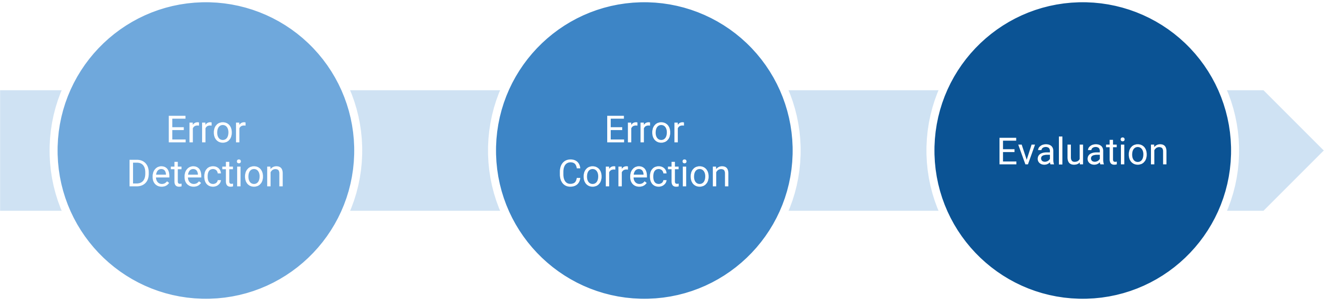
IMPROVEMENT ON POST-PROCESSING FOR OCR DATA

Improvement based on **D3** and **Statistical Learning for OCR Text Correction**

Group members: Yang Cai, Yunsheng Ma, Jiaxi Wu, Huiming Xie, Jiaqian Yu

2

SUMMARY



3

ERROR DETECTION

Based on

- Paper D3
- Statistical Learning for OCR Text Correction

4 COMPARISON OF CLASSIFICATION MODELS

	Random forest	SVM	Xgboost	GBM	Logistics Regression
Time	<5 mins	>30 mins	<5mins	<10 mins	<5mins
Precision	89%	82%	88%	88%	82%
Recall	90%	86%	84%	85%	80%
Additional benefit	❏ Feature selection ❏ Deal with interaction term ❏ Robust	❏ Deal with interaction term ❏ Do not have feature selections ❏ Inefficient to train	❏ Features selection ❏ Deal with interaction term		

5

ERROR DETECTION

Model Comparison

- ❏ SVM
- ❏ Random Forest
- ❏ Xgboost
- ❏ GBM
- ❏ Logistics Regression

SVM (Original)

- Involve expanding feature spaces: adding interaction terms
- Do not have feature selections
- Inefficient to train

Random Forest (Improved)

- Handles high dimensional spaces and large training samples well
- Train faster than SVM (SVM takes **30 min**, RF less than **10 min**)
- Generate a robust estimate

6

ERROR DETECTION

Based on

- Paper D3 Sec.5
- Statistical Learning for OCR Text Correction Sec. 4.1

1. the length l of the input string
2. the number v of vowels and the number c of consonants in the string, as well as the quotients v/l, c/l, v/c (for c = 0)
3. ~~the number of special (non alphanumerical) symbols s and the quotient s/l~~
4. the number of digits d and the quotient d/l
5. the number of lowercase letters low, ~~the number of uppercase letters upp~~, and the quotients low/l, upp/l
6. ~~for strings containing a sequence of at least three consecutive occurrences of the same symbol, we use the quotient of the length of the maximal sequence of identical letters divided by l. For other strings the feature receives value 0~~
7. ~~We calculate the number of all alpha numerical symbols la occurring in the string, and the number k of other symbols s. For k > la the value Feature 7 is 1,for other strings the value is 0~~

7

ERROR DETECTION

Based on

- Paper D3 Sec.5
- Statistical Learning for OCR Text Correction Sec. 4.1

8

~~If the input string contains a subsequence of ≥ 6 directly consecutive consonants, Feature 8 received value 1, otherwise value 0~~

9

~~We delete the first and last symbol of the input string. If the remaining infix contained two or more non alpha numerical symbols, Feature 9 receives value 1, and otherwise value 0~~

10

bigram sum(frequency of the ith bigram in the list Lb/10000)/number of bigrams in input string

11

~~We computed the number of occurrences i of the most frequent symbol of the input string of length l. For i ≥ 3 we used i/l as a feature value, for i ≤ 2 the value was set to 0~~

12

Let l1 denote the number of occurrences of alphabetical symbols in the input string, let l2 = l − l1 denote the number of occurrences of all other types of symbols. We used l2/l1 as a feature.

13

Levenshtein distance

8

ERROR DETECTION

Based on
- ❑ Paper D3 Sec.5
- ❑ Statistical Learning for OCR Text Correction Sec. 4.1

14

Consider a common word is less likely to be an error word, the 1-gram frequency of a word should be greater than a frequency threshold. The frequency threshold varies with different word length.

	ngrams	freq	prop
1	the	84	0.04783599
2	of	68	0.03872437
3	and	45	0.02562642
4	1n	35	0.01993166
5	to	30	0.01708428
6	on	28	0.01594533

Most frequent words

	ngrams	freq	prop
989	terr1tory	1	0.0005694761
990	P.	1	0.0005694761
991	no	1	0.0005694761
992	1tse1f	1	0.0005694761
993	Draft	1	0.0005694761
994	Clean	1	0.0005694761

Least frequent words

9

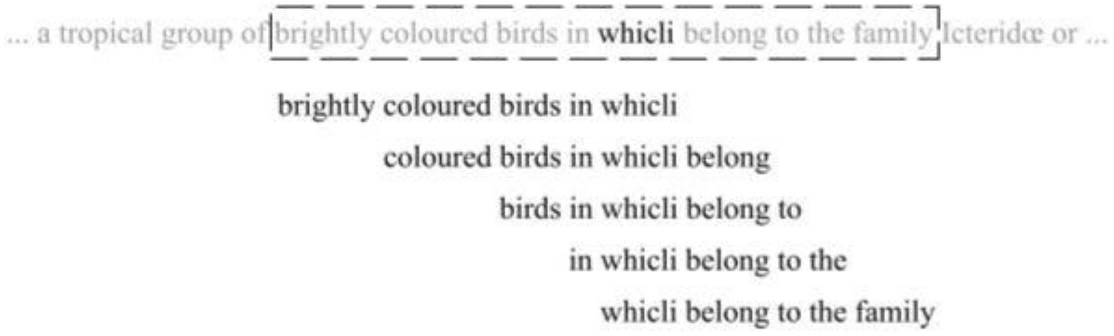
ERROR DETECTION

Based on

- Paper D3 Sec.5
- Statistical Learning for OCR Text Correction Sec. 4.1

15

A word is likely to be correct if this word with its context occurs in other places. We use a sliding window to construct n-gram contexts for a word. The frequency of one of the context in the n-gram corpus should be greater than a frequency threshold.



Example of sliding window of size five

10

ERROR CORRECTION

Based on

- Statistical Learning for OCR Text Correction

11 ERROR CORRECTION

From Paper Statistical Learning for OCR Text Correction Sec 4.2-4.4

Step 1. Candidate Search

Select a candidate for each error according to Levenshtein distance

Step 2. Compute Feature Scores

- Levenshtein edit distance
- String similarity
- Language popularity
- Lexicon existence
- Exact-context popularity

Step 3.Train Model

- Use scores as features
- Label 1 if the candidate is the same as ground truth, label 0 otherwise.

Step 4. Candidate Ranking & Correct

- Rank test set error candidates
- Choose the one with highest score for correction

12

ERROR CORRECTION

Based on

- Statistical Learning for OCR Text Correction Sec 4.2

Step 1. Candidate Search

Candidate Set for a detected error ***we***

$$\{\, w_c \,|w_c \in \mathcal{L}, dist(w_c, w_e) \leq \delta\},$$

Minimum
Levenshtein distance

Distance threshold

13

ERROR CORRECTION

Based on

- Statistical Learning for OCR Text Correction Sec 4.3

Step 2. Compute Feature Scores

1 Levenshtein edit distance

$$score(w_c, w_e) = 1 - \frac{dist(w_c, w_e)}{\delta + 1}$$

2 String similarity

$$nlcs(w_c, w_e) = \frac{2 \cdot len(lcs(w_c, w_e))^2}{len(w_c) + len(w_e)}.$$

14

ERROR CORRECTION

Based on

- Statistical Learning for OCR Text Correction Sec 4.3

Step 2. Compute Feature Scores

2 String similarity (contd.)

$$nmnlcs_1(\mathrm{w}_c, \mathrm{w}_e) = \frac{2 \cdot len(mclcs_1(\mathrm{w}_c, \mathrm{w}_e))^2}{len(\mathrm{w}_c) + len(\mathrm{w}_e)} \tag{4}$$

$$nmnlcs_n(\mathrm{w}_c, \mathrm{w}_e) = \frac{2 \cdot len(mclcs_n(\mathrm{w}_c, \mathrm{w}_e))^2}{len(\mathrm{w}_c) + len(\mathrm{w}_e)} \tag{5}$$

$$nmnlcs_z(\mathrm{w}_c, \mathrm{w}_e) = \frac{2 \cdot len(mclcs_z(\mathrm{w}_c, \mathrm{w}_e))^2}{len(\mathrm{w}_c) + len(\mathrm{w}_e)}. \tag{6}$$

$$\begin{aligned} &score(\mathrm{w}_c, \mathrm{w}_e) \\ &= \alpha_1 \cdot nlcs(\mathrm{w}_c, \mathrm{w}_e) + \alpha_2 \cdot nmnlcs_1(\mathrm{w}_c, \mathrm{w}_e) \\ &+ \alpha_3 \cdot nmnlcs_n(\mathrm{w}_c, \mathrm{w}_e) + \alpha_4 \cdot nmnlcs_z(\mathrm{w}_c, \mathrm{w}_e). \end{aligned}$$

15

ERROR CORRECTION

Based on

- Statistical Learning for OCR Text Correction Sec 4.3

Step 2. Compute Feature Scores

3 Language popularity

$$score(\mathrm{w}_c, \mathrm{w}_e) = \frac{freq_1(\mathrm{w}_c)}{\max_{\mathrm{w}'_c \in C} freq_1(\mathrm{w}'_c)}.$$

4 Lexicon existence

$$score(\mathrm{w}_c, \mathrm{w}_e) = \begin{cases} 1 & \text{if } \mathrm{w}_c \text{ exists in the lexicon} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

16

ERROR CORRECTION

Based on

- Statistical Learning for OCR Text Correction Sec 4.3

Step 2. Compute Feature Scores

5

Context popularity

N gram frequency

$$score(\mathbf{w}_c, \mathbf{w}_e) = \frac{\sum_{\mathbf{c} \in \mathcal{G}_c} freq_n(\mathbf{c})}{\max_{\mathbf{w}'_c \in \mathcal{C}} \{\sum_{\mathbf{c}' \in \mathcal{G}'_c} freq_n(\mathbf{c}')\}} \qquad (10)$$

17

ERROR CORRECTION

Based on

- Statistical Learning for OCR Text Correction

Step 3. Train model

1

Features

Use the scores computed in step 2 as features

2

Labels

Label 1 : if the candidate is the **same** as ground truth

Label 0: if the candidate is **different** from ground truth

3

Model

Random Forest, XGBOOST, adaboost

4

Cross Validation (5-fold)

18

ERROR CORRECTION

Based on
- Statistical Learning for OCR Text Correction Sec 4.4

Step 4. Candidate Ranking & Correction

- Rank test set candidates according to predicted scores
- Choose the one with highest score for correction

19

EVALUATION

20

EVALUATION

- Error Detection

$$\text{precision} = \frac{\text{number of correct items}}{\text{number of items in OCR output}}$$

$$\text{recall} = \frac{\text{number of correct items}}{\text{number of items in ground truth}}$$

	Tesseract (D3 detection)	Tesseract (Improved detection)
word_wise_recall	**0.82**	**0.89**
word_wise_precision	**0.86**	**0.90**

21

EVALUATION

- Error Correction

$$\text{precision} = \frac{\text{number of correct items}}{\text{number of items in OCR output}}$$

$$\text{recall} = \frac{\text{number of correct items}}{\text{number of items in ground truth}}$$

	Tesseract (C4 correction)	Tesseract (improved correction)
word_wise_recall	**0.71**	**0.76**
word_wise_precision	**0.70**	**0.75**
character_wise_recall	**0.88**	**0.90**
character_wise_precision	**0.89**	**0.91**

THANK YOU