GR 5243 Applied Data Science

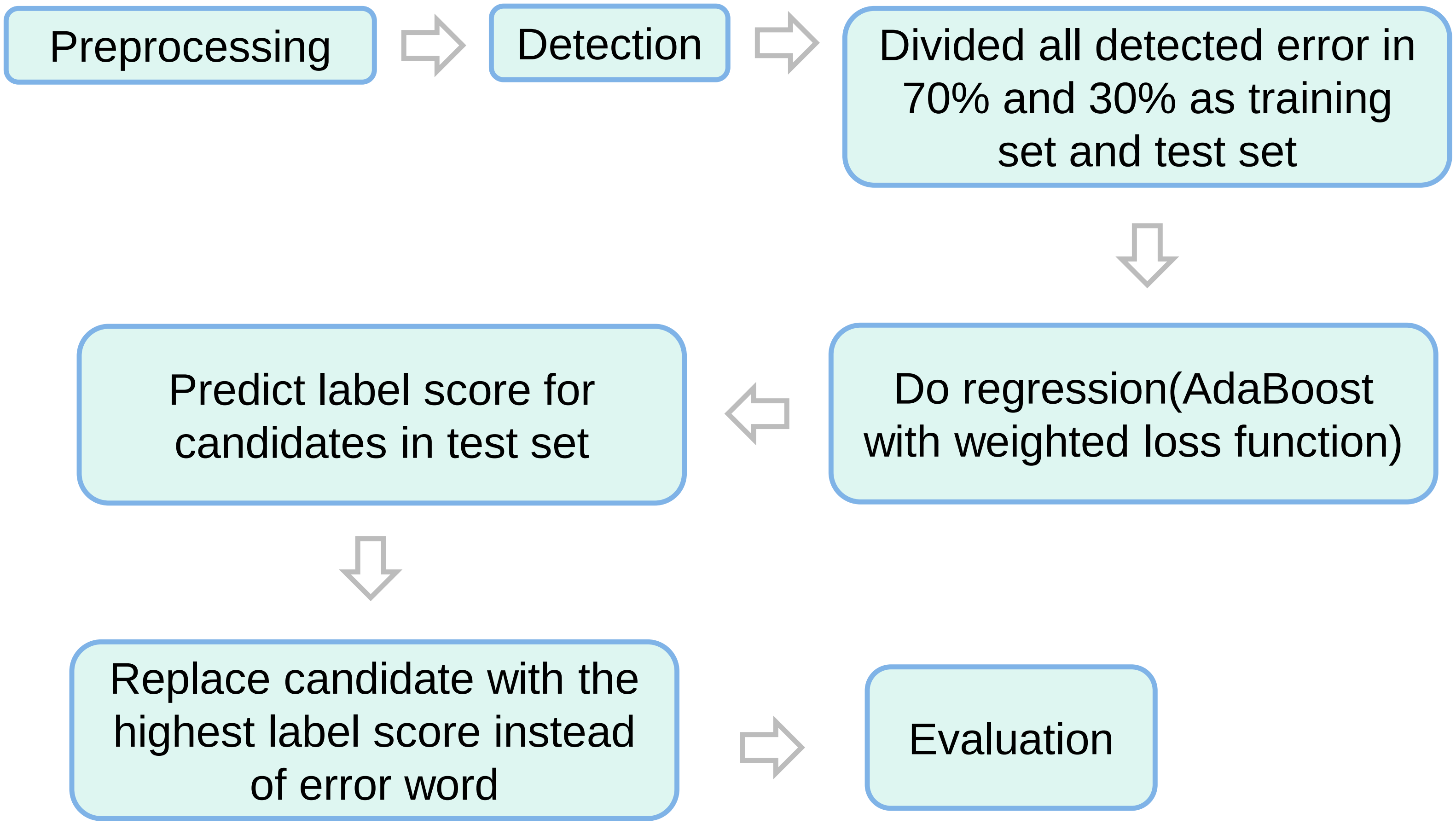
Project 4 Algorithm Implementation and Evaluation

Team Members:

Chen, Jannie (mc4398)
Chen, Sizhu (sc4248)
Li, Yunfan (yl3838)
Xu, Zhengyang (zx2229)
Yu, Chenghao (cy2475)

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

• Flow chart


Preprocessing
Detection
Divided all detected error in 70% and 30% as training set and test set
Do regression(AdaBoost with weighted loss function)
Predict label score for candidates in test set
Replace candidate with the highest label score instead of error word
Evaluation

• Detection: Binary n-Grams Method (D-2)

- **Idea:**
  - Check whether letter pairs in the word at different positions exists in ground truth. If any pairs of letters are not in ground truth, the word is detected as error.
- **Concepts:**
  - n-Grams

A length n word
$W = l_1 l_2 \dots l_k$

⇨

There are $\binom{k}{n}$ n-gram:

$$P_{1,2,\dots,n} = (l_1, l_2, \dots, l_n)$$
$$P_{1,3,\dots,n+1} = (l_1, l_3, \dots, l_{n+1})$$
$$\dots$$
$$P_{k-n,k-n+1,\dots,k} = (l_{k-n}, l_{k-n+1}, \dots, l_{n+1})$$

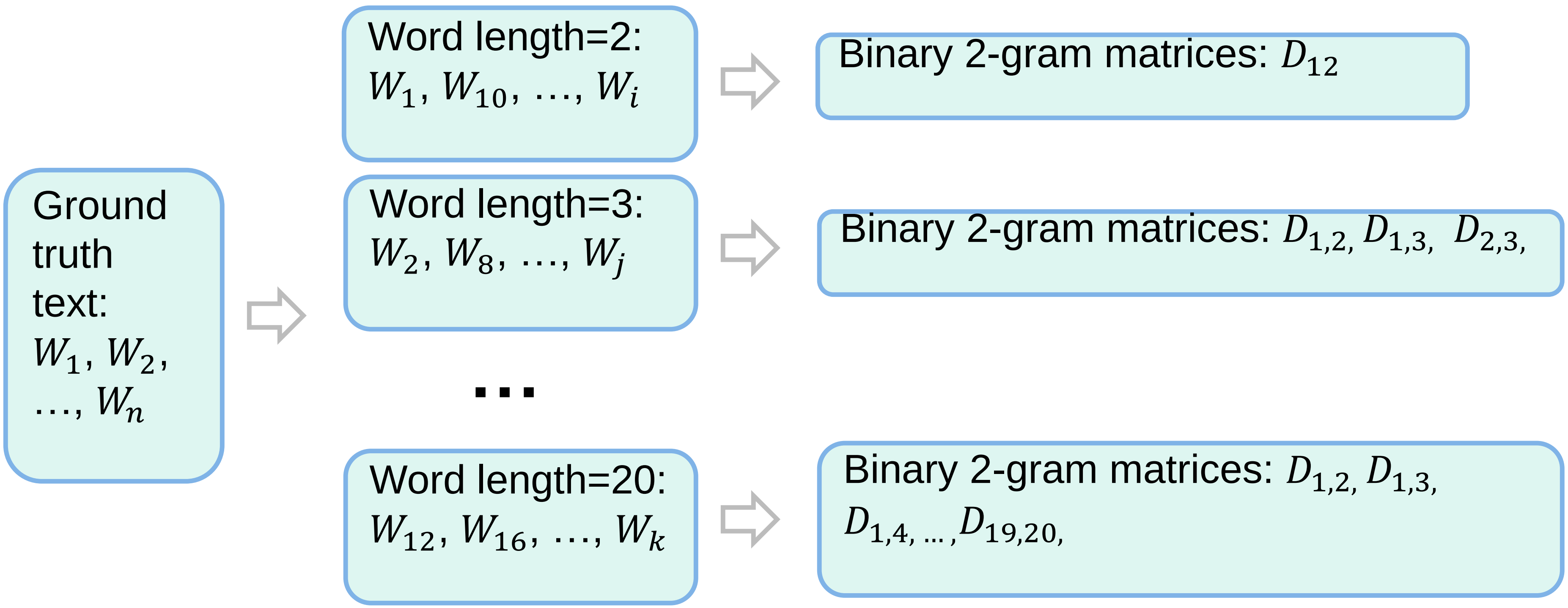
- Detection: Binary n-Grams Method (Cont.)

  – Binary 2-gram Matrix:

$$D_{ij} = \begin{array}{c c} & \begin{array}{cccc} a & b & \dots & z \end{array} \\ \begin{array}{c} a \\ b \\ \vdots \\ z \end{array} & \begin{bmatrix} 1 & 1 & \dots & 1 \\ 0 & 1 & \dots & 1 \\ \dots & & & \dots \\ 1 & 0 & \dots & 0 \end{bmatrix}_{26 \times 26} \end{array}$$

• Detection: Binary n-Grams Method (Cont.)

- **Creating Binary Matrices:**
  - Create all binary 2-gram matrices



Detection: Binary n-Grams Method (Cont.)

- **Detection:**

A new word with length n: W

Get $\binom{n}{2}$ 2-gram letter pairs: $P_{1,2}$, $P_{1,3}$, … , $P_{n-1,n}$

Check letter pairs in binary 2-gram matrices of length n, if $D_{i,j}$ is 0 with $P_{i,j}$ pairs, then $D_{i,j}$ detected an error

Find union set of the locations where detected the error happened, the result is the error location in the word.

Assumption:
Only one error in every ORC output word

• Detection: Binary n-Grams Method (Cont.)

• **Undetectable Case**:

Dictionary	SUT is an undetectable error
SAT	D_{12}: entry for SU is 1 due to SUN
CUT	D_{13}: entry for ST is 1 due to SAT
SUN	D_{23}: entry for UT is 1 due to CUT.

• Detection: Binary n-Grams Method (Cont.)

- Advantages:
  - Saving storage space
  - More efficient than dictionary based methods
  - Extendibility
- Disadvantages:
  - Theoretical Undetectable case
  - Lower detection rate than dictionary based methods
  - Cannot detect errors in non-alphabetic languages

• Correction: Statistical Learning Model (C-2)

- **Idea:**
  - Using a regression model with six feature scores to predict the class of candidates.
- **Concepts:**
  - Candidate Search

Damerau-Levenshtein distance

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

    - δ is chosen to ensure that every W_e has at least 10 candidates W_c and δ is not greater than 20
    - The candidates are chosen from dictionary created from ground truth

• Correction: Statistical Learning Model (C-2)

– Feature scores（some kinds of measurement of similarity between two strings):

- Feature 1：Levenshtein edit distance score:

Levenshtein distance

$$score(\mathrm{w}_c, \mathrm{w}_e) = 1 - \frac{dist(\mathrm{w}_c, \mathrm{w}_e)}{\delta + 1}$$

- δ is set as the same as δ in candidates search so that this score will in [0,1] interval.

Difference between two edit distance:

```
In [194]: damerau_levenshtein_distance("abc", "acb")
Out[194]: 1

In [195]: edit_distance("abc", "acb", substitution_cost=1, transpositions=False)
Out[195]: 2
```

• Correction: Statistical Learning Model (Cont.)

• Feature 2： String similarity score:

C-2 paper:

$$nlcs(\mathrm{w}_c, \mathrm{w}_e) = \frac{2 \cdot len(lcs(\mathrm{w}_c, \mathrm{w}_e))^2}{len(\mathrm{w}_c) + len(\mathrm{w}_e)}.$$

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

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

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

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

Original paper:

$$v_1 = NLCS(s_i, s_j) = \frac{2 \times len(LCS(s_i, s_j))}{len(s_i) + len(s_j)}$$

$$v_2 = NMCLCS_1(s_i, s_j) = \frac{2 \times len(MCLCS_1(s_i, s_j))}{len(s_i) + len(s_j)}$$

$$v_3 = NMCLCS_n(s_i, s_j) = \frac{2 \times len(MCLCS_n(s_i, s_j))}{len(s_i) + len(s_j)}$$

$$v_4 = NMCLCS_z(s_i, s_j) = \frac{2 \times len(MCLCS_z(s_i, s_j))}{len(s_i) + len(s_j)}$$

$$S(s_i, s_j) = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3 + \alpha_4 v_4$$

“$\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are weights and $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 1$. Therefore, the similarity of the two strings, $S \in [0, 1]$”

“We heuristically set equal weights for most of our experiments”

• Correction: Statistical Learning Model (Cont.)

– LCS: *Longest Common Subsequence*

```
In [193]: CS.find_common_subsequences("qweert", "qwwert")
Out[193]:
{'',
 'e',
 'er',
 'ert',
 'et',
 'q',
 'qe',
 'qer',
 'qert',
 'qet',
 'qr',
 'qrt',
 'qt',
 'qw',
 'qwe',
 'qwer',
 'qwert',
 'qwet',
 'qwr',
 'qwrt',
 'qwt',
 'r',
 'rt',
 't',
 'w',
 'we',
 'wer',
 'wert',
 'wet',
 'wr',
 'wrt',
 'wt'}
```

'qwert' ⇦ lcs, $mclcs_1$, $mclcs_z$

'wert' ⇦ $mclcs_n$

• Correction: Statistical Learning Model (Cont.)

• Feature 3：Language popularity score:

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

Dictionary:

	A	B
1	word	freq
2	the	14367
3	and	6905
4	for	2606
5	cma	2188
6	that	1642
7	with	1611
8	will	1349
9	committee	1211
10	this	1098
11	are	1072
12	chemical	1029
13	has	1021

Example:

```
In [239]: dictionary = pd.read_csv("../output/onegram.csv")
     ...: Dictionary = dictionary.set_index('word').T.to_dict("index")['freq']
     ...: Threshold = 1
     ...: We = 'rah'
     ...:
     ...: Candidates = p4.candidate_search(Dictionary, We, Threshold)

In [240]: Candidates
Out[240]:
{'raw': 11,
 'ray': 7,
 'rat': 4,
 'ran': 3,
 'rag': 1,
 'rahn': 1,
 'rak': 1,
 'ral': 1,
 'rath': 1,
 'rch': 1,
 'rnh': 1}
```

• Correction: Statistical Learning Model (Cont.)

- Feature 4：Lexicon existence：

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

- A lexicon is corresponding to a topic, so we created our lexicon by the different report groups.

Group 1:

	A	B
1		dictionary
2	1	exhibit
3	2	d
4	3	staff
5	4	report
6	5	ma
7	6	rch
8	7	by
9	8	william
10	9	j

Group 2:

	A	B
1		dictionary
2	1	exhibit
3	2	d
4	3	report
5	4	to
6	5	the
7	6	board
8	7	of
9	8	directors
10	9	manufacturing

Group 3:

	A	B
1		dictionary
2	1	agenda
3	2	meeting
4	3	of
5	4	the
6	5	mca
7	6	board
8	7	directors
9	8	tab
10	9	opening

Group 4:

	A	B
1		dictionary
2	1	agenda
3	2	meeting
4	3	of
5	4	the
6	5	cma
7	6	board
8	7	directors
9	8	yellowstone
10	9	room

Group 5:

	A	B
1		dictionary
2	1	r
3	2	agenda
4	3	meeting
5	4	of
6	5	cma
7	6	board
8	7	directors
9	8	monday
10	9	and

• Correction: Statistical Learning Model (Cont.)

- Feature 5：Exact-context popularity：

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

5-gram dictionary：

	A	B
1	5-gram	freq
2	of the clean air act	15
3	the toxic substances control act	15
4	of the board of directors	14
5	presented the annual report of	14
6	the annual report of the	14
7	the health and safety committee	14
8	the house ways and means	13
9	meeting of the board of	12
10	house ways and means committee	11
11	impact on the chemical industry	11
12	as a result of the	10
13	biomedical and environmental special programs	10

Not suitable for parallel processing: 5-gram need at least four successive correct words before or after error word to ensure we can find the 5-gram in dictionary given by ground truth.

• Correction: Statistical Learning Model (Cont.)

- Feature 6：Relaxed-context popularity：

... a tropical group of brightly coloured birds in **whicli** belong to the family Icteridœ or ...

brightly coloured birds in whicli

coloured birds in whicli belong

birds in whicli belong to

in whicli belong to the

whicli belong to the family

In 5-gram case, need to find the frequency of whole 4*5=20 relaxed 5-gram.

• Correction: Statistical Learning Model (Cont.)

- Regression model:
  - AdaBoost with weighted loss function to deal with unbalanced label problem

$$loss(\mathcal{D}) = \sum_{e \in \mathcal{E}} \sum_{c \in \mathcal{C}_e^{\mathcal{F}}} w_c \cdot loss(\mathbf{x}_c, y_c).$$

"We count the number of samples with label 1 and 0, respectively. Then, we use the ratio to weight for samples labeled 1, and 1 for samples labeled 0."

• Correction: Statistical Learning Model (Cont.)

- Advantage:
  - High correction rate
- Disadvantage:
  - Cannot correct “separated word case”
  - Need more computational resource

• Challenges

- Preprocessing:
  - How to find the error-ground pair
- Correction:
  - How to set reasonable threshold for candidates search
  - Different formula between papers
  - Weighted loss function in R

• Evaluation

	Tesseract	Tesseract with postprocessing
Word wise recall	0.655063	0.755128
Word wise precision	0.655063	0.755128
Character wise recall	0.898704	0.928360
Character wise precision	0.922792	0.951554

Thank you !

Q&A