Лабораторная работа № 3 по курсу криптографии

Выполнил студент группы М8О-308Б *Галкин Алексей Дмитриевич.*

Условие

Сравнить:

1. Два осмысленных текста на естественном языке.
2. Осмысленный текст и текст из случайных букв.
3. Осмысленный текст и текст из случайных слов.
4. Два текста из случайных букв.
5. Два текста из случайных слов.

Считать процент совпадения букв в сравниваемых текстах – получить дробное значение от 0 до 1 как результат деления количества совпадений на общее число букв. Расписать подробно в отчёте алгоритм сравнения и приложить сравниваемые тексты в отчёте хотя бы для одного запуска по всем пяти случаям. Осознать, какие значения получаются в этих пяти случаях. Привести соображения о том, почему так происходит. Длина сравниваемых текстов должна совпадать. Привести соображения о том, какой длины текста должно быть достаточно для корректного сравнения.

0.1 Основные понятия

Частота совпадений символов — вероятность того, что два символа на одинаковых позициях в разных текстах совпадут:

$$P_{\text{совпад}} = \frac{\text{Число совпавших символов}}{\text{Общая длина текста}} \tag{1}$$

Особенности:

- Естественный язык обладает избыточностью (неравномерное распределение символов)
- Теоретическая вероятность совпадения для случайных английских букв: $\frac{1}{26} \approx 3.85\%$
- Слова из случайных букв ближе к естественному языку, чем полностью случайные символы

0.2 Применение в криптографии

- Оценка стойкости шифров
- Обнаружение стеганографии
- Генерация криптографических ключей
- Анализ случайности ГПСЧ

Метод решения

В качестве осмысленных текстов на естественном языке были взяты «Государь» Н. Макиавелли и «Преступление и наказание» Ф. М. Достоевского, оба на английском языке. Ссылки:
https://www.gutenberg.org/files/1232/1232-0.txt
https://www.gutenberg.org/files/2554/2554-0.txt

Текст из случайных слов генерируется из следующего словаря (чуть больше 466 тысяч английских слов):
https://raw.githubusercontent.com/dwyl/english-words/master/words.txt

Текст из случайных букв генерируется из букв английского алфавита в обоих регистрах и состоит из слов длиной от 3 до 10 знаков.

Алгоритм сравнения: параллельно обходим оба текста, сравниваем знаки на одинаковых позициях. Если знаки совпадают, то увеличиваем счётчик совпавших символов на 1. Потом вычисляем процент совпадения. Сравнение регистрозависимое.

Результат работы программы

```
(.venv) alexey@alexey-Yoga-Slim-7-Pro-14IHU5:~/code/crypta/lab_3$ python3 mai
Case #1: two meaningful texts in natural language.
Text length: 287535
Match: 0.06482341280191976
Case #2: meaningful text and text from random letters.
Text length: 287535
Match: 0.03433599387900603
Case #3: meaningful text and text from random words.
Text length: 287535
Match: 0.05737110264837325
Case #4: two texts from random letters.
Text length: 1000000
Match: 0.0322717
Case #5: two texts from random words.
```

```
Text length: 1000000
Match: 0.05785960000000001
```

Анализ результатов

Case 1-2: Разница между 6.48% и 3.43% объясняется избыточностью естественного языка. Частые буквы (e, t, a) дают больше совпадений.

Case 3: 5.74% для случайных слов подтверждает, что даже случайная комбинация реальных слов сохраняет языковые паттерны.

Case 4-5: Стабильные результаты для больших текстов (3.23% и 5.79%) показывают, что 1 млн символов достаточно для устойчивой статистики.

Листинг программного кода

```
import random
import string
import getopt
import os
import sys

import urllib.request


CNT_RANDOM_TEXTS = 10
LEN_RANDOM_TEXT = 10 ** 6
CASES = 5


USAGE = """
Syntax: main.py [--cases=#]

    Flags:
      cases=#
            Numbers of cases to use. By default all cases are used.

            1 -- two meaningful texts in natural language
            2 -- meaningful text and text from random letters
            3 -- meaningful text and text from random words
            4 -- two texts from random letters
            5 -- two texts from random words

      Example:
```

```
        --cases=1,3

"""


def count_common_letters(text1, text2):
    cnt = 0
    for char1, char2 in zip(text1, text2):
        if char1 == char2:
            cnt += 1
    return cnt


def match_perc(text1, text2):
    return count_common_letters(text1, text2) / len(text1)


def gen_random_letters(n):
    text = ''
    while len(text) < n:
        len_word = random.randint(3, 10)
        word = ''.join(random.choice(string.ascii_letters) for _ in range
        text += ' ' + word
    rem = len(text) - n
    if rem != 0:
        text = text[:-rem]
    return text


def gen_random_words(n):
    # Alternative word list source
    url = 'https://raw.githubusercontent.com/dwyl/english-words/master/wo
    response = urllib.request.urlopen(url)
    words = response.read().decode().splitlines()
    text = ''
    while len(text) < n:
        text += ' ' + random.choice(words)
    rem = len(text) - n
    if rem != 0:
        text = text[:-rem]
    return text
```

```
def case1():
    print("Case #1: two meaningful texts in natural language.")
    url = 'https://www.gutenberg.org/files/1232/1232-0.txt'
    url2 = 'https://www.gutenberg.org/files/2554/2554-0.txt'
    response = urllib.request.urlopen(url)
    text1 = response.read().decode()
    response = urllib.request.urlopen(url2)
    text2 = response.read().decode()
    min_len = min(len(text1), len(text2))
    text1 = text1[:min_len]
    text2 = text2[:min_len]
    print("Text length: {0}".format(min_len))
    print("Match: {0}".format(match_perc(text1, text2)))


def case2():
    print("Case #2: meaningful text and text from random letters.")
    url = 'https://www.gutenberg.org/files/1232/1232-0.txt'
    response = urllib.request.urlopen(url)
    text1 = response.read().decode()
    s = 0
    for i in range(CNT_RANDOM_TEXTS):
        text2 = gen_random_letters(len(text1))
        with open('./tests/case2_text_{0}'.format(i + 1), 'w') as f:
            f.write(text2)
        s += match_perc(text1, text2)
    s /= CNT_RANDOM_TEXTS
    print("Text length: {0}".format(len(text1)))
    print("Match: {0}".format(s))


def case3():
    print("Case #3: meaningful text and text from random words.")
    url = 'https://www.gutenberg.org/files/1232/1232-0.txt'
    response = urllib.request.urlopen(url)
    text1 = response.read().decode()
    s = 0
    for i in range(CNT_RANDOM_TEXTS):
        text2 = gen_random_words(len(text1))
        with open('./tests/case3_text_{0}'.format(i + 1), 'w') as f:
            f.write(text2)
```

```
            s += match_perc(text1, text2)
        s /= CNT_RANDOM_TEXTS
        print("Text length: {0}".format(len(text1)))
        print("Match: {0}".format(s))


def case4():
    print("Case #4: two texts from random letters.")
    s = 0
    for i in range(CNT_RANDOM_TEXTS):
        text1 = gen_random_letters(LEN_RANDOM_TEXT)
        with open('./tests/case4_text1_{0}'.format(i + 1), 'w') as f:
            f.write(text1)
        text2 = gen_random_letters(LEN_RANDOM_TEXT)
        with open('./tests/case4_text2_{0}'.format(i + 1), 'w') as f:
            f.write(text2)
        s += match_perc(text1, text2)
    s /= CNT_RANDOM_TEXTS
    print("Text length: {0}".format(LEN_RANDOM_TEXT))
    print("Match: {0}".format(s))


def case5():
    print("Case #5: two texts from random words.")
    s = 0
    for i in range(CNT_RANDOM_TEXTS):
        text1 = gen_random_words(LEN_RANDOM_TEXT)
        with open('./tests/case5_text1_{0}'.format(i + 1), 'w') as f:
            f.write(text1)
        text2 = gen_random_words(LEN_RANDOM_TEXT)
        with open('./tests/case5_text2_{0}'.format(i + 1), 'w') as f:
            f.write(text2)
        s += match_perc(text1, text2)
    s /= CNT_RANDOM_TEXTS
    print("Text length: {0}".format(LEN_RANDOM_TEXT))
    print("Match: {0}".format(s))


def print_usage(message):
    print(USAGE)
    if message:
        sys.exit('\nFATAL ERROR: ' + message)
```

```
    else:
        sys.exit(1)


def parse_args(args):
    try:
        opts, args = getopt.getopt(args, '', ['help', 'cases='])
    except getopt.GetoptError:
        print_usage('Invalid arguments.')

    cases = [i for i in range(1, CASES + 1)]

    for (opt, val) in opts:
        if opt == '--help':
            print_usage(None)
        elif opt == '--cases':
            try:
                cases = set(map(int, val.split(',')))
            except ValueError:
                print_usage('Cases must be comma separated list.')

            for i in cases:
                if i not in range(1, CASES + 1):
                    print_usage('Incorrect cases')

    return cases


if __name__ == '__main__':
    cases = parse_args(sys.argv[1:])

    for i in cases:
        if i == 1:
            case1()
        elif i == 2:
            case2()
        elif i == 3:
            case3()
        elif i == 4:
            case4()
        elif i == 5:
            case5()
```

Выводы

Проведённый эксперимент позволил выявить ключевые закономерности в статистике текстов. Основной результат показывает, что естественные языки обладают выраженной избыточностью: частота совпадений символов (6.48%) существенно превышает теоретический уровень для случайных данных (3.23%).

Тексты из случайных слов демонстрируют промежуточные значения (5.7-5.8%), сохраняя часть статистических свойств естественного языка. Это подтверждает, что даже при случайной комбинации реальных слов остаются характерные языковые паттерны.

С практической точки зрения, полученные данные подчёркивают важность полного разрушения статистических закономерностей при криптографическом преобразовании текстов. Разница между естественными и случайными последовательностями (более чем в 2 раза) даёт чёткий количественный критерий для оценки криптостойкости алгоритмов.