1. Каким образом метод персептрона использовался для рубрикации текстов?

В программе использовался объектно-ориентированное программирование.

Исходными данными для программы являются тематические каталоги (имя каталога соответствует названию рубрики) с файлами образцами в данных директориях. Для кодирования исходных данных и дальнейшей обработки были определены классы :

* Класс Rubrics, содержит информацию о глобальном каталоге с тематическими подкаталогами (список рубрик, общее число образцов, метод подсчета количества рубрик):

class Rubrics

{

public List<Rubric> Rubr; // список рубрик (наименования директорий)

public int numberSF; // общее число файлов образцов

public Rubrics()

{ // конструктор

this.Rubr = new List<Rubric>(); // инициализируется новый список рублик

}

public int countSF() // метод подсчета количества рубрик

{

int numberSF = 0;

for (int i = 0; i < this.Rubr.Count(); i++)

{//вход в i-ую рубрику

numberSF += this.Rubr[i].SF.Count();

}

return numberSF;

}

}

* Класс Rubric, содержит информацию о тематическом каталоге с файлами образцами (имя рубрики, список класса файлов образцов, вектор решающей функции данной рубрики, метод инициализации значений для решающей функции)

class Rubric // рубрика

{

public string nameRubric; // имя рубрики

public List<SampleFile> SF; // список класса файлов образцов

public double[] funDecision; // решающая функция

public Rubric()

{ // конструктор

this.SF = new List<SampleFile>(); // инициализируется новый список файлов с текст-образцов

}

public void createFunDecision(int number)

{

this.funDecision = new double[number]; // инициализирует значения для решающей функции

for (int i = 0; i < number; i++)

this.funDecision[i] = 0;

}

}

* Класс SampleFile, содержит информацию о файле-образце данной рубрики (имя файла-образца, все ключевые слова с частотами, вектор признаков, количество ключевых слов в образце, метод инициализации вектор-признака)

class SampleFile // текст-образец из файла

{

public string nameFile; // имя файла-образца

public Dictionary<string, int> wordsDict; // все ключевые слова с частотами файла образца

public double[] vectorCharacter; // вектор признаков

public int numberWords; // количество ключевых слов в образце

public SampleFile()

{ // конструктор

this.wordsDict = new Dictionary<string, int>(); // инициализируется новый список слов

}

public void createVectorCharacter(int number)

{

this.vectorCharacter = new double[number]; // инициализирует вектор-признаков

for (int i = 0; i < number; i++)

this.vectorCharacter[i] = 0;

}

}

В основном модуле программы для инициализации данных по соответствующим классам, а также генерации значений словарей терминов, обобщенного вектора признака, векторов-признаков файлов-образцов и на основе этих данных построения методом персептрона разделяющих функций (весовые коэффициенты) по числу заданных рубрик, разработаны соответствующие функции :

- слова и частота вхождения в файл

private Dictionary<string, int> wordsfreqLawsZipf(string fileName)

{

Dictionary<string, int> wordsDict = new Dictionary<string, int>(); // коллекция слов и их количество в тексте

StreamReader sr = new StreamReader(fileName, System.Text.Encoding.UTF8);

string line;

while ((line = sr.ReadLine()) != null)

{

// приводим к нижнему регистру и разбиваем входную строку на массив слов

string[] wordsArray = line.ToLower().Split(delimiterChars);

foreach (string word in wordsArray) // обработка массива слов

{

string wordtmp = word.Trim(' '); // убираем пробелы

if (wordtmp != "" && !stopwords.Contains(wordtmp))

{

if (wordsDict.ContainsKey(wordtmp)) wordsDict[wordtmp]++;

else wordsDict.Add(wordtmp, 1);

}

}

}

sr.Close();

return wordsDict;

}

- формирование структуры описания рубрик

private void сategorization()

{

Rs = new Rubrics();

DirectoryInfo di = new DirectoryInfo(curDir); // текущий директорий

DirectoryInfo[] diSourceSubDir = di.GetDirectories(); // поддиректории тематической классификации заданных рубрик

foreach (DirectoryInfo subdi in diSourceSubDir)

{

Rubric R = new Rubric();

R.nameRubric = subdi.Name; // имя рубрики

FileInfo[] filesTxt = subdi.GetFiles("\*.txt");

for (int i = 0; i <= filesTxt.Length - 1; i++)

{

SampleFile Rsf = new SampleFile();

Rsf.nameFile = filesTxt[i].FullName;

wordsLawsZipf((int)numUpDnB.Value, ref Rsf);

R.SF.Add(Rsf);

}

Rs.Rubr.Add(R);

}

}

- создаем вектор признаков для данной коллекции ключевых слов

private double[] initVectorCharacter(int amount, Dictionary<string, int> wordsD)

{

double[] VC = new double[amount];

int index = 0;

for (int i = 0; i < Rs.Rubr.Count(); i++)

{//вход в i-ую рубрику

for (int j = 0; j < Rs.Rubr[i].SF.Count(); j++)

{ //вход в j-ый образец

for (int k = 0; k < Rs.Rubr[i].SF[j].wordsDict.Count(); k++)

{

if (wordsD.ContainsKey(Rs.Rubr[i].SF[j].wordsDict.Keys.ElementAt(k)))

VC[index++] = (double)Rs.Rubr[i].SF[j].wordsDict.Values.ElementAt(k) / Rs.Rubr[i].SF[j].numberWords;

else VC[index++] = 0;

}

}

}

VC[index] = 0.001; // дополнительно одна вещественная константа

return VC;

}

- метод обучения персептрона

private void methodPerceptron()

{

bool flag, flagD; // требуется ли обучение

double[] val = new double[Rs.Rubr.Count()]; // для результатов значений решающих функций

do

{

flag = false;

for (int i = 0; i < Rs.Rubr.Count(); i++)

{//вход в i-ую рубрику

for (int j = 0; j < Rs.Rubr[i].SF.Count(); j++)

{ //вход в j-ый образец

flagD = false; //требуется ли корректировка решающей функции

for (int l = 0; l < Rs.Rubr.Count(); l++)

val[l] = presentForm(ref Rs.Rubr[i].SF[j].vectorCharacter, ref Rs.Rubr[l].funDecision);

for (int l = 0; l < Rs.Rubr.Count(); l++)

{

if (l != i) // разные рубрики

{

if (val[i] <= val[l])

{

flag = true; // присутствует ошибка

flagD = true;

decFD(ref Rs.Rubr[i].SF[j].vectorCharacter, ref Rs.Rubr[l].funDecision); // уменьшить РФ

}

}

}

if (flagD) addFD(ref Rs.Rubr[i].SF[j].vectorCharacter, ref Rs.Rubr[i].funDecision); // увеличить РФ

}

}

}

while (flag);

}

// вычисление для объекта VC и решающей функции FD

private double presentForm(ref double[] VC, ref double[] FD)

{

double result = 0;

for (int i = 0; i < (Rs.numberSF \* (int)numUpDnB.Value + 1); i++)

{

result += VC[i] \* FD[i];

}

return result;

}

// добавление к решающей функции FD на объект VC

private void addFD(ref double[] VC, ref double[] FD)

{

for (int i = 0; i < (Rs.numberSF \* (int)numUpDnB.Value + 1); i++)

{

FD[i] += VC[i];

}

}

// уменьшение решающей функции FD на объект VC

private void decFD(ref double[] VC, ref double[] FD)

{

for (int i = 0; i < (Rs.numberSF \* (int)numUpDnB.Value + 1); i++)

{

FD[i] -= VC[i];

}

}

Таким образом метод персептрона реализован как итерационный метод (в цикле do {} while (flag);), в соответствии с алгоритмом метода персептрона перебираются векторы признаки соответствующих рубрик с одной дополнительной вещественной константой (0,001), используя решающее правило происходит уменьшение или увеличение решающей функции (весовые коэффициенты) пока не наступит итерация где не будут произведены изменения текущей решающей функции.

1. Как были построены разделяющие функции?

Для каждой рубрики (class Rubric) были построены разделяющие функции методом персептрона (функция и описание приведены в п.1)

1. Какой тип обучения в методе персептрона?

Это обучение с коррекцией ошибки. Представляет собой такой тип метода обучения, при котором весовые коэффициенты не изменяются до тех пор, пока текущая реакция персептрона остается правильной. При появлении неправильной реакции вес изменяется на единицу, а знак (+/-) определяется противоположным от знака ошибки.

Преподаватель :

"Здравствуйте. Я зачла ваши работы, хотя ответы на мои вопросы мне не очень понравились. 1. Меня интересовала суть использования персептрона для рубрикации текстов. Ответ я ожидала приблизительно такой: на основе обучающей выборки методом персептрона строились разделяющие функции для каждого класса, с помощью которых затем классифицировались все тексты. 2. Во втором вопросе я спрашивала об особенностях построения функций методом персептрона. 3. Типы обучения бывают "с учителем" и "самообучение". Здесь - с учителем, т.к. дана обучающая выборка в виде текстов-образцов."

Пояснения к ответу преподавателя ИОФС :

Здравствуй Влад. Первый вопрос расписан применительно к программе, даже детальнее чем вариант преподавателя, но именно такой алгоритм как и привела преподаватель. Второй вопрос был : "2. Как были построены разделяющие функции?" На этот вопрос и ответ, причем применительно к программе расписан в первом пункте. Вопрос "об особенностях построения функций методом персептрона" (звучит по другому, но можно третий ответ использовать для этого ). По третьему вопросу, понятие "тип обучения" явно в теории не упоминалось, поэтому если было б предложено выбрать "с учителем" или "самообучение", ответ был бы именно в этом направлении... В общем, спорить бессмысленно, поэтому просто придется согласится с замечаниями и сослаться, что думал вопросы по программе.