

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Комп'ютерний практикум №1

З дисципліни: «Криптографія»

Виконав:
Студент гр. ФБ-03
Гузенков А.М.
Перевірив:
Чорний О.М.

Київ – 2022

Тема

Експериментальна оцінка ентропії на символ джерела відкритого тексту.

Мета роботи

Засвоєння понять ентропії на символ джерела та його надлишковості, вивчення та порівняння різних моделей джерела відкритого тексту для наближеного визначення ентропії, набуття практичних навичок щодо оцінки ентропії на символ джерела.

Постановка задачі

1. Написати програми для підрахунку частот букв і частот біграм в тексті, а також підрахунку H_1 та H_2 за безпосереднім означенням. Підрахувати частоти букв та біграм, а також значення H_1 та H_2 на довільно обраному тексті російською мовою достатньої довжини (щонайменше 1Мб), де імовірності замінити відповідними частотами. Також одержати значення H_1 та H_2 на тому ж тексті, в якому вилучено всі пробіли.
2. За допомогою програми CoolPinkProgram оцінити значення $H^{(10)}$, $H^{(20)}$, $H^{(30)}$.
3. Використовуючи отримані значення ентропії, оцінити надлишковість російської мови в різних моделях джерела.

Хід роботи

Завдання 1

Для виконання першого завдання був написаний код (див. файл main.cpp), який підраховує частоту літер та біграм у тексті та рахує ентропію.

Таблиця з частотою літер:

Літера	Частота	Ймовірність
а	13 536	0,0782071
б	3 029	0,0175007
в	8 379	0,0484114
г	3 362	0,0194247
д	4 858	0,0280681
е	14 979	0,0865443
ж	1 841	0,0106368
з	3 015	0,0174198
и	12 162	0,0702685
й	2 419	0,0139763
к	5 533	0,0319681
л	8 060	0,0465683
м	5 229	0,0302116
н	12 496	0,0721982
о	19 720	0,113936
п	4 362	0,0252024
р	7 552	0,0436333
с	9 552	0,0551887
т	9 920	0,0573149
у	4 448	0,0256992
ф	284	0,00164087
х	1 644	0,00949855
ц	661	0,00381907
ч	2 349	0,0135718
ш	1 213	0,00700836
щ	576	0,00332796
ъ	46	0,000265775

ы	3 270	0,0188931
ь	3 109	0,0179629
э	522	0,00301596
ю	1 190	0,00687547
я	3 760	0,0217242
ё	3	0,0000173331

Таблиця з частотою біграм (перші 10):

Біграма	Частота	Ймовірність
аа	38	0,000439103
аб	132	0,00152531
ав	511	0,00590478
аг	102	0,00117865
ад	165	0,00190663
ае	141	0,0016293
аж	185	0,00213774
аз	430	0,0049688
аи	144	0,00166397
ай	46	0,000531546

Переглянути повну таблицю частот біграм можна у файлі `table_bigrams.csv`.

Результат роботи програми можна побачити нижче

H_1 : 4,45249 — ентропія джерела монограм за фільтрованим текстом

R_1 : 0,109503 — надлишковість джерела монограм за фільтрованим текстом

H_2 : 8,25735 — ентропія джерела біграм за фільтрованим текстом

R_2 : 0,174265 — надлишковість джерела біграм за фільтрованим текстом

Завдання 2:

[illegible]

Лабораторная работа №1

[illegible]

Произвольная часть текста:
 нию_к_кому_не_следует_быть_эгоистичным_только_ли_к_членам_своей_семьи_или_к

Использованные буквы:

Порядок n-граммы:
 5 символов
 10 символов
 15 символов
 20 символов
 25 символов
 30 символов
 35 символов
 40 символов
 45 символов
 50 символов

Введенный символ: o

Символ по счету: 1

Номер эксперимента: 51

Неравенство для энтропии:
 $1.22117915835807 < H < 1.7954265081213$

Двоичная таблица угаданных символов:

10000000000000000000000000000000
10000000000000000000000000000000
10000000000000000000000000000000
10000000000000000000000000000000
10000000000000000000000000000000
10000000000000000000000000000000

Поле ввода символов:
 o

Продолжить Другой

Вероятности:

q[1] = 0.7058823
q[2] = 0.0980392
q[3] = 0
q[4] = 0.0196078
q[5] = 0.0196078
q[6] = 0
q[7] = 0
q[8] = 0
q[9] = 0
q[10] = 0
q[11] = 0
q[12] = 0
q[13] = 0.019607
q[14] = 0
q[15] = 0.019607
q[16] = 0.019607
q[17] = 0.019607
q[18] = 0
q[19] = 0.019607
q[20] = 0
q[21] = 0
q[22] = 0
q[23] = 0
q[24] = 0
q[25] = 0.019607
q[26] = 0
q[27] = 0.019607
q[28] = 0
q[29] = 0.019607
q[30] = 0
q[31] = 0
q[32] = 0

Строка состояния:
 Вы угадали. Для продолжения опыта нажмите "Продолжить", или "Другой" для выбора другого порядка

Обрахуємо надлишковість для $H^{(10)}$, $H^{(20)}$, $H^{(30)}$:

$$R = 1 - \frac{H_{\infty}}{H_0}$$

$$H_0 = \log_2 m = \log_2 32 = 5$$

$$H^{(10)} \approx 1.64536$$

$$R^{(10)} \approx 0.670928$$

$$H^{(20)} \approx 1.89897$$

$$R^{(20)} \approx 0.620206$$

$$H^{(30)} \approx 1.508305$$

$$R^{(30)} \approx 0.698339$$

Труднощі, що виникли під час виконання практикуму

В ході роботи зтикнувся з труднощами кодування вводу-виводу. Рішенням цієї проблеми стало використання строк розширених (`sizeof(wchar_t) = 4` байта у компіляторі GNU) символів та встановлення локалі `ru_RU.UTF8` для всіх потокових об'єктів.

Додаток А

```
#include <cstring>
#include <iostream>
#include <iomanip>
#include <map>
#include <string>
#include <cmath>
#include <fstream>
#include <locale>

std::wstring read_text(std::string filename);
void filter_text(std::wstring& text);
std::map<std::wstring, int> letter_frequency(const std::wstring text);
std::map<std::wstring, int> bigram_frequency(const std::wstring text);
double entropy(std::map<std::wstring, int> ensemble);

int main(int argc, char** argv)
{
    if (argc != 2)
    {
        std::cout << "Usage: " << argv[0] << "<file name>" << std::endl;
        return 1;
    }
    wchar_t* arg = new wchar_t[strlen(argv[1])];
    std::locale mylocale("ru_RU.UTF8");
    std::wcout.imbue(mylocale);
    std::string filename = "plaintext";
    std::wstring text = read_text(filename);
    filter_text(text);
    std::wofstream filtered;
    filtered.open("filtered", std::wios::trunc);
    filtered.imbue(mylocale);
    filtered << text;
    filtered.close();
    std::map<std::wstring, int> letters = letter_frequency(text);
    std::map<std::wstring, int> bigrams = bigram_frequency(text);
    double h1 = entropy(letters);
    double h2 = entropy(bigrams);
    std::wcout << "H1: " << h1 << '\n'
                << "R1: " << 1 - h1/log2(32) << '\n'
                << "H2: " << h2 << '\n'
                << "R2: " << 1 - h2/log2(32*32) << std::endl;
    std::wcout << std::endl;

    unsigned suml{};
    unsigned sumb{};

    for(auto const& x : letters)
        suml += x.second;
    for(auto const& x : bigrams)
        sumb += x.second;

    std::wofstream tablel;
    tablel.open("table_letters.csv", std::wios::trunc);
    tablel.imbue(mylocale);
    tablel << L"Биграмма" << L";" << L"Частота" << L";"
            << L"Вероятность" << L";" << std::endl;
    for(auto const& x : letters)
    {
        tablel << x.first << L";"
                << x.second << L";"
```

```

        << float(x.second) / sum1 << L";"
        << std::endl;
    }
    std::wofstream tableb;
    tableb.open("table_bigrams.csv", std::wios::trunc);
    tableb.imbue(mylocale);
    tableb << L"Биграмма" << L";" << L"Частота" << L";"
        << L"Вероятность" << L";" << std::endl;
    for(auto const& x : bigrams)
    {
        tableb << x.first << L";"
            << x.second << L";"
            << float(x.second) / sumb << L";"
            << std::endl;
    }
    table1.close();
    tableb.close();
}

std::wstring read_text(std::string filename)
{
    std::wifstream input;
    std::locale mylocale("ru_RU.UTF8");    // get global locale
    input.imbue(mylocale);
    input.open(filename, std::wios::in);
    if(!input)
        std::wcout << L"Huston, we've got a problem!" << std::endl;
    std::wstring text;
    while(input)
    {
        std::wstring str;
        input >> str;
        text += str + L" ";
    }
    input.close();
    return text;
}

void filter_text(std::wstring& text)
{
    std::locale mylocale("ru_RU.UTF8");
    std::locale enlocale("en_US.UTF8");
    for (unsigned i{}; i < text.length(); i++)
    {
        if (!std::isalpha(text[i], mylocale))
        {
            text.erase(i, 1);
            if(i != 0) i--;
        }
        else if (text[i] == '\n')
            text[i] = ' ';
        else if(std::isalpha(text[i], mylocale))
        {
            text[i] = std::tolower(text[i], mylocale);
            if(text[i] >= L'a' && text[i] <= L'z')
            {
                text.erase(i, 1);
                if(i != 0) i--;
            }
        }
        if(text[i] == ' ')

```

```

        {
            if(i != 0 && text[i-1] == ' ')
            {
                text.erase(i-1, 1);
                if(i != 0) i--;
            }
        }
    }
}

std::map<std::wstring, int> letter_frequency(const std::wstring text)
{
    std::map<std::wstring, int> freq_map;
    for (unsigned i{}; i < text.length(); i++)
    {
        freq_map[text.substr(i, 1)]++;
    }
    return freq_map;
}

std::map<std::wstring, int> bigram_frequency(const std::wstring text)
{
    std::map<std::wstring, int> freq_map;
    short unsigned last_bg_size;
    last_bg_size = (text.length() % 2 != 0) ? 1 : 0;
    for (unsigned i{}; i < text.length() - last_bg_size; i += 2)
    {
        freq_map[text.substr(i, 2)]++;
    }
    if (last_bg_size == 1)
    {
        std::wstring last_bigram = text.substr(text.length() - 1, 1) +
std::wstring(L" ");
        freq_map[last_bigram]++;
    }
    return freq_map;
}

double entropy(std::map<std::wstring, int> ensemble)
{
    unsigned sum = 0;
    for (auto const& x : ensemble)
        sum += x.second;
    double h = 0;
    for (auto const& x : ensemble)
    {
        h += double(x.second) / double(sum) * log2(double(x.second) /
double(sum));
    }
    return -h;
}

```