Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и
информатики»
(СибГУТИ)
Институт информатики и вычислительной техники
Кафедра прикладной математики и кибернетики

Практическая работа №1
по дисциплине «Теория информации»
на тему «Вычисление энтропии Шеннона»

Выполнили:
студенты гр.ИП-014
Обухов А.И.

Проверила:
Старший преподаватель каф. ПМиК
Дементьева Кристина Игоревна

Новосибирск 2024 г.

Цель работы: Экспериментальное вычисление оценок энтропии Шеннона текстов. Изучение свойств энтропии Шеннона.

Язык программирования: C, C++, C#, Python

Результат: программа, тестовые примеры, отчет.

Задание:

1. Для выполнения работы потребуются три текстовых файла с различными свойствами. Объем файлов больше 10 Кб, формат txt.
В первом файле содержится последовательность символов, количество различных символов больше 2 (3,4 или 5). Символы **последовательно и независимо** с равными вероятностями генерируются с помощью датчика псевдослучайных чисел и записываются в файл.
Для генерации второго файла необходимо сначала задать набор вероятностей символов (количество символов такое же, как и в первом файле), а затем **последовательно и независимо** генерировать символы с соответствующей вероятностью и записывать их в файл, вероятности в процессе записи файла не меняются.
В качестве третьего файла необходимо выбрать художественный текст на русском (английском) языке. Для алфавита текста предполагается, что строчные и заглавные символы не отличаются, знаки препинания опущены, к алфавиту добавлен пробел, для русских текстов буквы «е» и «ё», «ь» и «ъ» совпадают.

2. Составить программу, определяющую несколько оценок энтропии созданных текстовых файлов. Вычисление значения по формуле Шеннона **настоятельно рекомендуется** оформить в виде отдельной функции, на вход которой подается массив (список) вероятностей символов, выходной параметр – значение, вычисленное по формуле Шеннона.

Вычислить три оценки энтропии Шеннона для каждого из файлов. Рекомендуется вычисление оценки оформить в виде отдельной функции с параметром имя файла:

Первая оценка H_1. Сначала определить частоты отдельных символов файла, т.е. отношения количества отдельного символа к общему количеству символов в файле. Далее используя полученные частоты как оценки вероятностей, рассчитать оценку энтропии по формуле Шеннона.

Вторая оценка H_2. Определить частоты всех последовательных пар символов в файле. Для того правильной оценки энтропии H_2 пары символов нужно рассматривать с перехлестом.

Пример. Пусть имеется такая последовательность ФЫВАФПРО

Под парами понимаются пары соседних символов, т.е.

ФЫ ЫВ ВА АФ ФП ПР РО

Для подсчета оценки энтропии H_2 необходимо подсчитать частоту каждой пары символов и подставить в формулу Шеннона. Полученное значение оценки энтропии следует разделить на 2.

Третья оценка H_3. Определить частоты всех последовательных троек символов в файле. Для того правильной оценки энтропии H_3 **тройки** символов нужно рассматривать с перехлестом.

Для подсчета оценки энтропии H_3 необходимо подсчитать частоту каждой тройки символов и подставить в формулу Шеннона. Полученное значение оценки энтропии следует разделить на 3.

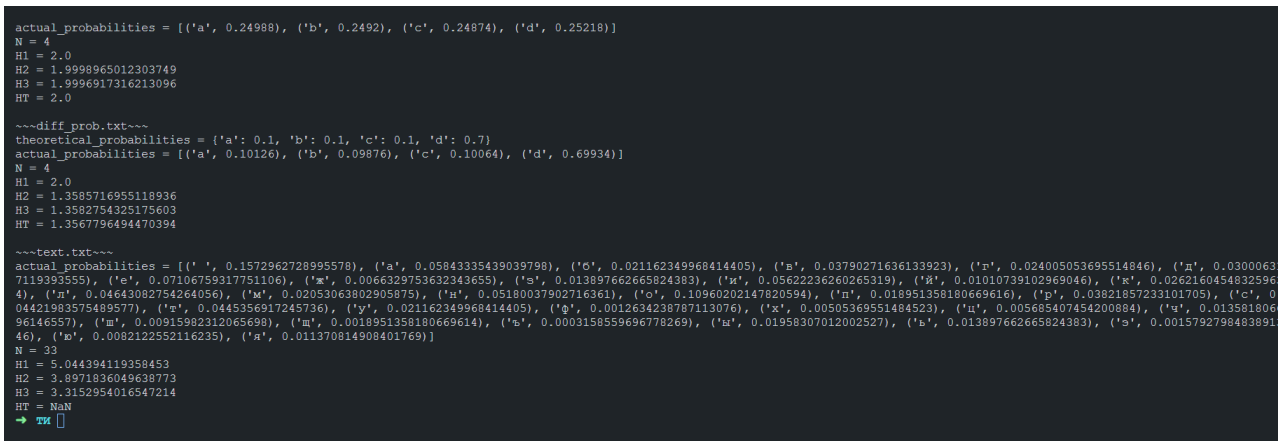
По желанию можно продолжить процесс вычисления оценок с использованием частот четверок, пятерок символов и т.д.

3. После тестирования программы необходимо заполнить таблицу для отчета и в отчете **проанализировать** полученные результаты, объяснить замеченные эффекты. Для получения теоретических значений энтропии использовать наборы вероятностей, которые использовались при

генерации файлов, для файла с текстом на естественном языке не заполнять.

Название файла	H_1	H_2	H_3	Максимально возможное значение энтропии	Теоретическое значение энтропии
equal_prob.txt	2.0	1.9998	1.99969	2.0	2.0
diff_prob.txt	2.0	1.3585	1.3582	2.0	1.3567
text.txt	5.0443	3.8971	3.3152	5.42626	-

Скриншот работы программы:



Анализ результатов работы программы

1. **Генерация файлов**: Реализация основана на генерации псевдослучайных чисел. Для каждого файла определяется набор символов и их вероятностей появления. Затем, с помощью функции random.choices(), символы выбираются случайным образом с учетом указанных вероятностей и записываются в файл заданное количество раз.
2. **Сравнение оценок энтропии**: Полученные оценки энтропии для каждого файла различаются в зависимости от его структуры и свойств. В файлах с равномерным распределением вероятностей символов, таких как equal_prob.txt, значения энтропии оценок примерно

одинаковы, поскольку все символы встречаются с одинаковой вероятностью. В файлах с неравномерным распределением вероятностей, например diff_prob.txt, значения оценок сильно различаются из-за различной частоты встречаемости символов. Эти значения энтропии близки к теоретическим и максимально возможным. Что касается текстов на естественном языке, таких как text.txt, оценки также различны, однако они значительно ниже максимально возможного значения энтропии из-за большого разнообразия символов и их последовательностей.

3. **Поведение последовательности оценок**: При увеличении длины последовательности символов, используемой для оценки энтропии, наблюдается снижение значений энтропии. Это происходит из-за возросшей предсказуемости последовательностей символов. Увеличение длины цепочки символов приводит к анализу более длинных последовательностей, что обычно делает вероятности их появления более сбалансированными или более предсказуемыми.
4. **Значения энтропии**: Исходя из предоставленных данных, для каждого файла вычисляется энтропия Шеннона при различных длинах цепочек символов. Например, в случае файла "text.txt" изначально наблюдается высокая энтропия, близкая к максимально возможному значению. Это может указывать на значительное разнообразие символов и низкую предсказуемость текста. Однако с увеличением длины цепочек символов энтропия начинает уменьшаться. Это может свидетельствовать о том, что более длинные последовательности символов становятся более предсказуемыми и менее разнообразными. Таким образом, значения энтропии Шеннона для текстов из разных файлов могут различаться в зависимости от степени их разнообразия и предсказуемости.

Заключение:

В процессе выполнения практической работы были применены различные методы оценки энтропии Шеннона к трем текстовым файлам. Основная цель работы заключалась в экспериментальном измерении энтропии Шеннона текстов с разнообразными характеристиками и изучении их свойств.

Результаты анализа показали, что значения энтропии Шеннона существенно зависят от разнообразия и предсказуемости символов в тексте. Тексты с большим разнообразием символов и низкой предсказуемостью обладают более высокими значениями энтропии, в то время как тексты с меньшим разнообразием и более предсказуемыми последовательностями имеют более низкие значения энтропии.

Также было обнаружено, что увеличение длины последовательностей символов обычно сопровождается снижением значения энтропии, что указывает на увеличение предсказуемости в тексте.

В общем, результаты работы способствуют лучшему пониманию свойств текстовых данных с использованием энтропии Шеннона. Это может быть полезно для анализа и обработки текстовой информации в различных областях, таких как обработка естественного языка, компьютерная безопасность и теория информации.

Листинг программы

```
from collections import Counter
import csv
import random
import re
from math import log2

def hartley_entropy(alphabet_size) -> float:
    return log2(alphabet_size)

def shannon_entropy(probabilities) -> float:
    return -sum(p * log2(p) for p in probabilities if p != 0)

def calc_entropy(line: str, symb_in_row: int):
    split_line = [line[i: i + symb_in_row] for i in range(len(line) - symb_in_row
+ 1)]
    actual_probability = {k: v / len(line) for k, v in
Counter(split_line).items()}
    result = shannon_entropy(actual_probability.values()) / symb_in_row
    return result, sorted(actual_probability.items())

def generate_file(filename: str, alphabet: dict[str, float], symbols_num: int) ->
None:
    symbols = list(alphabet.keys())
    weights = list(alphabet.values())
    with open(filename, 'w') as f:
        for i in range(symbols_num):
            f.write(''.join(random.choices(symbols, weights)))

def preprocess_file(filename: str, lang: str) -> str:
    with open(filename, 'r', encoding='utf-8') as f:
        line = f.read()
    line = line.lower()
    if lang == "ru":
        line = re.sub(r'[^а-яА-Я0-9 ]', '', line)
    elif lang == "en":
        line = re.sub(r'[^a-zA-Z0-9 ]', '', line)
    else:
        exit(1)
    return line

def main() -> None:
    equal_prob = {'a': 0.25, 'b': 0.25, 'c': 0.25, 'd':0.25}
    diff_prob = {'a': 0.1, 'b': 0.1, 'c': 0.1, 'd':0.7}

    generate_file('./input/equal_prob.txt', equal_prob, 50000)
    generate_file('./input/diff_prob.txt', diff_prob, 50000)

    equal_prob_line = preprocess_file('./input/equal_prob.txt', 'en')
    diff_prob_line = preprocess_file('./input/diff_prob.txt', 'en')
    pre_gen_line = preprocess_file('./input/text.txt', 'ru')
```

```
    print(f"{len(equal_prob_line) = }\n{len(diff_prob_line) =
}\n{len(pre_gen_line) = }\n")

    max_sequential_multiplicity = 3

    def print_entropy(file_name, probabilities, max_sequential_multiplicity):
        for i in range(1, max_sequential_multiplicity + 1):
            entropy, actual_probabilities = calc_entropy(probabilities, i)
            if i == 1:
                print(f"theoretical_probabilities = {probabilities}")
                print(f"actual_probabilities = {actual_probabilities}")
                print(f"N = {len(actual_probabilities)}")
                print(f"H{i} = {hartley_entropy(len(actual_probabilities))}")
                continue
            print(f"H{i} = {entropy}")
        print(f"HT = {shannon_entropy(probabilities.values())}")

    print("\n~~~equal_prob.txt~~~")
    print_entropy('./input/equal_prob.txt', equal_prob_line,
max_sequential_multiplicity)

    print("\n~~~diff_prob.txt~~~")
    print_entropy('./input/diff_prob.txt', diff_prob_line,
max_sequential_multiplicity)

    print("\n~~~text.txt~~~")
    print_entropy('./input/text.txt', pre_gen_line, max_sequential_multiplicity)

    with open('./output/entropy_results.csv', 'w', newline='') as file:
        writer = csv.writer(file)
        writer.writerow(['File name'] + [f'H{i}' for i in range(1,
max_sequential_multiplicity)] + ['Max Possible Entropy', 'Theoretical Entropy'])

        for filename, line in [('./input/equal_prob.txt', equal_prob_line),
('./input/diff_prob.txt', diff_prob_line), ('./input/text.txt', pre_gen_line)]:
            row = []
            h0 = None
            for i in range(1, max_sequential_multiplicity):
                entropy, actual_probabilities = calc_entropy(line, i)
                row.append(entropy)
                if i == 1:
                    h0 = hartley_entropy(len(actual_probabilities))
                if i == max_sequential_multiplicity - 1:
                    row.append(h0)
            if filename == './input/equal_prob.txt':
                row.append(shannon_entropy(equal_prob.values()))
            elif filename == './input/diff_prob.txt':
                row.append(shannon_entropy(diff_prob.values()))
            else:
                row.append('NaN')
```

```
            writer.writerow([filename] + row)

if __name__ == '__main__':
    main()
```