weight for each hidden state h_i and η is the function that computes the strength of each encoder hidden state according to current decoder hidden state. The loss function is the cross-entropy loss per time step summed over the output sequence y:

$$L(\mathbf{x}, \mathbf{y}) = -\sum_{t=1}^{T_S} \log p(y_t|\mathbf{x}, y_{<t}) \quad (5)$$

3.3 Multi-input Attention

Given a trained Seq2Seq model $p(\mathbf{y}|\mathbf{x})$, our goal is to combine multiple input sequences $\mathbf{X}$ to generate the target sequence $\mathbf{y}$, i.e., to utilize information from multiple sources at decoding time. Assume that N relevant source sequences $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]$ are observed, where each sequence $\mathbf{x}_l = [x_{l,1}, \cdots, x_{l,T_l}]$ $(1 \le l \le N)$ and T_l is the length of the l^{th} sequence. Then, a sequence of hidden states $\mathbf{h}_l = [h_{l,1}, \cdots, h_{l,T_l}]$ is generated by the encoder network for each input sequence $\mathbf{x}_l$. At each decoding time step t, the decoder searches through encoder hidden states $\mathbf{H} = [\mathbf{h}_1, \cdots, \mathbf{h}_N]$ to compute a global context vector c_t. Different strategies to combine attention from multiple encoders are described as follows.

Flat Attention Combination. Flat attention combination assigns a weight $\alpha_{t,l,i}$ to each encoder hidden state $h_{l,i}$ for each input sequence $\mathbf{x}_l$ as:

$$\alpha_{t,l,i} = \frac{e^{\eta(s_{t-1}, h_{l,i})}}{\sum_{l'=1}^{N} \sum_{\tau=1}^{T_{l'}} e^{\eta(s_{t-1}, h_{l',\tau})}}. \quad (6)$$

Therefore, the global context vector is given by

$$c_t = \sum_{l=1}^{N} \sum_{i=1}^{T_l} \alpha_{t,l,i} h_{l,i}. \quad (7)$$

Flat attention combination is similar to single-input decoding in that it concatenates all inputs into a long sequence, except that the encoder hidden states are computed independently for each input.

Hierarchical Attention Combination. The structure of hierarchical attention combination is presented in Figure 1. We first compute a context vector for each input as:

$$\mathbf{c}_{t,l} = \sum_{i=1}^{T_l} \alpha_{t,l,i} h_{l,i}; \quad \alpha_{t,l,i} = \frac{e^{\eta(s_{t-1}, h_{l,i})}}{\sum_{\tau=1}^{T_l} e^{\eta(s_{t-1}, h_{l,\tau})}}. \quad (8)$$

Then a global context vector c_t is computed as a weighted sum of all the context vectors:

$$c_t = \sum_{i=1}^{N} \beta_{t,l} c_{t,l}, \quad (9)$$

where $\beta_{t,l}$ is the weight assigned to each context vector $c_{t,l}$ and computed in different ways as follows:

(a) *Weighted Attention Combination.* In weighted attention combination, the weight for each context vector is given by its dot product with the decoder state in the transformed common space:

$$\beta_{t,l} = \frac{e^{\eta(s_{t-1}, c_{t,l})}}{\sum_{l'=1}^{N} e^{\eta(s_{t-1}, \mathbf{c}_{t,l'})}}. \quad (10)$$

(b) *Average Attention Combination.* In average attention combination, each input sequence is treated as equally weighted. Thus $\beta_{t,l} = \frac{1}{N}$ for each input sequence $\mathbf{x}_l$. It is more efficient than the weighted attention combination in that it does not need to compute a weight for each input.

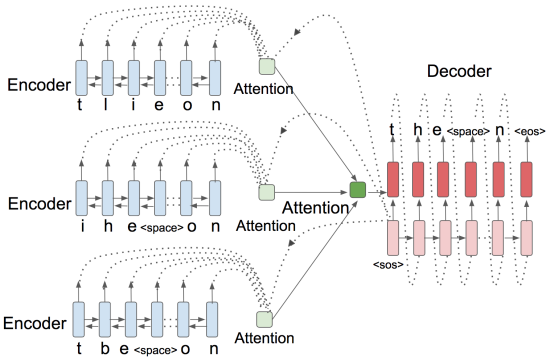


Figure 1: Hierarchical attention combination.

These attention-combination methods do not have parameters trained on multiple inputs and are only introduced at decoding time. In contrast, Libovický and Helcl (2017) and Zoph and Knight (2016) introduce parameters for each type of input and require training and decoding with the same number of inputs.

3.4 Training Settings

In this section, we introduce different settings for training our correction model, a single-input attention-based Seq2Seq model (§3.2), which transforms each OCR'd text line into a corrected version generated via different mechanisms.

Supervised Training. In this setting, the correction model is trained to map each OCR'd line into the corresponding manual transcription, i.e., the human annotation. We call the correction model trained in this setting Seq2Seq-Super.

Unsupervised Training. In the absence of ground truth transcriptions, we can use different methods to generate a noisy corrected version for each OCR'd line.

(a) *Noisy Training.* In this setting, the correction model is trained to transform each OCR'd text line to a selected high-quality witness. The quality of the witnesses is measured by a 5-gram character language model built on the New York Time Corpus (Sandhaus, 2008) with KenLM toolkit (Heafield, 2011). For each OCR'd line with multiple witnesses, a score is assigned to each witness by the language model, divided by the number of characters in it to reduce the effect of the length of a witness. Then a witness with the highest score is chosen as the noisy ground truth for each line. Those lines with low score for all witnesses are removed. We call the correction model trained in this setting Seq2Seq-Noisy.

(b) *Synthetic Training.* In this setting, the error correction model is trained to recover a manually corrupted out-of-domain corpus. We construct the synthetic dataset by injecting uniformly distributed insertion, deletion and substitution errors into the New York Times corpus. Firstly, the news articles are split into lines with random length between $[1, 70]$ following a Gaussian distribution $N(45, 5)$, which is similar to that of the real world dataset. Then, a certain number of lines are randomly selected and injected with equal number of insertion, deletion and substitution errors. The correction model is then trained to recover the original line from each corrupted line. We call this model Seq2Seq-Syn.

(c) *Synthetic Training with Bootstrapping.* In this setting, we propose to further improve the performance of synthetic training via bootstrapping. The correction model trained on synthetic dataset does not perform well when correcting a given input from real world dataset, due to their difference in error distributions. But it achieves comparable performance with the supervised model when decoding lines with multiple witnesses, since the model could further benefit from jointly aligning and voting among multiple inputs. Thus, with the multi-input attention mechanism introduced in §3.3, we first generate a high-quality consensus correction for each OCR'd line with witnesses via the correction model trained on synthetic data. Then, the a bootstrapped model is trained to transform those lines into their consensus correction results. We call the correction model trained in this setting Seq2Seq-Bootstrap.

4 Experiments

In this section, we first introduce the details of our experimental setup (§4.1). Then, the results of preliminary experiments comparing the performance of different options for the single-input Seq2Seq model and the multi-input attention combination strategies are presented in §4.2. The main experimental results for evaluating the correction model trained in different training settings and decoded with/without multi-input attention are reported and explained in §4.3. Further discussions of our model are described in §4.4.

4.1 Experimental Setup

We begin by describing the data split, training details, baseline systems, and evaluation metrics.

4.1.1 Training Details

For both RDD newspapers and TCP books, we randomly split the OCR'd lines into 80% training and 20% test either by the date of the newspaper or by the name of the books. For the RDD newspapers, we have 1.7M training lines and 0.44M test lines. For the TCP books, 2.8M lines are randomly sampled from the whole training set for different training settings to conduct a fair comparison with noisy training, and about 1.6M lines are used for testing.

Both the encoder and decoder of our model has 3 layers with 400 hidden units for each layer, where GRU is applied as the dynamic function. Adam optimizer with a learning rate of 0.0003 and default decay rates is used to train the correction model . We train up to 40 epochs with a mini-batch size of 128 and select the model with the lowest perplexity on the development set. The decoder implements beam search with a beam width of 100.

4.1.2 Baselines and Comparisons

In preliminary experiments, we first compare the neural translation model (§3.2) with a commonly used Seq2Seq model, pruned conditional random fields (PCRF) (Schnober et al., 2016) on the single-input correction task. CRF models have been shown to be very competitve on tasks such as OCR post-correction, spelling correction, and lemmatization. After that, we compare the different multi-input attention strategies introduced in §3.3 on multi-input correction task to choose the best strategy for the main experiments.

In the main experiment, we compare the performance of correction models trained in different training settings and decode with and without multiple witnesses. Two ensembles methods, language model ranking (LMR) and majority vote (Xu and Smith, 2017), are also considered as unsupervised baseline methods. LMR chooses a single high-quality witness for each OCR'd line by a language model as the correction for that line. Majority vote first aligns multiple input sequences using a greedy pairwise algorithm (since multiple sequence alignment is intractable) and then votes on each position in the alignment, with a slight advantage given to the original OCR output in case of ties. We also tried to use an exact unsupervised method for consensus decoding based on dual decomposition (Paul and Eisner, 2012). Their implementation, unfortunately, turned out not to return a certificate of completion on most lines in our data even after thousands of iterations.

4.1.3 Evaluation Metrics

Word error rate (WER) and character error rate (CER) are used to compare the performance of each method. Case is ignored. Lattice word error rate (LWER) and lattice character error rate (LCER) are also computed as the oracle performance for each method, which could reveal the capability of each model to be applied to downstream tasks taking lattices as input, e.g., reranking or retrieval of the correction hypotheses (Taghva et al., 1996; Lam-Adesina and Jones, 2006). We compute the macro average for each type of error rate, which allows us to use a paired permutation significance test.

4.2 Preliminary Experiments

In this section, we conduct two preliminary experiments to study different options for both the single-input correction models and the multi-input attention combination strategies.

4.2.1 Single Input Correction Model

Model	CER	WER
None	0.18133	0.41780
PCRF$_{(order=5,w=4)}$	0.11403	0.25116
PCRF$_{(order=5,w=6)}$	0.11535	0.25617
Attn	**0.11028***	**0.23405***

Table 3: CER and WER on single-input correction for PCRF and Attn-Seq2Seq on RDD newspapers. Results from Attn-Seq2Seq that are significantly better than the PCRF are highlighted with *($p < 0.05$, paired permutation test). The best result for each column is in **bold**.

We first compare the attention-based Seq2Seq (Attn-Seq2Seq) model, with a traditional Seq2Seq model, PCRF, on single input correction task. As the PCRF implementation of Schnober et al. (2016) is highly memory and time consuming for training on long sequences, we compare it with Attn-Seq2Seq model on a smaller dataset with 100K lines randomly sampled from RDD newspapers training set. The trained correction model is then applied to correct the full test set. CER and WER of the correction results from both models are listed in Table 3. We can find that the Attn-Seq2Seq neural translation model works significantly better than the PCRF when trained on a dataset of the same size. The performance of the Attn-Seq2seq model could be further improved by including more training data or by multi-input decoding for duplicated texts, while the PCRF could only be trained on limited data and is not able to work on multiple inputs. Thus, we choose Attn-Seq2Seq as our error correction model.

4.2.2 Multi-input Attention Combination

We also compare different attention combination strategies on a multi-input decoding task. The results from Table 4 reveal that average attention combination performs best among all the decoding strategies on RDD newspapers and TCP books datasets. It reduces the CER of single input decoding by 41.5% for OCR'd lines in RDD newspapers and 9.76% for TCP books. The comparison between two hierarchical attention combination strategies shows that averaging evidence from each input works better than a weighted summation mechanism. Flat attention combination, which merges all the inputs into a long sequence when computing the strength of each encoder hidden state, obtains the worst performance in terms

Decode	RDD Newspapers				TCP Books			
	CER	LCER	WER	LWER	CER	LCER	WER	LWER
None	0.15149	0.04717	0.37111	0.13799	0.10590	0.07666	0.30549	0.23495
Single	0.07199	0.03300	0.14906	0.06948	0.04508	0.01407	0.11283	0.03392
Flat	0.07238	0.02904*	0.15818	0.06241*	0.05554	0.01727	0.13487	0.04079
Weighted	0.06882*	0.02145*	0.15221	0.05375	0.05516	0.01392*	0.1330	0.03669
Average	**0.04210***	**0.01399 ***	**0.09397**	**0.02863***	**0.04072***	**0.01021***	**0.09786***	**0.02092***

Table 4: Results of correcting lines in the RDD newspapers and TCP books with multiple witnesses when decoding with different strategies using the same supervised model. Attention combination strategies that statistically significantly outperform single-input decoding are highlighted with * ($p < 0.05$, paired-permutation test). Best result for each column is in **bold**.

of both CER and WER.

4.3 Main Results

We now present results on the full training and test sets for the Richmond Daily Dispatch newspapers and Text Creation Partnership books. All results are on the same test set. The multi-input decoding experiments have access to additional witnesses for each line, where available, but fall back to single-input decoding when no additional witnesses are present for a given line.

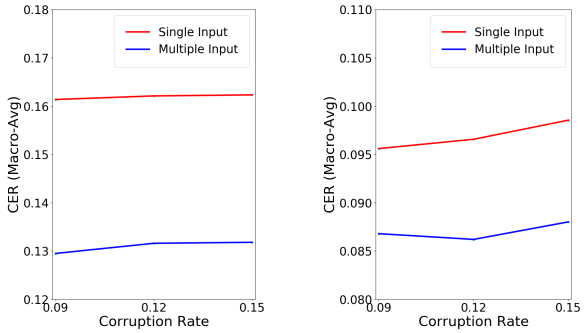
Table 5 presents the results for our model trained in different training settings as well as the baseline language model reranking (LMR) and majority vote methods. Multiple input decoding performs better than single input decoding for every training setting, and the model trained in supervised mode with multi-input decoding achieves the best performance. The majority vote baseline, which works only on more than two inputs, performs worst on both the TCP books and RDD newspapers. Our proposed unsupervised framework Seq2Seq-Noisy and Seq2Seq-Boots achieves performance comparable with the supervised model via multi-input decoding on the RDD newspaper dataset. The performance of Seq2Seq-Noisy is worse on the TCP Books than the RDD newspapers, since those old books contain the character long **s** [6], which is formerly used where **s** occurred in the middle or at the beginning of a word. These characters are recognized as **f** in all the witnesses because of similar shape. Thus, the model trained on noisy data are unable to correct them into **s**. Nonetheless, by removing the factor of long **s**, i.e., replacing the long **s** in the ground truth with **f**, Seq2Seq-Noisy could achieve a CER of 0.062 for single-input decoding and 0.058 for multi-input decoding on the TCP books. Both Seq2Seq-Syn and Seq2Seq-Boots work better on the RDD newspapers than the TCP books dataset. We conjecture that it is because the synthetic dataset is trained on (modern) newspapers, which are more similar to the nineteenth-century RDD newspapers. The long **s** problem also makes it more difficult for the model trained on synthetic data to work on the TCP books.

[6] https://en.wikipedia.org/wiki/Long_s

4.4 Discussion

In this section, we provide further analysis on different aspects of our method.

Does Corruption Rate Affect Synthetic Training? We first examine how the corruption rate of the synthetic dataset would affect the performance of the correction model. Figure 2 presents the results of single-input correction and multi-input correction tasks on the RDD newspapers and TCP books when trained on synthetic data corrupted with different error rate: 0.9, 0.12, 0.15. For both tasks, the character error rate increases a little bit when the correction model is trained to recover the synthetic date with higher corruption rate. However, the performance is more stable on the RDD newspapers than the TCP books when more errors are introduced.



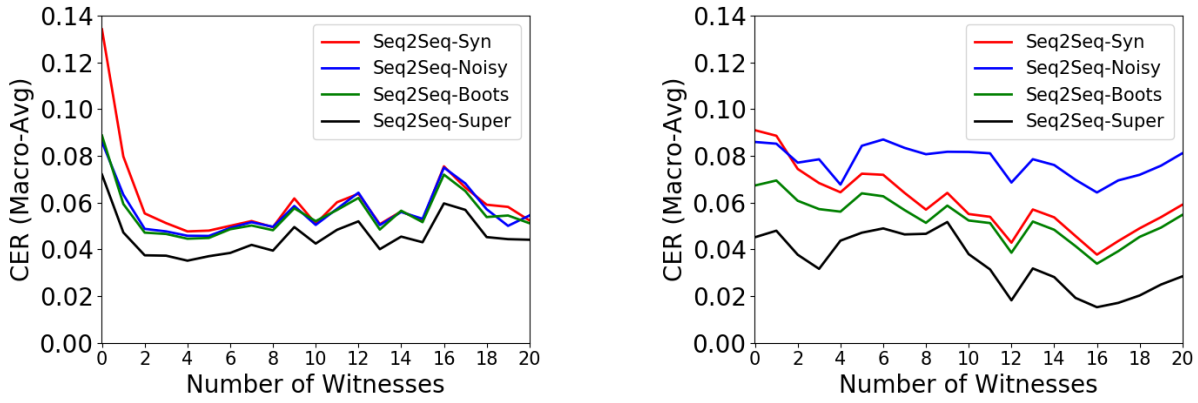
(a) RDD Newspapers (b) TCP Books

Figure 2: Performance of Seq2Seq-Syn trained on synthetic data with different corruption rates.

Does Number of Witnesses Matter for Multiple-Input Decoding? Here we want to study the impact of the number of witnesses on

Decode	Model	RDD Newspapers				TCP Books			
		CER	LCER	WER	LWER	CER	LCER	WER	LWER
	None	0.18133	0.13552	0.41780	0.31544	0.10670	0.08800	0.31734	0.27227
Single	*Seq2Seq-Super*	*0.09044*	*0.04469*	*0.17812*	*0.09063*	*0.04944*	*0.01498*	*0.12186*	*0.03500*
	Seq2Seq-Noisy	0.10524	0.05565	0.20600	0.11416	0.08704	0.05889	0.25994	0.15725
	Seq2Seq-Syn	0.16136	0.11986	0.35802	0.26547	0.09551	0.06160	0.27845	0.18221
	Seq2Seq-Boots	0.11037	0.06149	0.22750	0.13123	0.07196	0.03684	0.21711	0.11233
Multi	LMR	0.15507	0.13552	0.34653	0.31544	0.10862	0.08800	0.33983	0.27227
	Majority Vote	0.16285	0.13552	0.40063	0.31544	0.11096	0.08800	0.34151	0.27227
	Seq2Seq-Super	*0.07731*	*0.03634*	*0.15393*	*0.07269*	*0.04668*	*0.01252*	*0.11236*	*0.02667*
	Seq2Seq-Noisy	**0.09203***	**0.04554***	**0.17940**	**0.09269**	0.08317	0.05588	0.24824	0.14885
	Seq2Seq-Syn	0.12948	0.09112	0.28901	0.19977	0.08506	0.05002	0.24942	0.15169
	Seq2Seq-Boots	0.09435	0.04976	0.19681	0.10604	**0.06824***	**0.03343***	**0.20325***	**0.09995***

Table 5: Results from model trained under different settings on single-input decoding and multiple-input decoding for both the RDD newspapers and TCP books. All training is unsupervised except for supervised results in *italics*. Unsupervised training settings with multi-input decoding that are significantly better than other unsupervised counterparts are highlighted with * ($p < 0.05$, paired-permutation test). Best result among **unsupervised** training in each column is in **bold**.

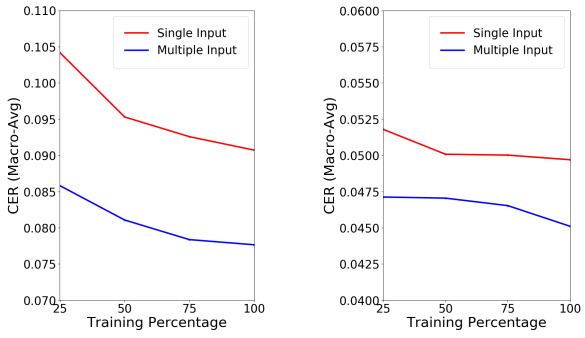


(a) RDD Newspapers (b) TCP Books

Figure 3: Performance of different models on multiple decoding of lines with different number of witnesses.

the performance of multiple-input decoding. The test set is divided into subgroups with varying size according to their number of witnesses. Figure 3 presents the performance of multi-input correction on subgroups with different number of witnesses. We can see that supervised training achieves the best performance on each subgroup for both datasets. On the RDD newspapers, the performance of each training setting is significantly improved when the number of witnesses increases from 0 to 2, then the error rate tends to be flat when more witnesses are observed. For the TCP books, the character error rate for both Seq2Seq-Syn and Seq2Seq-Boots decreases with small fluctuation when the number of witnesses increases. Seq2Seq-Noisy performs the worst almost on all subgroups on the TCP books since all the witnesses suffers from the long **s** problem.

Can More Training Data Benefit Learning? Figure 4 shows the test results for our correction model trained on datasets of different size. As the size of the training set increases, the CER of our model decreases consistently for both single and multiple input correction on the RDD newspapers. However, the performance curve of correction model on TCP books dataset is flatter since it is larger overall than RDD newspapers.



(a) RDD Newspapers (b) TCP Books

Figure 4: Performance of the supervised correction model trained on different proportions of the RDD newspapers and TCP books dataset.

5 Related Work

Multi-Input OCR Correction. Ensemble methods have been shown to be effective in OCR post-correction by combining OCR output from multiple scans of the same document (Lopresti and Zhou, 1997; Klein and Kopel, 2002; Cecotti and Belaïd, 2005; Lund et al., 2013). Existing methods aim at generating consensus results by aligning multiple inputs, followed by supervised methods such as classification (Boschetti et al., 2009; Lund et al., 2011; Al Azawi et al., 2015), or unsupervised methods such as dictionary-based selection (Lund and Ringger, 2009) and voting (Wemhoener et al., 2013; Xu and Smith, 2017). While supervised ensemble methods require human annotation for training, unsupervised selection methods work only when the correct word or character exists in one of the inputs. Furthermore, those methods could not correct single inputs.

Multi-Input Attention. Multi-input attention has already been explored in tasks such as machine translation (Zoph and Knight, 2016; Libovický and Helcl, 2017) and summarization (Wang and Ling, 2016). Wang and Ling (2016) propose to concatenate multiple inputs to generate a summary; this flat attention combination model might be affected by the order of input sequences. Zoph and Knight (2016) aims at developing a multi-source translation model on a trilingual corpus where the encoder for each language is combined to pass to the decoder; however, it requires the same number of inputs at training and decoding time since the parameters depend on the number of inputs. Libovický and Helcl (2017) explore different attention combination strategies for multiple information sources such as image and text. In contrast, our method does not require multiple inputs for training, and the attention combination strategies are used to integrate multiple inputs when decoding.

6 Conclusions

We have proposed an unsupervised framework for OCR error correction, which can handle both single-input and multi-input correction tasks. An attention-based sequence-to-sequence model is applied for single-input correction, based on which a strategy of multi-input attention combination is designed to correct multiple input sequences simultaneously. The proposed strategy naturally incorporates aligning, correcting, and voting among multiple sequences, and is thus effective in improving the correction performance for corpora containing duplicated text. We propose two ways of training the correction model without human annotation by exploiting the duplication in the corpus. Experimental results on historical books and newspapers show that these unsupervised approaches significantly improve OCR accuracy and, when multiple inputs are available, achieve performance comparable to supervised methods.

Acknowledgements

This work was supported by NIH grant 2R01DC009834-06A1, the Andrew W. Mellon Foundation's Scholarly Communications and Information Technology program, and a Google Faculty Research Award. Any views, findings, conclusions, or recommendations expressed do not necessarily reflect those of the NIH, Mellon, or Google. We would like to thank the anonymous reviewers for their valuable comments.

References

Mayce Al Azawi, Marcus Liwicki, and Thomas M. Breuel. 2015. Combination of multiple aligned recognition outputs using WFST and LSTM. In *ICDAR*, pages 31–35.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Federico Boschetti, Matteo Romanello, Alison Babeu, David Bamman, and Gregory Crane. 2009. Improving OCR accuracy for classical critical editions. In *JCDL*, pages 156–167.

Hubert Cecotti and Abdel Belaïd. 2005. Hybrid OCR combination approach complemented by a specialized ICR applied on ancient documents. In *ICDAR*, pages 1045–1049.

Guillaume Chiron, Antoine Doucet, Mickael Coustaty, Muriel Visani, and Jean-Philippe Moreux. 2017. Impact of OCR errors on the use of digital libraries: Towards a better access to information. In *JCDL*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*.

Shamil Chollampatt, Kaveh Taghipour, and Hwee Tou Ng. 2016. Neural network translation models for grammatical error correction. In *IJCAI*.

Kenneth Heafield. 2011. KenLM: Faster and smaller language model queries. In *Proc. Workshop on Statistical Machine Translation*, pages 187–197.

Monika Henzinger. 2006. Finding near-duplicate web pages: A large-scale evaluation of algorithms. In *SIGIR*, pages 284–291.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Rose Holley. 2009. Many hands make light work: Public collaborative OCR text correction in Australian historic newspapers. Technical report, National Library of Australia.

Shmuel T. Klein and Miri Kopel. 2002. A voting system for automatic OCR correction. In *Proc. SIGIR Workshop on Information Retrieval and OCR*.

Okan Kolak, William Byrne, and Philip Resnik. 2003. A generative probabilistic OCR model for NLP applications. In *HLT-NAACL*, pages 55–62.

Okan Kolak and Philip Resnik. 2002. OCR error correction using a noisy channel model. In *HLT*, pages 257–262.

Adenike M. Lam-Adesina and Gareth J. F. Jones. 2006. Examining and improving the effectiveness of relevance feedback for retrieval of scanned text documents. *Information Processing & Management*, 42(3):633–649.

Jindřich Libovický and Jindřich Helcl. 2017. Attention strategies for multi-source sequence-to-sequence learning. In *ACL*.

Daniel Lopresti and Jiangying Zhou. 1997. Using consensus sequence voting to correct OCR errors. *Computer Vision and Image Understanding*, 67(1):39–47.

William B. Lund, Douglas J. Kennard, and Eric K. Ringger. 2013. Combining multiple thresholding binarization values to improve OCR output. In *Proc. Document Recognition and Retrieval (DRR)*.

William B. Lund and Eric K. Ringger. 2009. Improving optical character recognition through efficient multiple system alignment. In *JCDL*, pages 231–240.

William B Lund, Eric K Ringger, and Daniel D Walker. 2014. How well does multiple OCR error correction generalize? In *Proc. Document Recognition and Retrieval (DRR)*.

William B. Lund, Daniel D. Walker, and Eric K. Ringger. 2011. Progressive alignment and discriminative error correction for multiple OCR engines. In *ICDAR*, pages 764–768.

Michael J. Paul and Jason Eisner. 2012. Implicitly intersecting weighted automata using dual decomposition. In *NAACL*, pages 232–242.

Evan Sandhaus. 2008. The New York Times annotated corpus. *Linguistic Data Consortium*, 6(12):e26752.

Carsten Schnober, Steffen Eger, Erik-Lân Do Dinh, and Iryna Gurevych. 2016. Still not there? comparing traditional sequence-to-sequence models to encoder-decoder neural networks on monotone string translation tasks. In *COLING*.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

David A. Smith, Ryan Cordell, Elizabeth Maddock Dillon, Nick Stramp, and John Wilkerson. 2014. Detecting and modeling local text reuse. In *JCDL*.

T. F. Smith and M. S. Waterman. 1981. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197.

Kazem Taghva, Julie Borsack, and Allen Condit. 1996. Effects of ocr errors on ranking and feedback using the vector space model. *Information Processing & Management*, 32(3):317–327.

Lu Wang and Wang Ling. 2016. Neural network-based abstract generation for opinions and arguments. In *NAACL*, pages 47–57.

David Wemhoener, Ismet Zeki Yalniz, and R. Manmatha. 2013. Creating an improved version using noisy OCR from multiple editions. In *ICDAR*, pages 160–164.

John Wilkerson, David A. Smith, and Nick Stramp. 2015. Tracing the flow of policy ideas on legislatures: A text reuse approach. *American Journal of Political Science*.

Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y Ng. 2016. Neural language correction with character-based attention. *arXiv preprint arXiv:1603.09727*.

Shaobin Xu and David A. Smith. 2017. Retrieving and combining repeated passages to improve OCR. In *JCDL*.

Takafumi Yamazoe, Minoru Etoh, Takeshi Yoshimura, and Kousuke Tsujino. 2011. Hypothesis preservation approach to scene text recognition with weighted finite-state transducer. In *ICDAR*, pages 359–363.

Barret Zoph and Kevin Knight. 2016. Multi-source neural translation. In *NAACL*.