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:** С, С++, С#, 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** *H1*. 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** *H2*. Определить частоты всех последовательных пар символов в файле. Для того правильной оценки энтропии *H2* пары символов нужно рассматривать с перехлестом.   
Пример. Пусть имеется такая последовательность ФЫВАФПРО  
Под парами понимаются пары соседних символов, т.е.

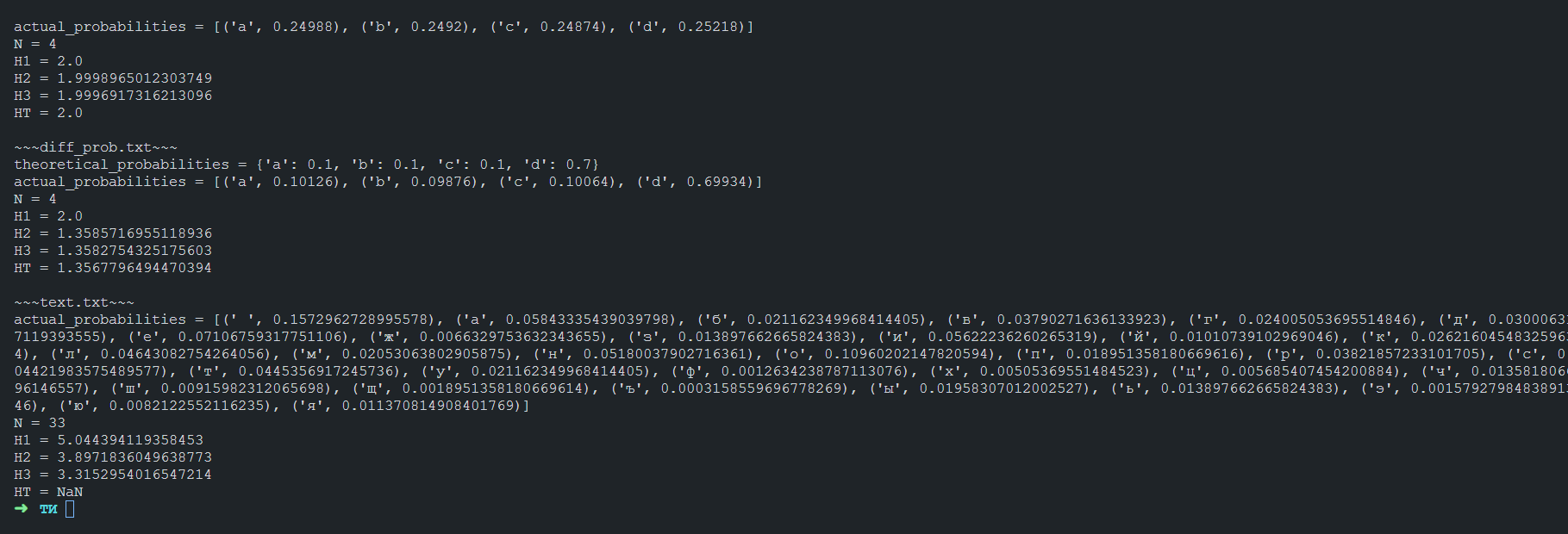
ФЫ ЫВ ВА АФ ФП ПР РО

Для подсчета оценки энтропии *H2* необходимо подсчитать частоту каждой пары символов и подставить в формулу Шеннона. Полученное значение оценки энтропии следует разделить на 2.

**Третья оценка** *H3*. Определить частоты всех последовательных троек символов в файле. Для того правильной оценки энтропии *H3* **тройки** символов нужно рассматривать с перехлестом.   
Для подсчета оценки энтропии *H3* 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 |  |  |  | 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()